Universitat Oberta de Catalunya

uoc.edu

Implementación de una plataforma de respuesta a incidentes de seguridad

Víctor Juidias Rodríguez
Máster Universitario en Seguridad de las Tecnologías de la Información y de las Comunicaciones (MISTIC)
Seguridad Empresarial

Miguel Ángel Flores Terrón
Víctor García Font

10/01/2023

B) GNU Free Documentation License (GNU FDL)

C) Copyright

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Implementación de una Plataforma de respuesta a incidentes de seguridad*
Nombre del autor:	*Víctor Juidias Rodríguez*
Nombre del consultor/a:	*Miguel Ángel Flores Terrón*
Nombre del PRA:	*Víctor García Font*
Fecha de entrega (mm/aaaa):	*01/2023*
Titulación o programa:	Máster Universitario en Tecnologías de la Información y las Comunicaciones (MISTIC)
Área del Trabajo Final:	*Seguridad Empresarial*
Idioma del trabajo:	*Castellano*
Palabras clave	*SIRP, Incident response, SIEM*

Resumen del Trabajo

El presente trabajo describe la puesta en funcionamiento de una plataforma de respuesta ante incidentes de seguridad, utilizando software de código abierto, escalable y cuyo despliegue y configuración se hace de manera automatizada. La idea es desarrollar una plataforma que sirva como prueba de concepto o punto de partida para que pueda ser implementada por cualquier empresa u organización en un entorno real para la gestión de incidentes.

La plataforma debe gestionar un incidente de seguridad durante todo su ciclo de vida, desde que es detectado, analizado, puesto en conocimiento de los analistas para que lo puedan resolver, su documentación y su cierre. Permitiendo, además, compartir la información con terceros para ayudar al resto de la comunidad.

Abstract

This project describes the implementation of a security incident response platform, using open source software, scalable and whose deployment and configuration is done automatically. The idea is to develop a platform that can serve as a proof of concept or starting point so that it can be implemented by any company or organisation in a real environment for incident management.

The platform should manage a security incident throughout its lifecycle, from the moment it is detected, analysed, reported to the analysts so that they can resolve it, its documentation and its closure. It also allows information to be shared with third parties to help the rest of the community.

Índice

Lista de figuras

Tablas

Ilustraciones

1. Introducción

1.1. Contexto y justificación del Trabajo

La digitalización de las empresas siempre ha sido una asignatura pendiente, que la mayoría de las veces se ha quedado más en promesas que en hechos reales. Cuando hablamos de digitalización de las empresas, hablamos no solo de tener presencia en Internet para vender sus productos, sino también de trasladar el día a día de las operaciones del negocio al mundo digital. De pasar del papel a la nube, de los registros escritos a manos a utilizar un ERP. Todo esto cambió cuando llegó el Covid19.

La pandemia del Covid19 ha hecho que muchas empresas tengan ahora que avanzar a marchas forzadas en su proceso de digitalización. En un momento en el que el contacto empresa-cliente solo podía ocurrir a través de Internet, muchas empresas se encontraron con que no estaban en absoluto preparadas. A esto hubo que sumarle la implantación del trabajo remoto, que se ha mantenido, en mayor o menor medida, hasta nuestros días, pero que durante muchos meses fue la única forma de trabajar que existió y muchas empresas, tampoco estuvieron preparadas.

Todo este proceso de transformación digital supone asumir más riesgos en materia de seguridad por parte de las empresas y organizaciones, sobre todo de cara a un mundo que no terminan de conocer del todo, que cada día se hace más complicado y confuso, con más sistemas y aplicaciones y, por ende, con más amenazas. Hace que las empresas estén expuestas a todos los riesgos que conlleva exponerse a Internet, tanto sus propios sistemas internos como sus diferentes canales de negocio. Y cuando se hace a marchas forzadas, el riesgo es todavía mayor.

Cada día, la tecnología avanza un paso más, haciendo que el software y el hardware sean capaces de hacer muchas más cosas que antes, algunas incluso impensables hasta hace nada. Pero todo esto se hace a expensas de hacer el software más complicado, con más capas entre medias que ya no controlan los propios desarrolladores de las aplicaciones, pues terminan dependiendo de terceros, algo que puede suponer que de la noche a la mañana puedan tener un gran agujero de seguridad, como ha ocurrido en casos recientes, como el de Log4shell [1], por ejemplo.

A todo lo anterior hay que sumar el tratamiento de los datos, tanto de los clientes y trabajadores, como los de la empresa en sí. Los ataques cuyo objetivo son el robo de información o el secuestro de los datos están a la orden del día. Además de protegerlos como corresponde, hay que tratarlos también debidamente, esto es, conocer la legislación vigente y aplicarla, como la conocida GDPR europea.

Por tanto, vemos que las empresas se enfrentan a grandes riesgos en cuestiones de seguridad en su proceso de digitalización, riesgos que tienen que saber afrontar pues, de no ser así, mermaría su capacidad de negocio y podrían verse superadas por sus competidores.

En el mundo de la seguridad informática se tiene muy claro que no se puede pensar en términos de si seré o no víctima de un ciberataque, sino cuándo sucederá y si seré capaz de defenderme y recuperarme. Si soy víctima de un ciberataque tengo que ser capaz de mitigarlo en el menor tiempo posible, sufriendo los menos daños posibles e intentar recuperar el servicio lo antes posible. Como vemos, lograr todo esto es muy complicado, pero no imposible, pues la clave está en la previsión, la planificación y la monitorización de los sistemas. Por eso es muy importante que las empresas dispongan de un plan de respuesta ante incidentes, no ya de seguridad, sino ante incidentes informáticos que puedan ocurrir, como es que falle un servidor crítico, que tengan un problema con la red, con el almacenamiento o con un proveedor. Todos estos problemas también ocasionan un daño importante al negocio y a los datos de este.

En cuanto al tema que nos interesa, la seguridad, es de vital importancia disponer de un plan de estas características. Un plan de respuestas a incidentes engloba desde la preparación, la detección y el análisis, a la contención, neutralización, recuperación y a la actividad post incidente, donde se analiza lo ocurrido.

Tener un plan de estas características nos permite actuar ante los posibles incidentes que puedan ocurrir, nos permite seguir un plan, un protocolo establecido y no improvisar sobre la marcha, nos permite saber cuáles son los activos más importantes de la empresa, cuáles son las infraestructuras críticas, donde están localizadas, como protegerlas y como recuperarlas.

Es por eso por lo que es de vital importancia disponer de una plataforma que permita gestionar los incidentes de seguridad que se vayan detectando. Una infraestructura de estas características puede ayudar al personal especializado, analistas, gente de sistemas, etc. a detectar las amenazas, contenerlas y responder a ellas de la mejor forma posible.

Hoy en día en el mundo del cibercrimen se mueve muchísimo dinero y actúa más como una empresa que como criminales aislados [2], como ocurría hace años o como muchas veces siguen caracterizados en películas o series. Tienen equipos enteros de gente dedicados a esta actividad, que cobran un sueldo, personas que se dedican a buscar vulnerabilidades en sistemas y buscar la manera de explotarlas y toda esta información se compra y se vende. Por tanto, ante este auge de la criminalidad en las redes, es más necesario que nunca que los buenos también colaboren entre sí, compartiendo toda la información posible, como potenciales vectores de ataque o programas sospechosos de ser malware, cualquier tipo de información que ayude a mejorar la seguridad a nivel global.

Cabe mencionar también que no solo hablamos de criminales, pues cada vez vemos como más gobiernos de todo el mundo utilizan herramientas para espiar a ciudadanos que considerar peligrosos o para extraer todo tipo de información de distintas organizaciones o empresas.

El presente proyecto plantea una solución para la implantación de una plataforma de respuesta ante incidentes de seguridad. Algo simplificado y a

pequeña escala, pero que puede servir perfectamente como punto de partida o prueba de concepto para que empresas y organizaciones puedan implementar el suyo propio, adaptándolo a sus necesidades, requerimientos, capacidades y recursos disponibles.

1.2. Objetivos del Trabajo

Los principales objetivos para este TFM son los siguientes:

Objetivos de investigación y estudios:

- Revisión bibliográfica sobre la detección de amenazas y gestión de incidentes.
- Estudiar las diversas fuentes que permiten la detección de amenazas.
- Estudiar un sistema de gestión de información, eventos y logs y como se utilizaría desde el punto de vista de seguridad.
- Estudiar posibles herramientas de gestión de incidentes, como se integrarían con las fuentes de datos para enriquecer los casos y como gestionar la respuesta.

Objetivos de implantación y desarrollo:

- Crear y configurar un piloto que sirva para la simulación de un escenario de pruebas y añadir elementos que añadan eventos e incidentes para verificar el funcionamiento de la infraestructura y la integración de todos los elementos.
- Automatización del proceso de instalación y puesta a punto de todos los elementos de la plataforma.
- La solución tiene que ser escalable y reproducible en un entorno real.
- Integración de las herramientas de la solución ante los incidentes simulados.

Objetivos de entrega:

- Generar los entregables parciales de la solución en base a los plazos de entrega establecidos en el TFM.
- Desarrollar la memoria final del TFM.
- Elaborar una presentación y vídeo resumen sobre el TFM.
- Defensa del TFM ante un tribunal.

1.3. Impacto en sostenibilidad, ético-social y de diversidad

Sostenibilidad

Por todos es sabido el impacto energético que tienen los grandes centros de datos que hacen funcionar Internet y las aplicaciones que utilizamos hoy día. La energía que alimenta dichos centros de datos a veces proviene de energías verdes y otras veces, la mayoría, de combustibles fósiles. Es un largo camino el que aún queda por recorrer para hacer más sostenible el uso de la tecnología.

No cabe duda de que el desarrollo de este proyecto tendrá un impacto negativo en este sentido, pues, o bien hará uso de recursos computacionales que están en la nube, o se utilizarán ordenadores y dispositivos personales cuya energía no proviene de fuentes verdes. Esto es algo que, por desgracia, no está en nuestra mano elegir.

Aun así, aunque el desarrollo del proyecto pueda tener un impacto negativo en el ámbito de la sostenibilidad, sí que puede ser positivo el resultado que de él se obtenga. El aumento de la seguridad o la prevención, a la larga, puede reducir el consumo de energía, pues si bien es verdad que aumentar la infraestructura en más sistemas, dedicados éstos a mejorar la seguridad, puede acarrear un mayor consumo energético, los problemas derivados de los ataques informáticos son más dañinos. Los procesos de recuperación ante estos desastres suelen ser tareas costosas que consumen tiempo y recursos.

Comportamiento ético y de responsabilidad social

El impacto de este proyecto es positivo en este sentido. La seguridad es algo costoso y las empresas que no quieran asumir el coste, van a ver que les cuesta mucho más recuperarse de un ataque. Cuando hablamos de coste, no solo nos referimos a software bajo licencia de pago, sino también al personal especializado necesario para hacerlo funcionar y mantener toda la infraestructura.

La idea de este proyecto es utilizar herramientas *opensource* y gratuitas para construir esta plataforma de respuesta ante incidentes de seguridad. La idea es utilizar herramientas accesibles a todo el mundo, sin coste alguno, de forma, que cualquier empresa u organización, independientemente de sus capacidades económicas, pueda tener un mínimo de seguridad con el que enfrentarse a los peligros que se pueda encontrar en la red de redes. Todo esto con la idea en mente de que la seguridad no sea algo solo accesible para los que se la puedan permitir, sino algo asequible a todo el mundo. No olvidemos que estamos todos interconectados y que nuestra seguridad también depende de cuánto seguro es nuestro entorno.

Dimensión de diversidad, género y derechos humanos

En cuanto a la diversidad y el género, creo que este trabajo no tiene aspectos ni positivos ni negativos. La seguridad informática es un tema que no entiende de diversidad o género. Las cosas o son seguras o no lo son,

independientemente de quién sea la persona que esté detrás, ya sea el analista o el jefe. Lo único que hay que hacer es que las personas responsables entiendan lo importante que este tema y que es algo que se debe tomar en serio y no relegarlo a algo de menor importancia porque no da dinero, sino que cuesta dinero, tiempo y personal. Más caro resulta ser víctima de un ataque.

Podríamos decir, que este proyecto si tiene al menos un aspecto positivo en lo que a derechos humanos se refiere. Vemos como muchos gobiernos llevan a cabo acciones de espionaje contra sus ciudadanos o contra organizaciones. Si bien es cierto que la seguridad en este ámbito es muy compleja y conlleva más elementos, una plataforma como esta puede ayudar a prevenir accesos malintencionados o la denegación del acceso a la información.

1.4. Enfoque y método seguido

La metodología escogida para el desarrollo de este proyecto es la siguiente:

- Análisis de las funcionalidades de los diferentes elementos que forman parte de la plataforma.
- Diseño de la solución.
- Implementación e integración de todos los componentes.
- Fase de pruebas de los diferentes casos de uso.
- Completar la documentación y presentación.

Para poder empezar con el desarrollo de la plataforma es necesario saber que componentes la van a formar, así como el diseño y la arquitectura de la propuesta. Como punto de partida, se montará un piloto que tenga las funcionalidades básicas de lo que estamos buscando y que integre de la mejor manera posible todos los elementos que hemos seleccionado. Una vez pasado esta etapa inicial, podremos dar el salto e implementar la solución en un entorno de producción.

A medida que el proyecto avance, la memoria se deberá ir actualizando con las nuevos entregables y adaptándose a los posibles cambios de diseño y otras decisiones que se vayan tomando para poder lograr los objetivos propuestos.

Al final, la idea del proyecto es que no solo cumpla su cometido de gestionar los posibles incidentes de seguridad que puedan ocurrir, sino automatizar la instalación y sobre todo la configuración de la plataforma. De esta manera, se puede escalar más fácilmente y se puede mantener y documentar de una mejor manera. Todo con la idea en mente de replicar en la medida de lo posible un entorno de producción real.

Para lograr los objetivos propuestos podemos hacer uso de la metodología Kanban, ya que tenemos desglosadas las tareas principales en subtareas y

repartidas en el tiempo, por lo que podemos hacer un seguimiento de la ejecución de estas y de cómo va avanzando el desarrollo del proyecto.

1.5. Planificación del Trabajo

	Actividad	Categoría	Inicio	Fin	Días
1	**Planificación**	Objetivo			
1.1	Establecer el problema a resolver	Tarea	28/09/2022	29/09/2022	2
1.2	Definición de objetivos	Tarea	30/10/2022	01/10/2022	2
1.3	Descripción metodología	Tarea	02/10/2022	03/10/2022	2
1.4	Planificación del trabajo	Tarea	04/10/2022	06/10/2022	2
1.5	Estado del arte	Tarea	07/10/2022	08/10/2022	1
1.6	Comportamiento ético y global	Tarea	09/10/2022	10/10/2022	1
1.7	Entrega planificación	Hito	11/10/2022	11/10/2022	1
2	**Análisis y diseño solución**	Objetivo			
2.1	Análisis en profundidad de los componentes de detección y respuesta	Tarea	12/10/2022	20/10/2022	7
2.2	Análisis en profundidad de los componentes de recolección, indexación y almacenamiento de información y eventos	Tarea	21/10/2022	27/10/2022	7
2.3	Análisis en profundidad de gestión de incidentes e intercambio de información	Tarea	28/11/2022	03/11/2022	7
2.4	Elección final de los elementos que compondrán la solución final	Tarea	04/11/2022	05/11/2022	2
2.5	Diseño definitivo de la solución y requisitos	Tarea	06/11/2022	07/11/2022	2
2.6	Entrega documentación análisis	Hito	08/11/2022	08/11/2022	1
3	**Implementación**	Objetivo			
3.1	Instalación de las componentes que formarán parte del entorno de pruebas	Tarea	09/11/2022	12/11/2022	4
3.2	Instalación y configuración elementos de detección y respuesta	Tarea	13/11/2022	19/11/2022	7
3.4	Instalación y configuración de los elementos de recolección, indexación y almacenamiento de información y eventos	Tarea	20/11/2022	27/11/2022	7
3.5	Instalación y configuración de los elementos de gestión de incidentes	Tarea	28/11/2022	04/12/2022	7
3.6	Componentes instalados	Hitos	05/12/2022	05/12/2022	1
4	**Integración**	Objetivo			
4.1	Configuración de la integración entre los diferentes productos	Tarea	06/12/2022	10/12/2022	5
4.2	Automatización de la instalación y la configuración	Tarea	11/12/2022	15/12/2022	5
4.3	Plan de pruebas	Tarea	16/12/2022	18/12/2022	3
4.4	Documentar en la memoria	Hitos	19/12/2022	21/12/2022	3
5	**Testing**	Objetivo			
5.1	Generar eventos y simular amenazas	Tarea	22/12/2022	25/12/2022	4
5.2	Pruebas de gestión de incidentes	Tarea	26/12/2022	27/12/2022	2
5.3	Pruebas de respuestas automatizadas	Tarea	28/12/2022	29/12/2022	2
5.4	Simulación completa	Hito	30/12/2022	31/12/2022	2
6	**Presentación de la Memoria**	Objetivo			
6.1	Revisión final de la memoria	Tarea	04/01/2023	07/01/2023	4

6.2	Elaborar conclusiones y resumen	Tarea	08/01/2023	09/01/2023	2
6.3	Entrega documento memoria	Hito	10/01/2023	10/01/2023	1
7	**Presentación del Vídeo**	Objetivo			
7.1	Elaboración del video	Tarea	11/01/2023	14/01/2023	4
7.2	Revisión final del vídeo	Tarea	15/01/2023	16/01/2023	2
7.3	Presentación y video	Hito	17/01/2023	17/01/2023	1

Tabla 1 - Planificación temporal del proyecto

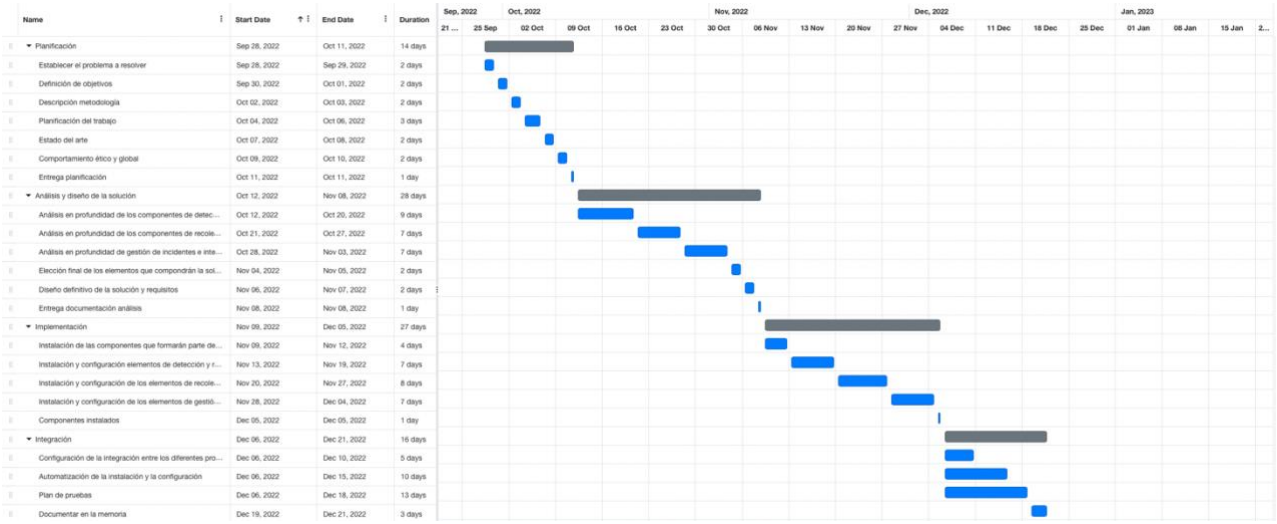

Ilustración 1 - Diagrama de Gantt (1)

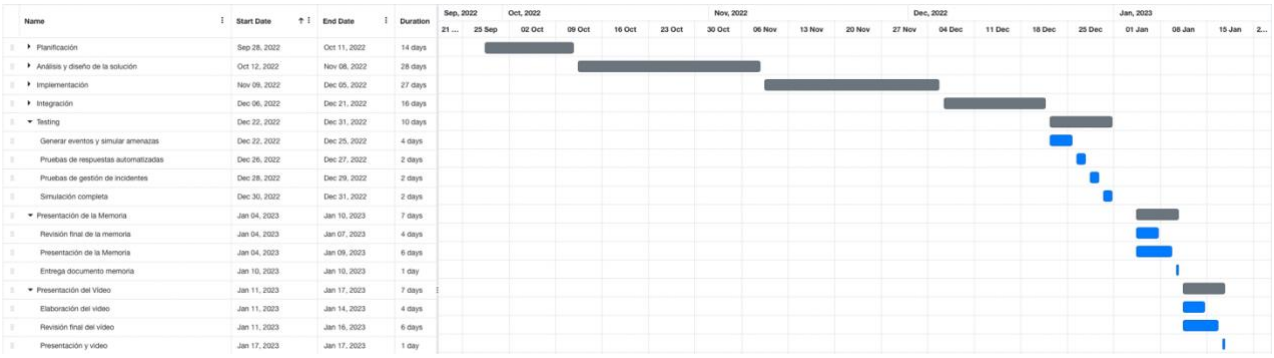

Ilustración 2 - Diagrama de Gantt (2)

1.6. Breve sumario de productos obtenidos

El desarrollo de una plataforma de estas características requiere del uso de varios componentes, no de uno solo. Es posible que podamos encontrar alguna solución comercial que disponga de todas o casi todas las funcionalidades que necesitamos, pero suelen ser opciones muy costosas y, además, no forma parte de la naturaleza de este proyecto, que tiene por objetivo utilizar herramientas *opensource*.

Para poder utilizar una plataforma de estas características necesitamos elementos que se encarguen de la detección de las posibles amenazas, de almacenar y ordenar toda la información que se irá generando, de relacionar y analizar dicha información en busca de patrones y de poder generar alertas en base a los resultados de dichos análisis. Además, necesitamos un lugar donde poder visualizar de una manera intuitiva toda esta información, de forma que toda la información pueda ser compartida con todo el personal involucrado. Por último, necesitamos poder gestionar los casos que vayan surgiendo, es decir, documentar los incidentes de seguridad ocurridos y tratarlos como sea oportuno.

A priori, se han escogido una serie de productos, todos ellos *opensource*, que cumplen con nuestras necesidades.

- **Detección y respuesta**

Wazuh [3]: es un sistema de detección de intrusos. Lo forman un agente que se instala en las máquinas que se quieren monitorizar y de un servidor que centraliza la información obtenida por los diferentes agentes.

Con Wazuh cubrimos las máquinas de la infraestructura, pero no la red, que es otro elemento importante. Para la protección de las comunicaciones tenemos los **IPS** o **IDS** [4], como **SNORT** [5] y **Suricata** [6], que actúan monitorizando el tráfico de red y que tienen la capacidad de bloquear ciertos tipos de ataques.

Un IDS o sistema de detección de intrusos, es utilizado para detectar accesos no autorizados a una máquina o a una red. Es decir, son sistemas que analizan el tráfico entrante y ante cualquier actividad sospechosa, envían una alerta. SNORT es un ejemplo de un IDS.

Por otro lado, un IPS o sistema de prevención de intrusos, es un sistema que se utiliza para proteger sistemas de ataques e intrusiones. Su actuación es preventiva. Estos sistemas analizan en tiempo real las conexiones y protocolos para determinar si se está produciendo o se va a producir un incidente, identificando los ataques según patrones, anomalías o comportamientos sospechosos y bloqueando los potenciales ataques. Es decir, un IPS además de lanzar alarmas, puede descartar paquetes y desconectar conexiones. Suricata actúa tanto como un IDS que como un IPS.

A todo lo anterior tendríamos que sumarle el cortafuegos o firewall, un elemento básico de seguridad que forma la primera línea de defensa de cualquier empresa u organización. A nivel profesional suele ser un dispositivo hardware dedicado, pero en muchos otros casos suele ser un añadido al router, como ocurre a nivel doméstico o en pequeñas empresas.

Puede aportar algunas capacidades en cuanto a detección a nivel de red y formar parte del sistema de respuesta en determinados tipos de incidentes, pues tiene la potestad de bloquear conexiones entrantes y/o salientes.

Si vamos por la vía de las herramientas opensource, encontramos al clásico software de **iptables** [7], disponible en la mayoría de las distribuciones GNU/Linux, que, aunque es un software antiguo (su primera versión salió en 1998), sigue cumpliendo su función a la perfección y sigue siendo una opción muy válida. Por otro lado, tenemos **nftables** [8], un software que vino a sustituir a iptables y que actualmente lo podemos encontrar funcionando junto a **firewalld** [9], el firewall por defecto en los sistemas Linux modernos.

Por otro lado, y siguiendo en el terreno de los firewalls, encontramos a **Pfense** [10], basado en FreeBSD, es un firewall de código abierto que viene preparado para funcionar tanto como firewall que como router y que nos permite la instalación de componentes extras de seguridad como SNORT o Suricata u otros elementos que no están centrados en la seguridad.

Otro componente de detección y respuesta es un **WAF** [11] (Web Aplication Firewall), un firewall que funciona a nivel de capa de aplicación. Monitoriza, filtra o bloquea el tráfico HTTP hacia o desde una aplicación web. Su función es proteger las aplicaciones de diversos ataques como cross-site-scriptinng (XSS), ataques de inyección SQL, etc. [12].

- **Recolección, indexación y almacenamiento de la información**

Para la recolección de datos tenemos la opción de optar por el **agente de Wazuh**, el cual podrá recolectar información procedente de diferentes fuentes, como ficheros de logs o eventos de Windows, de esta manera recolectará información acerca de intrusiones y comportamientos extraños y la enviará al servidor de Wazuh para que sea procesada. Toda la información es enviada mediante un canal cifrado entre el agente y el servidor.

Por otro lado, tenemos **Logstash** [13], que podemos utilizar junto con el agente de Wazuh. Logstash forma parte del stack ELK y su función es procesar la información que le llegan de los diferentes "*beats*" [14], para luego ser enviada al servidor de ElasticSearch para su almacenamiento. Los beats, como Filebat, Packetbeat o Winlogbeat, son agentes que se instalan en los endpoints, y que, al igual que el Wazuh agent, recolectan diversa información en base a determinadas reglas.

Para el almacenamiento e indexación de la información tenemos la opción de **ElasticSearch** [15], que junto a Kibana y Logstash, conforman el stack completo de ELK. ElasticSearch es una base de datos que sirve para almacenar,

indexar y realizar búsquedas. La idea es que la información que es recogida por los agentes y enviada al servidor de Wazuh, sea luego redirigida al servidor de ElasticSearch para su almacenamiento e indexado. Toda esta información se podría luego visualizar en **Kibana** [16], que cuenta con un plugin dedicado para Wazuh.

Junto a estos elementos, tendríamos que sumar **ElastAlert** [17], un sistema que genera alertas sobre la base de la información almacenada en ElasticSearch siguiendo unos patrones y reglas establecidos.

Por otro lado, tenemos **Wazuh Indexer** [18] y **Wazuh Dashboard** [19]. A partir de la **versión 4.3** [20], Wazuh ha añadido dos nuevos componentes, Wazuh indexer y Wazuh dashboard, que vendrían a sustituir a ElasticSearch y Kibana. Wazuh Indexer está basado en **Opensearch** [21], un fork de ElasticSearch y Kibana que nació a inicios de 2021, y, junto con herramientas desarrolladas por el propio equipo de Wazuh, pasa a convertirse en su motor de búsqueda principal. Y Wazuh Dasboard se convertirá en la interfaz web de la plataforma Wazuh.

- **Gestión de incidentes y compartición de la información**

TheHive [22] es una plataforma de código abierto de respuesta a incidentes de seguridad escalable, diseñada para la gestión de incidentes, desde que son abiertos, hasta que se dan por cerrados. Puede recibir alertas procedentes de diferentes fuentes de información SIEM, IDS/IPS, firewalls, etc. A partir de estas alertas, TheHive creará un caso para la nueva incidencia para que sea investigada por los analistas.

Por parte del mismo equipo responsable de TheHive tenemos **Cortex** [23], un motor de análisis de código abierto que busca complementar a TheHive para obtener información adicional analizando los observables, como direcciones IP, correos electrónicos, URLs, etc. utilizando analizadores, como puede ser VirusTotal, aunque también admite la posibilidad de crear tu propio analizador.

Junto a TheHive y Cortex encontramos a **MISP** [24]. Es una plataforma de código abierto de intercambio de amenazas. Lo forman una gran comunidad de usuarios que comparten información sobre amenazas o indicadores de seguridad de todo el mundo. Permite suscribirse a fuentes de inteligencia de amenazas, lo cual puede ayudar a prevenir un incidente creando reglas de detección o bloqueo en base a patrones para Snort o Suricata, por ejemplo. Cortex también tiene la capacidad de buscar observables dentro de MISP.

2. Análisis y diseño de la solución

2.1. Análisis de los componentes de detección y respuesta

Como primer paso para nuestra plataforma de respuesta a incidentes de seguridad necesitamos elegir herramientas capaces de realizar una detección eficaz ante los peligros a los que se enfrentan los sistemas de una organización.

Para llevar a cabo una detección eficaz es necesario disponer de herramientas que puedan trabajar en los equipos clientes y servidores, para poder detectar usos de los sistemas que vayan en contra de las políticas de seguridad de la organización. Es necesario disponer de herramientas que puedan, no solo detectar el malware, sino también bloquear su actividad y si es posible, eliminarlos. Este tipo de herramientas se conocen por **Endpoint Detection Response** [25].

Un sistema EDR es un sistema de protección de los equipos, una solución de seguridad diseñada para detectar y bloquear amenazas a nivel de dispositivo. Aúna varias funcionalidades de seguridad, combina una antivirus tradicional junto con herramientas de monitorización e inteligencia artificial, para ofrecer una respuesta rápida y eficiente.

Otro elemento del ámbito de la detección y respuesta que nos puede ayudar es un IDS o IPS, como ya vimos, pero a nivel de red. Wazuh se puede considerar también un IPS, pero a nivel de host (HIDS), es decir, se instala en una máquina y monitorizando su actividad y es todo lo que abarca. La red es uno de los elementos más importantes de cualquier empresa u organización y, por tanto, se debe monitorizar su actividad de la misma forma, para detectar posibles amenazas o comportamientos extraños.

Como ya vimos en apartados anteriores, un IDS se encarga de la detección y alerta en caso de detectar alguna amenaza, mientras que, al segundo, un IPS, se le añade la capacidad de bloquear amenazas. Para el caso de nuestra plataforma de respuesta, ambos se complementan a la perfección, los EDR monitorizan los hosts y pueden detectar comportamientos que a nivel de red no se ven y los IDS/IPS monitorizan la red en su conjunto, donde los EDR no llegan.

Además, puede darse el caso de tener equipos a los cuáles no se les pueda instalar nada fuera de lo que permite el fabricante, por lo que estaríamos completamente ciegos por esa parte. La red, por otro lado, es algo que controlamos y gestionamos nosotros y que podemos monitorizar sin estar cerrados a uno u otro fabricante.

Para el caso de la herramienta de detección en hosts y dispositivos tenemos los agentes de Wazuh y como herramientas para la monitorización de la red, podemos optar por Snort o Suricata.

2.1.1. Wazuh

Wazuh es una herramienta de código abierto que nos permite monitorizar equipos para detectar amenazas, realizar comprobación de integridad del sistema de ficheros, realizar análisis de logs o realizar escáneres de vulnerabilidades en equipos, entre otras cosas. Proporciona detección de intrusiones para la mayoría de los sistemas operativos modernos, como Linux, macOS, Windows, Solaris, HP-UX o AIX (sistema operativo de IBM). Es una plataforma que aúna varias funcionalidades en un único lugar, evitando que debamos tener varias herramientas diferentes y tener que integrarlas entre sí para que todo funcione.

Lleva funcionando desde 2015 y nació como un fork de otro proyecto de seguridad, **OSSEC** [26]. Aunque OSSEC también es un software de código abierto, tiene varias modalidades de uso, desde la gratuita y limitada, pasando por una versión intermedia gratuita, pero a la que se accede previo registro y una última versión de pago [27]. La versión OSSEC+ es gratuita e incluye el soporte para el stack ELK, pero se acceder previo registro. Por otro lado, Wazuh ofrece el producto completo sin necesidad de registro previo. Además, ofrece más elementos extras que OSSEC, como una API REST completa, por ejemplo.

Wazuh es un producto que cuenta con una comunidad muy grande y activa detrás que le brinda un gran soporte al producto. Es un producto suficientemente maduro como para que sea viable implementarlo en una organización, en un entorno de producción real. Actualmente, Wazuh va por su versión 4.3.9, por lo que se publican de manera regular nuevas versiones enfocadas a solucionar bugs y añadir nuevas funcionalidades.

Además de la comunidad, Wazuh tiene detrás una empresa, Wazuh Inc., la cual es la encargada de desarrollar y mantener el producto. Esto también es un punto a favor de Wazuh, pues el que haya una empresa detrás hace que muchas empresa y organizaciones se puedan tomar más en serio este producto para implementarlo internamente, ya sea en su formato gratuito o de pago. Wazuh dispone de dos versiones de pago en las que brinda soporte a los clientes:

- Versión **estándar,** con soporte 8/5 (horario de oficina) y con un tiempo máximo de respuesta de 8 horas.

- Versión **premium**, con soporte 24/7 y con un tiempo máximo de respuesta de 4 horas.

Además de estas opciones, Wazuh se ofrece como SaaS (*software as a service*) en su propia infraestructura cloud. También dispone de servicios de formación previo pago sobre la plataforma.

Wazuh tiene una arquitectura centralizada y multiplataforma que permite que múltiples sistemas se puedan monitorizar sin problema. Es un producto escalable y, por tanto, puede implementarse en una organización con independencia de su tamaño. Tiene una estructura cliente/servidor, siendo los

clientes los equipos y sistemas a proteger, donde se instalaría el agente de Wazuh, y uno o varios servidores en donde se recoge y procesa la información remitida por los agentes.

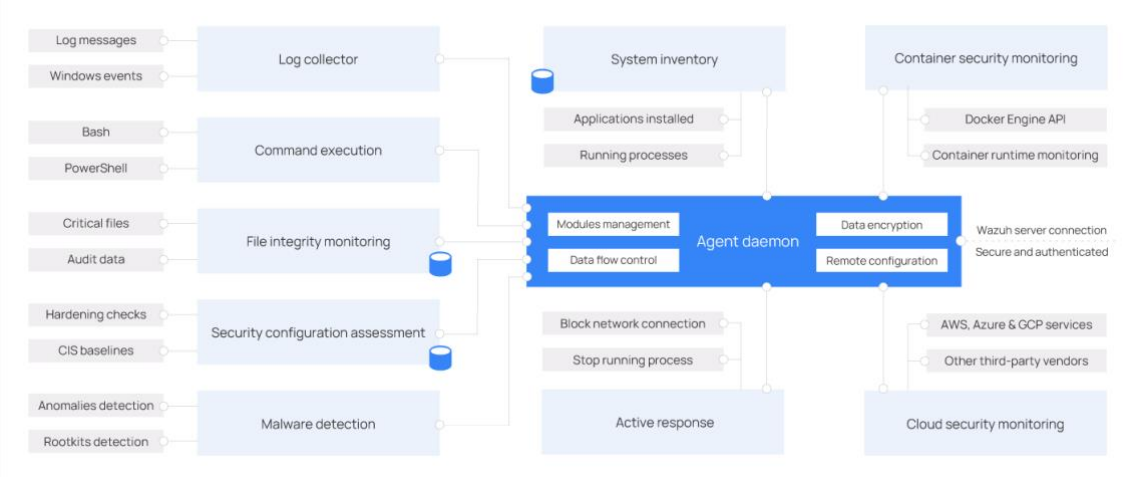

Ilustración 3 - Capacidades del agente Wazuh

Como podemos observar, el agente de Wazuh [28] tiene una arquitectura modular, donde cada componente se encargar de realizar una tarea determinada, como puede ser la monitorización del sistema de ficheros, la recolección de los logs del sistema, el escaneo del sistema en busca de vulnerabilidades o detección de malware. Todo es adaptable según las necesidades y requerimientos del usuario y la organización. El agente envía al servidor no solo toda la información que recolecta, sino también información acerca de su operatividad, configuración y estado, de forma que podemos gestionar todos los agentes y su configuración desde el propio servidor. La comunicación entre el agente y el servidor está cifrada.

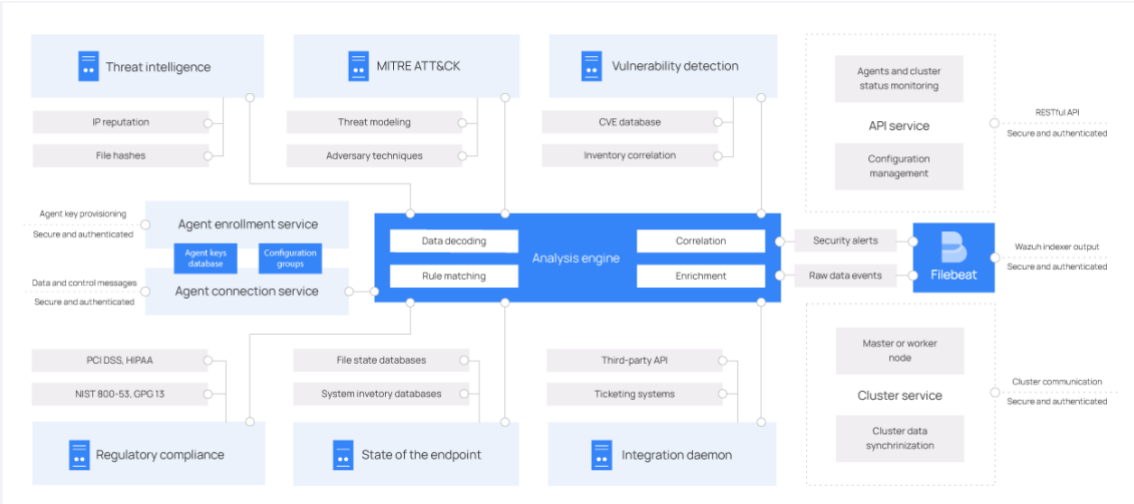

Ilustración 4 - Capacidades del servidor de Wazuh

El servidor de Wazuh se compone de varios elementos [29], cada uno encargado de una función en concreto. Por un lado, encontramos componentes encargados de la gestión de los agentes y la comunicación con éstos. También

encontramos el motor de análisis, que es el encargado de analizar los datos recolectados por los agentes, siendo capaz de identificar la clase de información que está procesando, si son eventos de Windows, registros de conexiones SSH, etc. También encontramos una API REST, que se utiliza para interactuar con el resto de los elementos de la infraestructura. Filebeat es el encargado de enviar los eventos y alertas emitidos por el motor de análisis hacia Wazuh Indexer.

Para el tema de la escalabilidad, tenemos el Wazuh Cluster Daemon, que es el servicio encargado de escalar horizontalmente Wazuh. Es el servicio que utilizarían los n servidores Wazuh para comunicarse entre sí. Si necesitásemos escalar Wazuh, además de este servicio necesitaríamos implementar un balanceador de carga para que reparta las peticiones entre los diferentes servidores y que, además, proveyese de alta disponibilidad. Eso sí, eso sería un servicio externo a Wazuh. Una de las opciones que podríamos utilizar podría ser **HAProxy** [30].

Wazuh, como ya hemos dicho, es multiplataforma y dispone de múltiples opciones de instalación, podemos hacerlo de manera manual, instalando todos los paquetes y dependencias, de forma automatizada (*infraestructura as a code*) o a través de contenedores Docker o utilizando Kubernetes. Dispone de toda la documentación necesaria para realizar una instalación automática [31]. Como alternativa, también disponemos de una OVA, una máquina virtual donde están todos los elementos ya instalados y configurados, listos para ser utilizados [32], aunque no sería la mejor opción para implementarlo en una organización, sino más bien para probar el producto y hacer alguna prueba pequeña sin querer complicarse con la instalación y configuración de cada uno de los componentes.

2.1.2. Snort

Snort es un sistema de detección de intrusos (IDS) basado en red, de código abierto, desarrollado en C y cuya primera versión salió en 1998. Actualmente pertenece a Cisco quién se encarga de su desarrollo y mantenimiento. Es un software multiplataforma disponible tanto para Linux, como para Windows, como para FreeBSD.

Mediante una serie de reglas que definen la actividad de red maliciosa, se identifican paquetes maliciosos y se envían alertas.

Snort implementa un lenguaje de creación de reglas flexible, potente y sencillo, permitiendo que Snort pueda llegar a detectar desde ataques de denegación de servicio (DoS), ataques Common Gateway Interface (CGI) hasta escaneos de puertos con Nmap. Además de permitirnos crear nuestras propias reglas, Snort ya viene con un conjunto de reglas oficiales, pero podemos, además, utilizar conjuntos de reglas ya definidas y probadas por la comunidad.

Snort tiene tres modos de funcionamiento:

- **Modo Sniffer**, donde muestra en tiempo real los paquetes que circulan por la red. La visualización de dichos paquestes es configurable, pudiendo mostrar diferentes tipos de paquetes, TCP, UDP, ICMP, etc.

- **Modo Packet Logger**, donde registrará y guardará todos los paquetes que circulan por la red detectados en el modo Sniffer.

- **Modo NIDS**, en este modo el programa monitorizará y analizará el tráfico en base al conjunto de reglas que tiene definido. Si algún paquete coincide con alguna regla, enviará una alerta.

A partir de la versión 3, Snort soporta multihilo y multinúcleo.

2.1.3. Suricata

Suricata es un sistema de detección de intrusos (IDS) y también un sistema de prevención de intrusos (IPS). Es de código abierto, desarrollado en C y Rust, cuya primera versión salió en 2010. Esta desarrollado y mantenido por la Open Information Security Foundation, una organización sin ánimo de lucro, dedicada a fomentar la comunidad y el soporte de tecnologías de seguridad [33].

Al igual que Snort, trabaja con un conjunto de reglas, mediante las cuáles analiza el tráfico de red para detectar comportamientos maliciosos.

Tiene dos modos de funcionamiento:

- **Modo IDS** o modo pasivo: monitoriza y analiza la red en busca de actividades maliciosas, como intrusiones, para enviar una alerta al respecto.

- **Modo IPS** o activo: en este modo además puede prevenir intrusiones que pueden poner en riesgos el sistema. Es decir, no solo envía una alerta al detectar una posible intrusión, sino que además la bloquea.

Suricata es una solución escalable y multihilo de forma que solo con ejecutar una instancia, el programa será capaz de balancear su carga entre todos los procesadores disponibles. También lo podemos configurar para que no utilice todos los procesadores disponibles si así lo deseamos.

2.2. Análisis de componentes de recolección y almacenamiento de eventos

2.2.1. Wazuh Indexer y Wazuh Dasboard

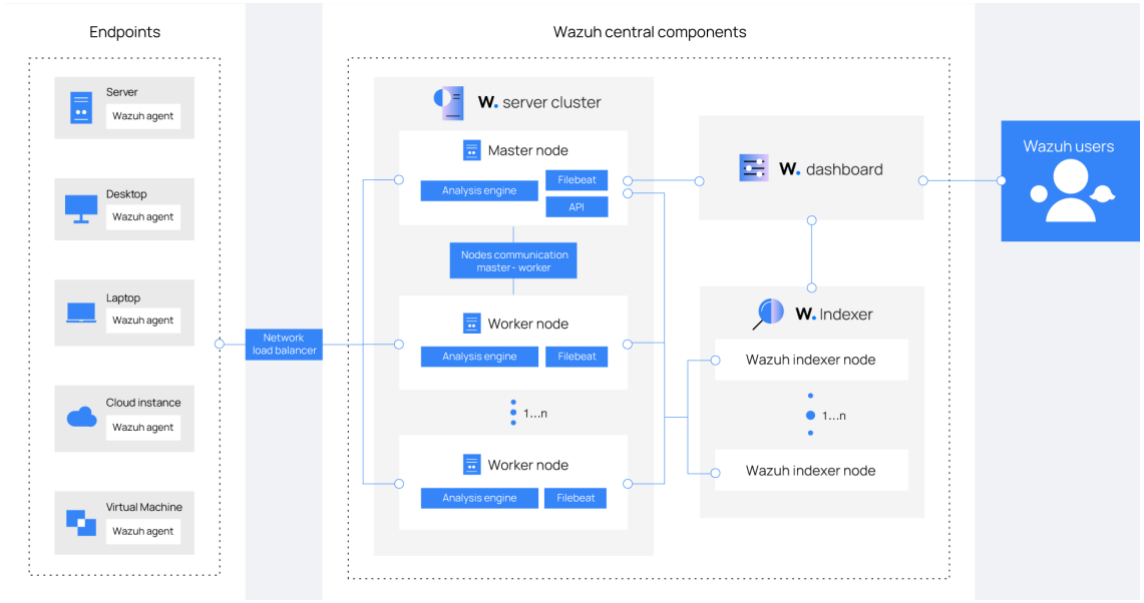

Ilustración 5 - Despliegue de Wazuh

Como podemos ver, esta sería la estructura al completo de Wazuh, los agentes instalados en cada uno de los endpoints, ya sean ordenadores, servidores, máquinas virtuales o instancias en la nube (también es capaz de monitorizar contenedores, aunque no aparezca en la imagen), recolectan información y la envían al servidor de Wazuh, que pudiendo ser uno solo o ser un cluster con n servidores. La información es analizada en el servidor y es enviada utilizando Filebeat a otro de los componentes clave de la infraestructura, Wazuh Indexer, para que sea almacenada e indexada. Este componente también puede ser un único servidor o un cluster de servidores. A su vez, toda la información se puede visualizar en el último componente, Wazuh Dashboard.

Como se comentó anteriormente, a partir de la versión 4.3 Wazuh deja de utilizar el stack de ELK, ElasticSearch, Logstash y Kibana, para pasar a utilizar herramientas propias que sustituyen a las anteriores, Wazuh Indexer y Wazuh Dasboard. Ambos elementos están basados en OpenSearch, un fork de ElasticSearch. Con esto han conseguido no depender de una herramienta externa como era ELK, a herramientas propias sobre las cuales tiene control absoluto, consiguiendo una mayor y mejor integración entre las tres. Recordemos que Wazuh disponía de plantillas para integrar con logstash y de un plugin dedicado para trabajar con Kibana. Ahora la instalación, configuración y mantenimiento de toda la infraestructura es mucho más fácil y sencilla.

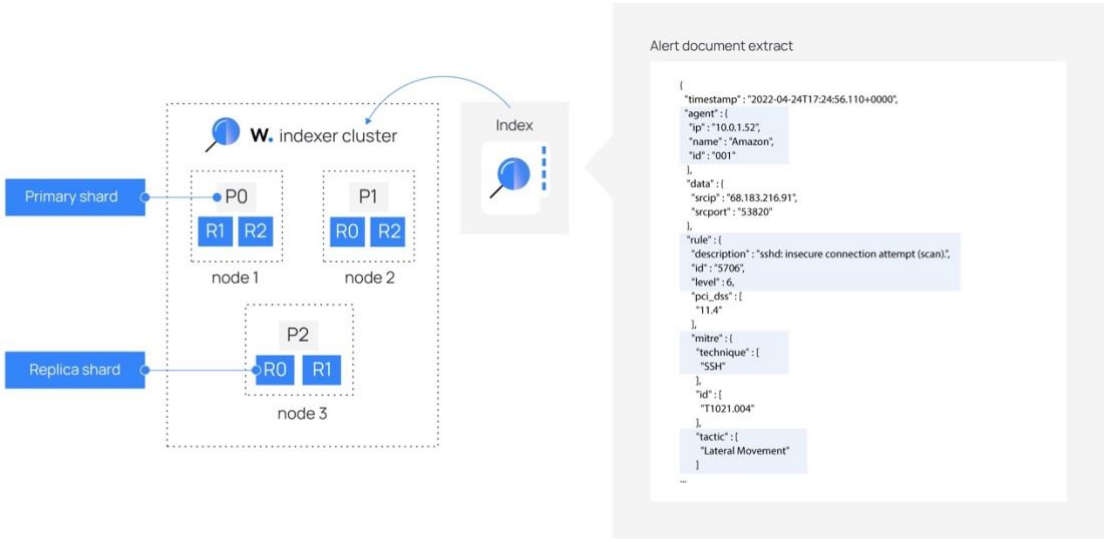

Ilustración 6 - Wazuh Indexer cluster and alert document extract

Wazuh Indexer es similar a ElasticSearch y emula el papel que tenía éste en las versiones anteriores de Wazuh. Wazuh Indexer es un motor de búsqueda y análisis, altamente escalable, que almacenará e indexará las alertas generadas por el servidor de Wazuh. Además, es capaz de realizar búsquedas y análisis en tiempo real sobre dichos datos.

Al igual que se puede hacer con ElasticSearch, se puede configurar como un único nodo o se puede configurar como un cluster multinodo de n nodos, por lo que es escalable y permite alta disponibilidad.

El servidor de Wazuh Indexer almacena la información en ficheros en formato JSON. Funciona como Elastic al almacenar los documentos de forma distribuida a lo largo de varios contenedores denominados “shards”. Al distribuir los documentos entre varios shards y a su vez entre varios nodos, se consigue que haya redundancia en los datos y así, mantener la integridad de estos. Esto permite que los datos no se vean afectados en caso de un fallo a nivel de hardware o de software. Obviamente, esto aumenta según se añaden nuevos nodos al cluster.

Para el caso de ElasticSearch, Filebeat es un agente que, instalado en los endpoints, recolecta la información necesaria y la envía a Logstash, para ser procesada y enviada a ElasticSearch. Con esta nueva arquitectura, vemos que Filebeat para a ser un componente más del servidor de Wazuh y que el propio agente de Wazuh es capaz de realizar, entre otras muchas cosas, la recolección de logs en la máquina donde está instalado. Por tanto, también prescindimos de Filebeat y de Logstash, para tener todo más centrado en el agente, el servidor y el indexador de Wazuh.

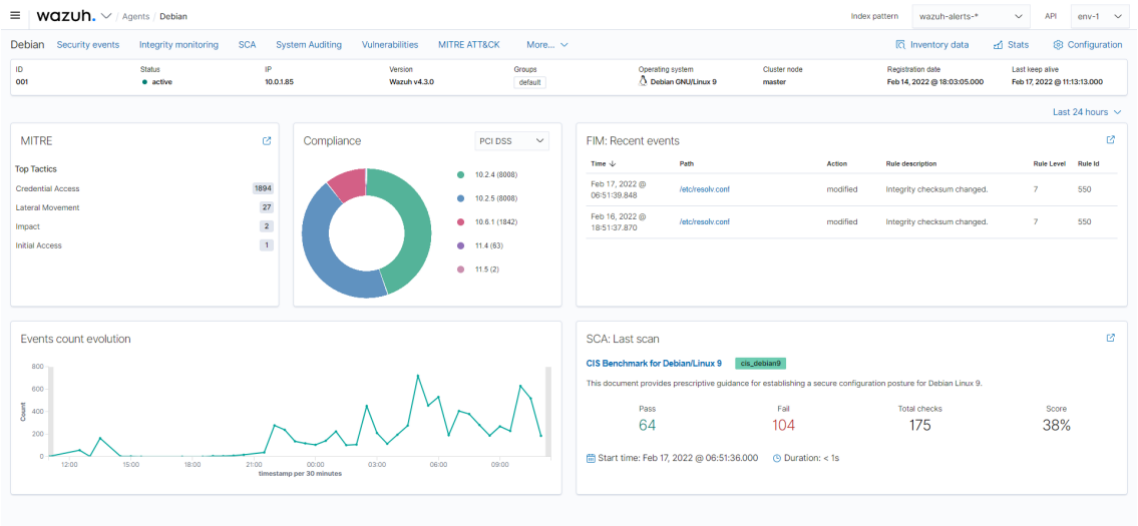

Ilustración 7 - Wazuh Dashboard

El último elemento que nos falta es Wazuh Dashboard es la herramienta de visualización de Wazuh. En versiones anteriores teníamos a Kibana para visualizar la información, alertas y realizar búsquedas sobre los datos y, por otro lado, el servidor de Wazuh también disponía de una interfaz para gestionar el propio servidor y los agentes. Ahora no, ahora todo se ha centralizado en Wazuh Dashboard, una interfaz web para analizar y visualizar los eventos y las alertas de seguridad, así como gestionar y monitorizar la plataforma entera de Wazuh. Además, dispone de un control de acceso basado en roles (RBAC), así como de autenticación single-sing-on.

En resumen, para poner en funcionamiento nuestra plataforma de detección de intrusos y amenazas hemos pasado de tener diferentes elementos de diferentes fuentes, como eran los agentes y el servidor de Wazuh por un lado, y el stack de ELK, ElasticSearch, Logstash y Kibana, por otro lado, a tener solo cuatro, los agentes, el servidor, el indexador y la web de visualización, todos ellos, de Wazuh, sin necesidad de herramientas de terceros.

2.3. Análisis de componentes de gestión de incidentes

2.3.1. TheHive

TheHive es una herramienta de código abierto que permite gestionar incidentes de seguridad, mediante la utilización de casos, a través de los cuáles se investigan los incidentes y se dan por finalizados. Está diseñada para ser utilizada por SOCs, CSIRTs, CERTs y cualquier otro miembro de la comunidad de seguridad que deba lidiar y gestionar incidentes de seguridad.

Para el almacenamiento e indexación de la información, TheHive soporta diferentes formas de almacenamiento, aunque recomiendan utilizar para un entorno de producción la base de datos de tipo NoSQL Apache Cassandra, por ser escalable y tolerante a fallos.

La herramienta permite sincronizarse con una o varias instancias de MISP para empezar investigaciones a partir de eventos emitidos por MISP. Además, TheHive es también utilizado junto a Cortex para analizar observables, como direcciones IP, correos electrónicos, etc. y aportar más información.

Tal y como se resume en su página oficial, estas son las características principales de TheHive [34]:

- **Gestión de alertas**: a través de una detallada página, se pueden realizar comentarios, identificar alertas similares, definir campos y estados personalizados, etc. Permite decidir si la alerta se clasifica como una investigación o como una respuesta ante un incidente.

- **Gestión de casos**: permite crear casos y asociar tareas y observables. Permite, además, identificar casos similares y alertas, definir un nivel PAP o *Permissible Actions Protocol* para cada observable, y permite la utilización de plantillas para mejorar el proceso de respuesta de incidentes.

- **Definición de entornos**: permite definir diferentes organizaciones y equipos y que puedan o no trabajar de manera colaborativa entre ellos.

- **Gestión avanzada de usuarios**: permite una gestión y personalización de perfiles y asignarlos a usuarios, además de permitir mapeo de usuarios con LDAP o *Active Directory*.

- **Framework de notificaciones**: permite definir reglas de notificación y enviarlas a través de diferentes canales, como email, Slack, etc.

- **Métricas y dashboard**: permite la visualización de diferentes estadísticas como casos, tareas, observables, etc.

- **API**: acceso completo y documentación a una API que permite implementar flujos de trabajo y diferentes automatizaciones.

- **Integración con MISP**: permite compartir indicadores de compromiso desde MISP.

TheHive se puede instalar en sistemas Linux, en distribuciones Debian/Ubuntu o RedHat/Centos/Fedora, pero también permite la instalación vía contenedores de Docker.

La última versión disponible es la versión 4.1.24, aunque la versión 5 [34] acaba de salir y también está disponible para su uso.

TheHive es una herramienta escalable, lo que permite su instalación en un único servidor o en varios nodos en un cluster.

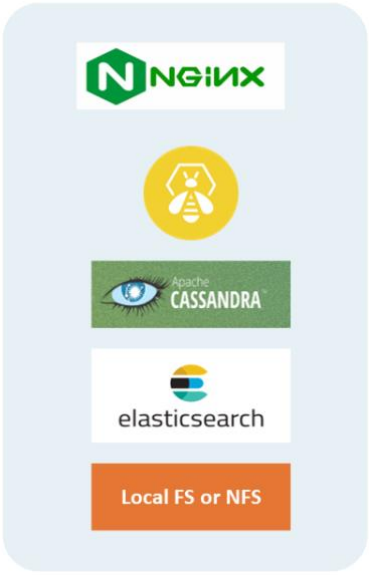

Ilustración 8 - TheHive Standalone server

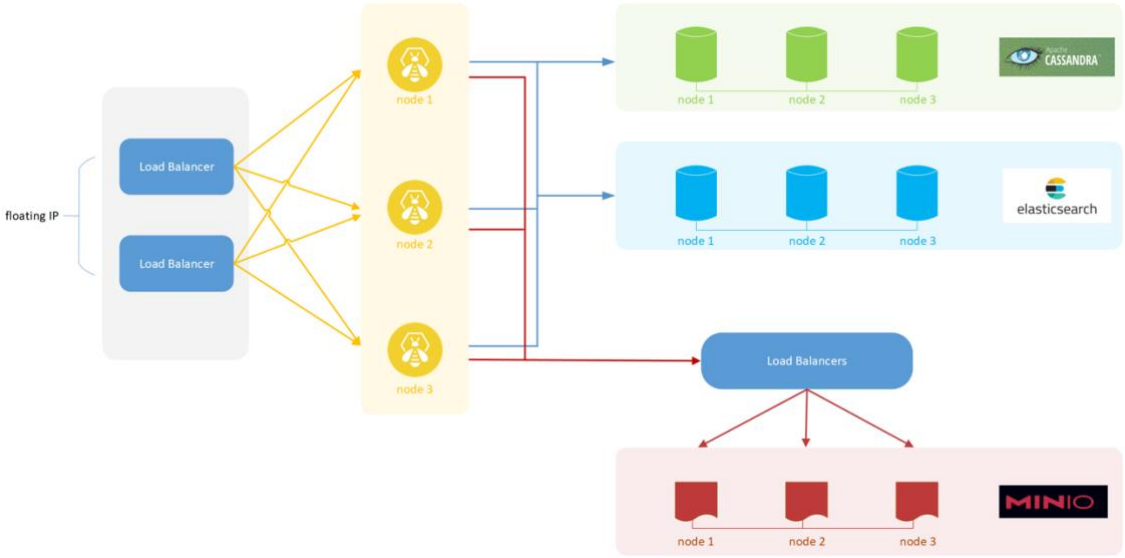

Ilustración 9 - Thehive Cluster architecture

Para su funcionamiento, TheHive requiere de:

- **Una base de datos**. Recomiendan utilizar Apache Cassandra, que es escalable y permite alta disponibilidad.

- **Un sistema de almacenamiento de ficheros**, como puede ser una carpeta local o compartida por NFS, un sistema como Apache Hadoop u otro sistema de ficheros distribuido.

- **Un sistema o motor de indexado**. Si optamos por una instalación en un único servidor, es más que suficiente utilizar el sistema de indexado que trae, pero si queremos un cluster, entonces necesitaremos hacer uso de ElasticSearch.

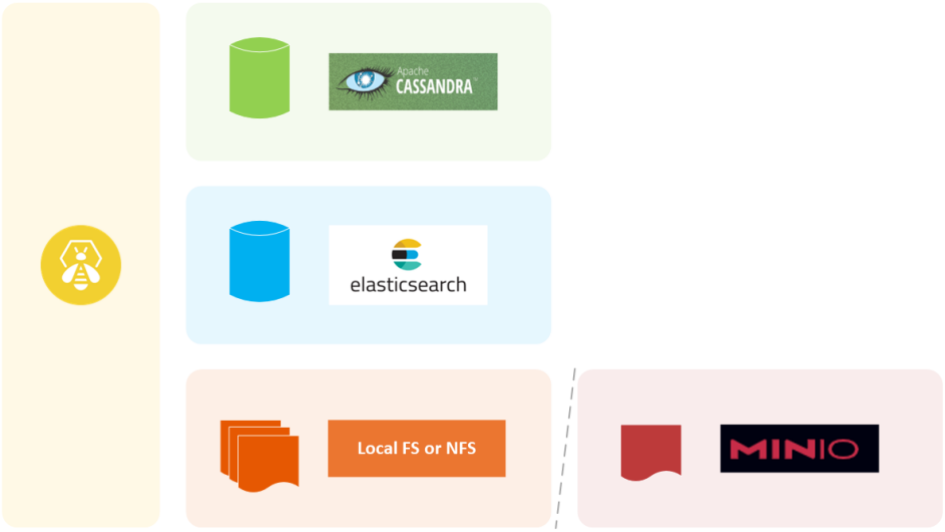

Ilustración 10 - TheHive Stack

2.3.2. Cortex

Cortex es un motor de análisis de observables desarrollado por el mismo equipo que de TheHive y pensado para que sea utilizado junto a TheHive. Observables son datos como direcciones IP, direcciones de correo electrónico, URLs, nombres de dominio, ficheros, hashes, etc. que pueden ser analizados uno a uno o todos junto utilizando la interfaz web que provee Cortex. El análisis también se puede automatizar gracias a la API REST que pone a disposición Cortex.

Los observables pueden ser analizados por Cortex utilizando analizadores, como VirusTotal. Antes de usar cualquier analizador es conveniente leer los requerimientos porque, entre otras cosas, puede que alguno de ellos sea de pago.

Podemos escribir nuestros propios analizadores, pero tiene que ser en un lenguaje que esté soportado por un sistema Linux, como Python.

Cortex está escrito en Scala y utiliza AngularJS para su interfaz web. Se puede instalar en distribuciones de Linux Debian/Ubuntu y RedHat/Centos/Fedora, pero también se puede instalar en contenedores Docker.

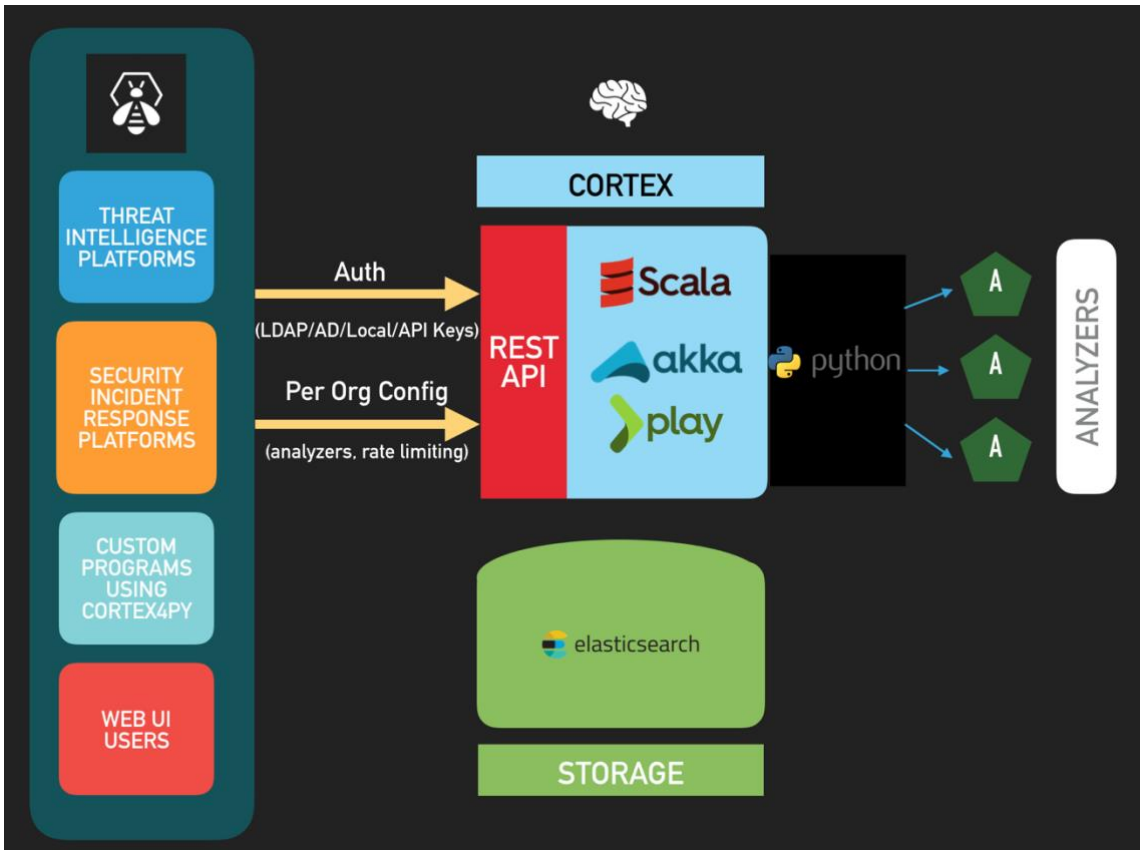

Ilustración 11 - Cortex Architecture

Gracias a su API REST *stateless*, Cortex es escalable horizontalmente y TheHive puede conectarse a una o varias instancias de Cortex.

A partir de su versión 2, Cortex permite gestionar múltiples organizaciones, usuarios y asignar roles a los usuarios. Además, puedes asignar cuotas de consumo a determinadas organizaciones para que no monopolicen os recursos disponibles, dejando al resto sin poder utilizar la herramienta.

2.3.3. MISP

MIPS (Malware Information Sharing Platform) es una plataforma de código abierto de inteligencia contra amenazas, permite analizar, gestionar y compartir información relacionada con muestras de malware. Está pensada para compartir información entre diferentes organizaciones del ámbito de la seguridad.

A través de MISP, equipos de seguridad pueden compartir información sobre diferentes muestras de malware con el fin agregar esa información a motores de antivirus, por ejemplo.

Para la instalación, MIPS requiere de una distribución Linux, pero recomiendan que se use la distribución de Ubuntu, concretamente, Ubuntu 20.04. También tenemos otros medios de instalación, entre ellos una imagen para VirtualBox o VMWare. También podemos instalar MISP utilizando contenedores Docker y también tenemos a nuestra disposición las instrucciones para llevar a cabo una instalación automatizada con Puppet o Ansible.

Al ser de código abierto, tenemos a nuestra disposición todo el código en su repositorio de GitHub por si queremos instalarlo desde las fuentes.

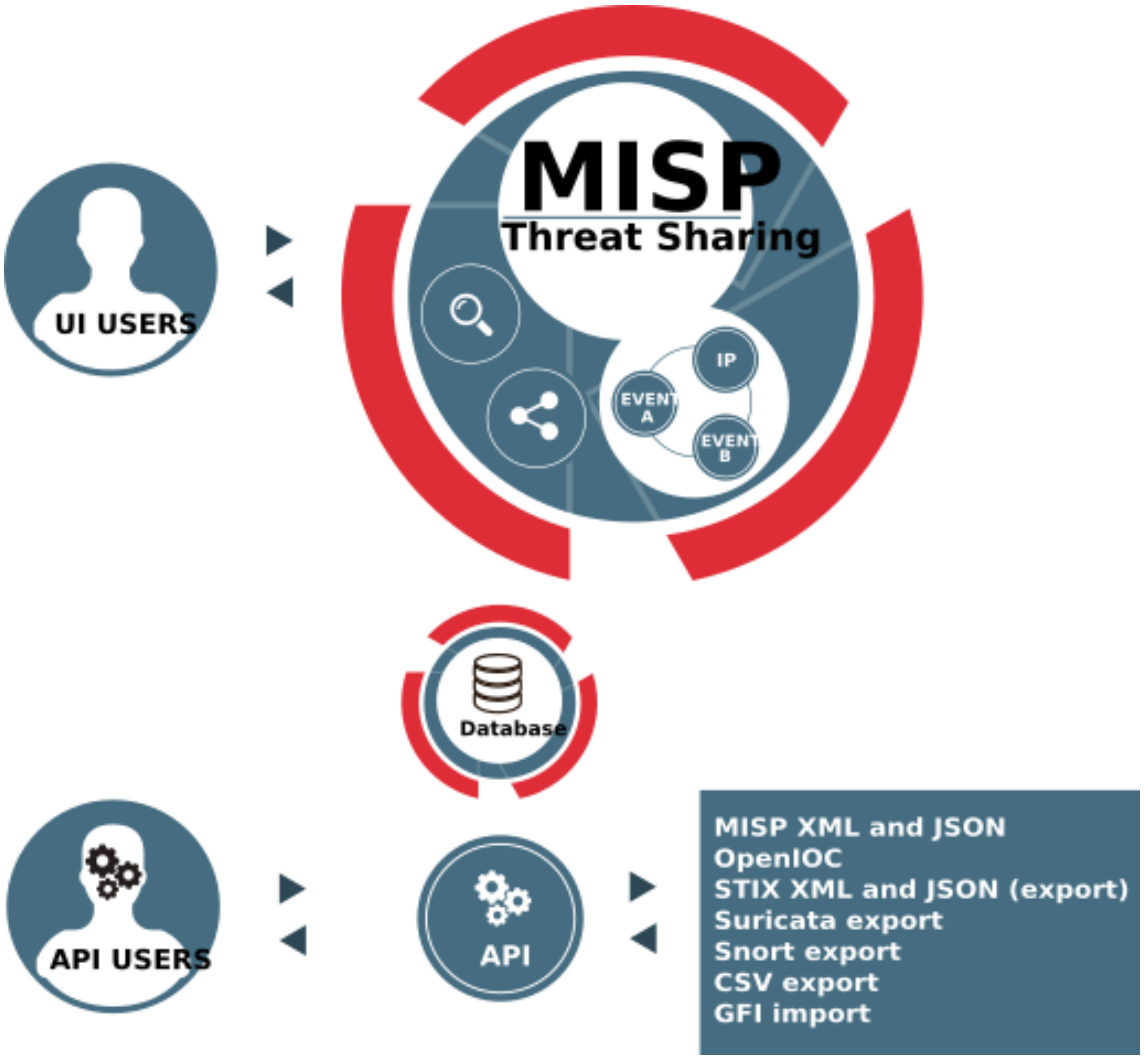

Ilustración 12 - Esquema de funcionamiento de MISP

Entre las principales funcionalidades que podemos encontrar en su página oficial [35] encontramos:

- Permite disponer de una base de datos de indicadores y de IoCs que permite guardar información tanto técnica, como no técnica sobre las muestras de malware, incidentes, atacantes e inteligencia.
- Correlación automática encontrando relaciones entre atributos e indicadores de malware, campañas de ataque o análisis.

- Proporciona un modelo de datos flexible donde los objetos complejos se pueden expresas y vincular para expresas amenazas de inteligencia, incidentes u otros elementos conectados.
- Una interfaz de usuario intuitiva donde los usuarios pueden crear, actualizar y colaborar en eventos, atributos y eventos. Interfaz gráfica para navegar entre eventos y sus diferentes correlaciones.
- Exportar información para generar IDS (Suricata, Snort, por ejemplo), OpenIOC, texto sin formato, CSV, MISP XML o JSON.
- Importación de datos masiva, por lotes, importación de texto libre y desde otras fuentes.
- Intercambio de datos automático y sincronización con otras partes y grupos de confianza utilizando MISP.
- Herramienta flexible para importar e integrar en MISP cualquier fuente de inteligencia u OSINT de terceros.
- API flexible para integrar MISP con tus propias soluciones.
- Clasificación personalizable o utilización de clases ya definidas. La clasificación puede ser local pero también se puede compartir entre otras instancias de MISP.

En nuestro caso, utilizaremos la información generada a partir de Wazuh y TheHive para importarla dentro de MISP.

2.4. Otros elementos

Como elemento adicional e innovador para nuestra plataforma, se ha incorporado **n8n** [36], una herramienta de código abierto, hecha en Nodejs, que permite automatizar tareas. Funciona mediante elementos denominados nodos que se pueden interconectar entre sí para crear diferentes automatizaciones o workflows, permitiendo, por ejemplo, utilizar condicionales para controlar el flujo. Además, también permite conectarse a servicios externos, ya sea haciendo uso de conectores oficiales, mediante webhooks o a través de una REST API.

Existe una gran comunidad de usuarios de n8n que han creado multitud de integraciones con diferentes servicios. También cuenta con una librería de worklows creados por la comunidad que podemos utilizar como plantilla para crear los nuestros. Para el caso que nos ocupa, tenemos integraciones para TheHive y Cortex.

Detrás de n8n hay una empresa **n8n Gmbh**, creada en 2019 y con sede en Alemania. Bajo el mismo criterio con el que hemos seleccionado otras herramientas, el que haya una empresa detrás, junto con una gran comunidad, hacen que sea una opción viable y madura para que sea implantada en empresas y organizaciones.

Hay tres modalidades en las que podemos utilizar n8n:

- **Modo Cloud** o uso en la nube. Es la versión de pago de la aplicación, en la que hay tres modalidades de uso, que varían en la cantidad de

workflows que podemos crear y el número de usuarios disponibles. Todas ellas cuentan con soporte.

- **Modo *Self-Hosted***, en esta modalidad nos podemos descargar la aplicación para instalarla en un servidor, siendo nosotros quiénes la gestionemos. Tenemos la opción de instalarla mediante Docker o como un paquete de npm. Aquí el número de usuario y workflows es ilimitado, dependiendo solo de los recursos dedicados al servidor.

- **Modo aplicación de escritorio**, un modo gratuito en el que se instala la aplicación en un ordenador de escritorio, ya sea macOS o Windows. Está orientado a su uso para testing o para llevar a cabo ejecuciones locales.

2.5. Elementos que compondrán nuestra plataforma

Una vez vistos y analizados los diferentes elementos que podrían formar parte de nuestra plataforma de respuesta a incidentes, tenemos que escoger cuáles elegimos implementar.

Para el desarrollo del TFM se han propuesto una serie de casos de uso que consisten en simular varios escenarios, a pequeña escala por supuesto, que se podrían encontrar perfectamente en cualquier organización o empresa:

- Gestión de un incidente con malware detectado por un EDR y que es reportado al SIEM, creando un caso, investigándolo, añadiendo nueva información si es necesaria y resolver el caso.

- Gestión de un incidente con amenazas detectables por red con nuestro NIPS, reportado al SIEM, creando un caso, investigación añadiendo nueva información si es necesaria y resolver el caso.

- Gestión de un incidente con información de eventos recogidos por el SIEM, creación del caso, investigación, añadiendo nueva información si es necesaria y resolver el caso.

El caso de uso escogido para este TFM sería el primero propuesto con algunas diferencias o añadidos. Lo he escogido por su similitud a una situación que tengo en mi propio entorno laboral y que, quién sabe, puede que acabe teniendo que implantar a partir de los conocimientos adquiridos en el desarrollo de este TFM.

El caso de uso propuesto sería un firewall que conectaría una empresa u organización con Internet, estando detrás de dicho firewall toda la infraestructura interna de la propia empresa. Este firewall también haría de enrutador. A partir de este punto, monitorizaríamos la actividad firewall y, por ende, de la red, utilizando las herramientas que tenemos a nuestra disposición. Podríamos también incluir alguna máquina que juegue el rol de workstation de una oficina y

que está dentro de una red privada detrás de dicho firewall o se pueda conectar vía VPN, por ejemplo.

Estos serían los componentes de nuestra plataforma SIRP:

- Wazuh agent
- Wazuh server
- Wazuh indexer
- Wazuh dashboard
- Suricata
- TheHive
- Cortex
- MISP
- n8n

Estos serían los componentes elegidos para implementar la plataforma de respuesta a incidentes.

Entre Snort y Suricata, se ha escogido Suricata por ser un software más moderno y tener más capacidades que Snort. Además, es compatible con todas las reglas de Snort.

Como hemos elegido Wazuh y vistos los cambios hechos en sus versiones más recientes, descartamos utilizar el stack de ELK, al tener una solución completa solo utilizando componentes de Wazuh.

Como firewall se puede optar por utilizar Pfsense, una solución robusta, sergura y muy utilizada para implementar en un mismo sistema un router y un firewall. Además, tiene la capacidad de instalar herramientas externas como Suricata.

2.6. Diseño de la solución

La solución para nuestra plataforma sería la implementación de los elementos escogidos como una solución automatizada, escalable y perfectamente instalable en los sistemas, ya sean físicos o en la nube, de cualquier organización. Ofrecer una plataforma de respuesta a incidentes completamente integrada y configurada en la que todos los elementos estuviesen listos para funcionar. La idea sería dejar que el propio usuario fuera quién gestionase las incidencias y los casos haciendo uso de esta plataforma, en ningún caso se ofrecería la gestión. Se plantea de esta forma para que sea

algo más flexible y se pueda adaptar a los requerimientos y necesidades que pueda tener una empresa u organización. Pudiendo, entre otras cosas, escalar la plataforma según lo necesiten.

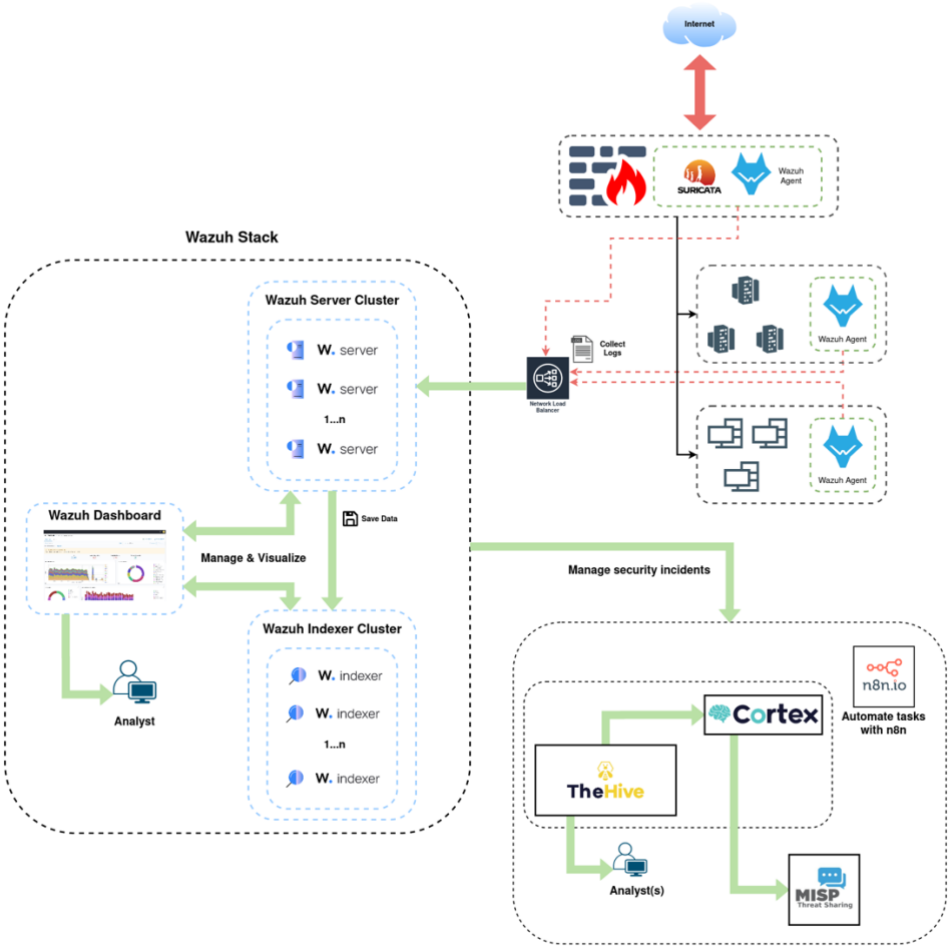

Ilustración 13 - Diagrama de la solución planteada

Teniendo en cuenta este diseño, el ciclo de vida de un incidente a través de esta plataforma sería el siguiente:

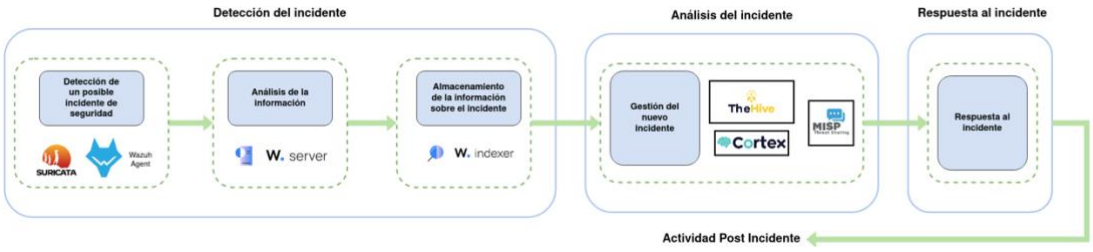

Ilustración 14 - Ciclo de vida de un incidente en la plataforma

Los agentes de Wazuh y Suricata (u otro IDS de red) serían los elementos principales en la fase de Detección de incidentes. Cuando éstos detectasen un incidente, lo reportarían al servidor de Wazuh, que analizaría la información

recibida, la procesaría y la almacenaría. Tanto el servidor de Wazuh como el Indexer, también forman parte de esta fase de detección.

Una vez recibida esta información, Wazuh reportaría el incidente a TheHive, para que abra un nuevo caso para gestionar de manera adecuada el incidente. Para la gestión del incidente se ayudaría también de otras herramientas como Cortex y MISP. Esta sería la fase de Análisis, de ver lo que ha ocurrido, analizarlo y plantear la solución al respecto. Aunque se podría incluir también a TheHive en la fase de Respuesta, esta fase puede variar muchísimo dependiendo de la organización y del tipo de incidente, por tanto y en mi opinión, quedaría fuera del ámbito de nuestra plataforma.

Una vez se por solucionado a nivel técnico la incidencia, podemos decir que TheHive vuelve a estar presente en la fase de Actividad Post Incidente, pues es aquí donde damos por cerrado el caso y documentamos lo que ha ocurrido.

3. Implementación

3.1. Opciones de implementación

Como requisitos para la implementación tiene que haber posibilidad de alta disponibilidad y que tiene que ser una solución escalable, además de disponer de un despliegue y configuración lo más automatizado posible.

Si revisamos las opciones de instalación de cada uno de los elementos, vemos que cumplimos con los requisitos, puesto que en todos ellos existe la posibilidad de instalación en modo cluster para la escalabilidad y para disponer de alta disponibilidad, aunque hay que utilizar alguna herramienta externa como HAproxy, como balanceador de carga.

En mi caso en concreto, no dispongo de tantas máquinas físicas como para instalar todos los servicios, aunque hoy en día, instalar servicios en máquinas físicas ya no es lo habitual, pues se ha optado por virtualizar todo o casi todo. En mi caso, esto pasa por utilizar algún software de virtualización como puede ser VirtualBox.

Dada la naturaleza de la plataforma y del proyecto en sí, no podríamos optar por el uso de los servicios a través de sus propias plataformas o nubes, al ser un entorno cerrado sin posibilidad de instalación desde cero en una máquina, o de automatizar el despliegue, por ejemplo. La idea es que esta plataforma de respuesta a incidentes de seguridad se pueda implantar en una organización desde cero, personalizada y adaptada a los requisitos y recursos de los que se puedan disponer, intentado depender lo mínimo posible de herramientas externas, como nubes u otros servicios de SaaS (*software as a service*).

Por tanto, tenemos las siguientes opciones de implementación:

3.1.1. Instalación manual

Todas las herramientas seleccionadas cuentan con opción de instalación manual, es decir, una instalación o bien a través de los diferentes sistemas de paquetes, como apt, yum, etc. o, a través de repositorios. También contamos con la posibilidad de descargarnos el código fuente para compilar y construir el software desde cero, lo cual puede ser útil si quieres adaptarlo a un entorno en particular o utilizarlo con unas opciones muy concretas o necesitas el mayor rendimiento posible.

También encontramos la opción de descargar una máquina virtual completa con la herramienta ya instalada y configurada, lista para utilizarse. Este caso puede ser útil para un entorno de pruebas o para practicar, donde no queramos dedicar tiempo a instalar y configurar cada componente. En nuestro caso, esta opción no nos sirve.

3.1.2. Instalación automatizada con Ansible

Ansible [37] es una herramienta de código abierto, propiedad de RedHat, que permite automatizar configuraciones y despliegues. Permite, de una forma centralizada, gestionar una gran cantidad de máquinas, realizando un despliegue de aplicaciones o gestionando la configuración de una manera automatizada. Todo se realiza a través de conexiones SSH y a través de los llamados "*playbooks*", que son ficheros en formato YAML, donde podemos instalar paquetes, o realizar distintas configuraciones. Esto es lo que se conoce como *infraestructure as a code*.

Para el caso que nos ocupa, el de la plataforma de respuesta a incidentes, necesitamos una herramienta similar de orquestación que nos permita gestionar la configuración de diferentes máquinas, de si tenemos que desplegar una nueva máquina para escalar un servicio, se haga de forma automatizada y se instalen y configuren los servicios necesarios. No solo eso, sino si tenemos que desplegar desde cero la plataforma, poder hacerlo de la manera más rápida y eficiente posible, sin tener que ir máquina a máquina configurando e instalando los diferentes servicios.

Aunque existen otros servicios de aprovisionamiento, como Chef, Salt o Puppet, Ansible es el que más se adapta para el desarrollo de este proyecto, al ser el más sencillo y rápido de utilizar, basta con declarar nuestra configuración en un fichero YAML, sin necesidad programar, de utilizar o dedicar servidores, agentes u otros elementos. Además, lo podemos sin tener que pagar. La idea es tener automatizado el despliegue y la configuración de las máquinas y servicios, la herramienta que se utilice ya dependerá de cada organización, en nuestro caso, optaremos por Ansible.

Casi todas las herramientas que forman parte de la plataforma tienen documentada la instalación mediante Ansible, otro punto a favor de utilizar esta herramienta sobre las demás, pero, aunque no tuvieran documentada esta opción, nosotros podemos, o bien buscar algún playbook ya existente o, crear uno desde cero, adaptado a nuestras necesidades.

3.1.3. Instalación con Docker

Docker es la herramienta por excelencia para gestionar contendores, la más utilizada hoy día y la base de otras herramientas de orquestación de contenedores como Kubernetes. Docker suele ser una herramienta muy utilizada para desarrollar y desplegar aplicaciones, pues al funcionar dentro de contenedores, están aisladas del resto del sistema operativo anfitrión y de otros contenedores, lo cual permite, entre otras ventajas, tener múltiples versiones de una misma aplicación, aislar por completo unas aplicaciones de otras y no "ensuciar" el sistema operativo anfitrión con multitud de librerías, aplicaciones y otros programas que pueden provocar problemas de compatibilidades. A diferencia de la virtualización, aquí no hay que virtualizar un hardware ni un kernel, con lo cual, todo el proceso es más ligero y todos los despliegues son más rápidos, pudiendo mover contenedores de aplicaciones de una máquina a otra sin necesidad de instalar nada.

Para nuestro caso, utilizar Docker sería una buena solución, ya que permite realizar una instalación automatizada y escalable, pudiendo disponer de alta disponibilidad. Además, todas las herramientas de la plataforma tienen una modalidad de instalación bajo Docker.

Para utilizar Docker en un entorno de producción deberíamos hacerlo utilizando algún sistema de orquestación de contenedores, como Kubernetes o Docker Swarm. Estos sistemas nos hacen la vida más fácil cuando tenemos multitud de contenedores que además trabajan entre sí, es decir, nos ayudan a realizar toda la fontanería para que todos los contenedores se comuniquen y trabajen entre sí. Además, proveen de sistemas para levantar y eliminar contenedores según sea necesario, entre otras cosas.

3.1.4. Modo de instalación escogida

Tal y como se ha orientado el proyecto se ha escogido la opción de implementar la plataforma sobre máquinas virtuales. La idea para probar funcionalidad del proyecto es desplegar una máquina Pfsense que haga el papel de router/firewall, algo que tienen todas las organizaciones para conectarse a Internet y a su vez protegerse, conectar varias máquinas a él, ya sean máquinas de la plataforma o servidores o máquinas de usuarios finales, por ejemplo, e intentar en la medida de lo posible replicar un entorno real.

La idea de optar por máquinas virtuales y no por Docker, por ejemplo, es poder tener un mayor control en aspectos como las redes. Dado que se virtualizará un Pfsense, la forma escogida para acceder y trabajar será a través de una VPN y de ahí, accedemos a las máquinas. También se ha escogido esta opción por la posibilidad de probar la instalación y configuración en modo cluster de las herramientas, que, aunque no se pueda implementar por falta de recursos, sí que se pueda validar dicha opción, que sería imprescindible en muchos casos, y documentarla y automatizarla.

Otra razón es que, aunque conozco Docker, no he trabajado con Kubernetes, y me siento más cómodo trabajando con máquinas virtuales. Ya son bastantes los elementos nuevos con los que trabajar y poner en funcionamiento que no quiero añadir otro que me pueda quitar tiempo para dedicarle al resto.

3.2. Recursos

Si revisamos los requerimientos de hardware para cada uno de los elementos que forman parte del proyecto encontramos que:

- Pfsense pide como mínimo una CPU de 500Mhz y 512MB de memoria.
- Wazuh Server pide como mínimo 2 cores de CPU y 2 GB de memoria.
- Wazuh Indexer pide como mínimo 2 cores de CPU y 4GB de memoria.

- Wazuh Dashboard pide como mínimo 2 cores de CPU y 4GB de memoria.
- TheHive pide como mínimo 4 cores de CPU y 16GB de memoria.
- Cortex pide como mínimo 4 cores de CPU y 16GB de memoria.
- MISP pide como mínimo 2 cores de CPU y 8GB de memoria.
- El servidor de n8n pide como mínimo 1 core de CPU y 1GB de memoria.

Como vemos, hacen falta bastantes recursos para montar un laboratorio e implementar la plataforma. Sobre todo, para montar los servidores de TheHive y Cortex. Revisando la documentación de estos servicios, necesitan utilizar Java e instalan ElasticSearch como motor de indexado, de ahí la necesidad tan alta de memoria RAM.

Para montar el proyecto se dispone de los siguientes recursos:

- Ordenador de escritorio con un AMD Ryzen 5 5600x, con 6 Cores y 12 hilos y 40GB de memoria RAM.
- Un minipc IntelNUC con un i5, cuatro Cores y 16GB de memoria RAM.

Si sumamos todos los requerimientos mínimos de todos los elementos, vemos que sobrepasan los recursos de los que podemos disponer. Aun así, tenemos algunas opciones para implementar todo con los recursos de los que disponemos.

Wazuh dispone de un modo de instalación “todo en uno”, llamado Wazuh Manager [38], el cuál integra los tres elementos, el servidor, el indexador y el dashboard. Según la documentación, es capaz de gestionar hasta 100 agentes y los recursos varían según la cantidad:

Agents	CPU	RAM	Storage (90 days)
1-25	4 vCPU	8 GiB	50 GB
25-50	8 vCPU	8 GiB	100 GB
50-100	8 vCPU	8 GiB	200 GB

Ilustración 15 - Requerimientos de Wazuh Manager

En nuestro laboratorio no llegaremos a desplegar más de 25 agentes, de nuevo, por temas de falta de recursos, así que podemos desplegar Wazuh en una máquina virtual con 4 Cores y 8 GB de memoria. El almacenamiento en principio no es problema. Por tanto, ya hemos reducido en parte los recursos necesarios.

En cuanto a TheHive y Cortex, necesitan tanta memoria para instalar ElasticSearch y java. Es cierto que podemos instalar ElasticSearch por separado, pero no nos vamos a librar de los recursos necesarios. Revisando la documentación de la instalación, vemos que tenemos que descargar un script de instalación el cual, antes de nada, comprueba si se cumplen los requisitos mínimos. Básicamente comprueba la cantidad de memoria RAM de la máquina con un valor guardado en una variable. Podemos cambiar dicho valor y rebajarlo un poco, de 16GB a 12GB o incluso a 10GB. Se ha hecho la prueba y sigue quedando bastante memoria libre.

El estar jugando con los recursos de esta manera no es lo mejor, pero si podemos adaptar el software a lo que tenemos y sigue funcionando como debe hacerlo, no debería haber ningún problema. Más si cabe cuando la cantidad de recursos computacionales disponibles en muchas organizaciones es escasa y además, es un recurso muy caro, por tanto, es normal que se intente optimizar todo lo que se pueda, sin pasarse.

Al hilo de esta cuestión, vamos a añadir otros dos elementos al laboratorio, dos elementos que nos ayudarán a monitorizar el consumo de recursos por parte de los servidores y comprobar si, efectivamente, hemos hecho bien en disminuir los recursos o no. Los elementos que vamos a añadir son Prometheus [39]y Grafana [40].

Prometheus es una herramienta de monitorización muy utilizada, sobre todo en el mundo de Kubernetes, que nos permite monitorizar todos los aspectos importantes de un servidor. Funciona mediante unos programas, llamados *exporters*, que recogen métricas en las máquinas donde se ejecutan, para luego Prometheus recolectarlas y almacenarlas. Es decir, no son los *exporters* quiénes envían la información al servidor, sino al revés, es el servidor quién va a las máquinas a recoger las métricas. Esto es así gracias a que los exportes dejan las métricas accesibles a través de un puerto determinado.

Por otro lado, Grafana una herramienta de visualización, de lo que sea, es decir, desde datos sobre el tiempo atmosférico, hasta datos de la bolsa de valores a métricas de CPU, RAM, etc. La usaremos para que lea la base de datos de Prometheus y a través de unos paneles que nosotros configuraremos, mostrar los consumos a lo largo del tiempo de CPU, memoria, etc.

Todo esto es por falta de memoria RAM, no de CPU. La CPU en virtualización se puede compartir entre las diferentes máquinas virtuales, no así la memoria, que se reserva y es de uso exclusivo de dicha máquina.

Como sistema operativo base se utilizará Ubuntu Server 20.04, por ser el más compatible entre todos los elementos. Esto es así porque MISP recomienda encarecidamente utilizar esta y no otra distribución. El único caso distinto es Pfsense, que es FreeBSD. Pero en este caso estamos utilizando Pfsense como prueba de router/firewall, pero en una organización puede haber otro tipo de router o firewall u otra configuración, por tanto, no es elemento que forme estrictamente parte de la plataforma de respuesta a incidentes, más bien un punto desde el que podemos leer datos y comprobar que funciona la integración

de todos los elementos para así, en otra situación, trabajar con datos procedentes de otra fuente.

3.3. Esquema del entorno de simulación

Una vez vistos los recursos disponibles y los requerimientos de cada uno de los elementos, el esquema del entorno de simulación quedaría de la siguiente manera:

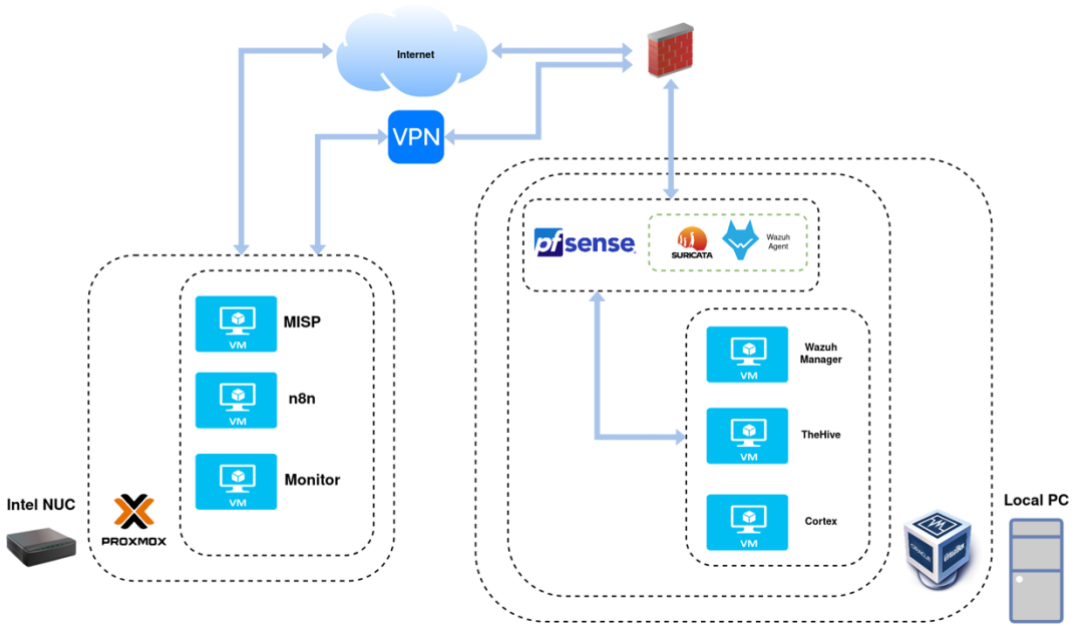

Ilustración 16 - Diagrama de la infraestructura del laboratorio

Como se puede observar en el diagrama, hay un elemento más. Para virtualizar en el PC de escritorio voy a utilizar VirtualBox, pero en el IntelNUC voy a utilizar Proxmox. La idea es que el minipc esté conectado a la corriente y a la red, sin teclado ni ratón ni pantalla. Por tanto, la forma de gestionar, tanto el minipc como las máquinas virtuales sería a través de red. Proxmox es ideal para esto. Es un entorno de virtualización de código abierto que ya he utilizado con anterioridad. Es una distribución modificada de Debian que permite el despliegue y la gestión de máquinas virtuales y contenedores. Incluye, entre otras cosas, una consola Web tanto para el anfitrión, como para las máquinas virtuales, por tanto, es como si nos estuviéramos conectando con una pantalla.

Se ha hecho esta distribución de máquinas por el tema de carecer recursos para implantar todo en un único ordenador. Se ha dejado Wazuh, TheHive y Cortex en dentro de VirtualBox para que estén detrás del Pfsense y tanto Suricata como el agente tengan conexión directa con el servidor de Wazuh y éste, a su vez, con TheHive para abrir un caso nuevo, además de ser donde hay más recursos disponibles. La comunicación entre ambos grupos de máquinas se hará a través de una VPN instalada en el Pfsense, siendo las máquinas virtuales del IntelNUC las que se conecten a la VPN.

El esquema de máquinas virtuales y recursos quedaría de la siguiente manera:

- 1VM – Pfsense, 1 Core, 1GB de ram, FreeBSD.
- 1VM – Wazuh Manager, 4 Cores, 8GB de ram, Ubuntu Server 20.04.
- 1VM – TheHive, 4 Cores, 12GB de ram, Ubuntu Server 20.04.
- 1VM – Cortex, 4 Cores, 12GB de ram, Ubuntu Server 20.04.
- 1VM – MISP, 2 Cores, 6GB de ram, Ubuntu Server 20.04.
- 1VM – Monitorización, 1 Core, 1GB de ram, Ubuntu Server 20.04.
- 1VM – n8n, 1 Core, 1GB de ram, Ubuntu Server 20.04.

Con este planteamiento podemos crear un laboratorio para montar nuestra plataforma de respuesta a incidentes de seguridad, integrar cada elemento y comprobar que funciona todo, desde que se detecta un posible incidente, pasando por la creación de un nuevo caso y su resolución.

Como hemos visto, todos los elementos que componen la plataforma tienen una opción de despliegue en modo cluster para escalar y para tener alta disponibilidad, pero con los recursos con los que cuento, no me es posible implementarlo. Esto no quiere decir que no lo pueda implementar por separado para comprobar su viabilidad, su puesta a punto y funcionamiento y así, documentarlo.

Esto es un entorno simulado, no un entorno real o de producción, y la idea es mostrar el funcionamiento de la plataforma, así como las posibilidades que puede ofrecer.

4. Integración

Una vez hemos visto en profundidad todos los elementos que formarán parte de nuestra plataforma de seguridad, toca el turno de instalarlos y, sobre todo, de configurarlos e integrarlos entre sí para que todo funcione correctamente.

La configuración de cada una de las herramientas seleccionadas, si utilizásemos todas las posibilidades que nos ofrecen llevaría mucho tiempo. Por tanto, hemos realizado las configuraciones mínimas necesarias para que se puedan utilizar correctamente y que puedan integrarse unas con otras. Cabe mencionar que dependerá de las necesidades de cada organización configurar las herramientas de una u otra forma, no hay una única opción.

4.1. Wazuh

Tal y como comentamos en apartados anteriores, hemos optado por la opción que instala los tres componentes en un único servidor, puesto que no tenemos recursos para plantearnos un cluster. Siguiendo la documentación, la instalación del todos los componentes se hacen de manera automática a través de un script de bash que ponen a nuestra disposición los propios desarrolladores del producto. Como hemos dicho, en esta versión se integra tanto el servidor que gestiona los agentes, como el indexador que almacena, gestiona y organiza la información, así como el dashboard, la interfaz a través de la cual se puede gestionar la herramienta y visualizar la información o realizar búsquedas sobre ella.

Para acceder a la aplicación, tenemos que hacerlo a través de la VPN que hemos configurado en nuestro Pfsense.

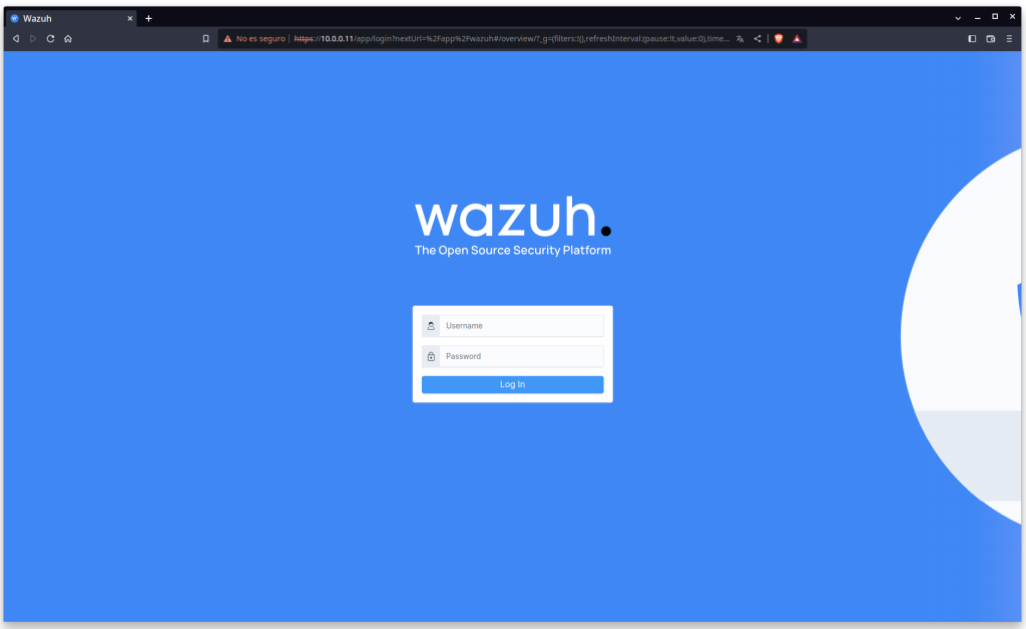

Ilustración 17 - Pantalla de login de Wazuh

Como vemos, Wazuh viene con un certificado SSL autofirmado. Para este proyecto nos puede valer, pero no para una puesta en producción, donde deberíamos tener un certificado válido. Al finalizar la instalación, se nos proporciona una contraseña para el usuario “admin” que usaremos para iniciar sesión en la aplicación:

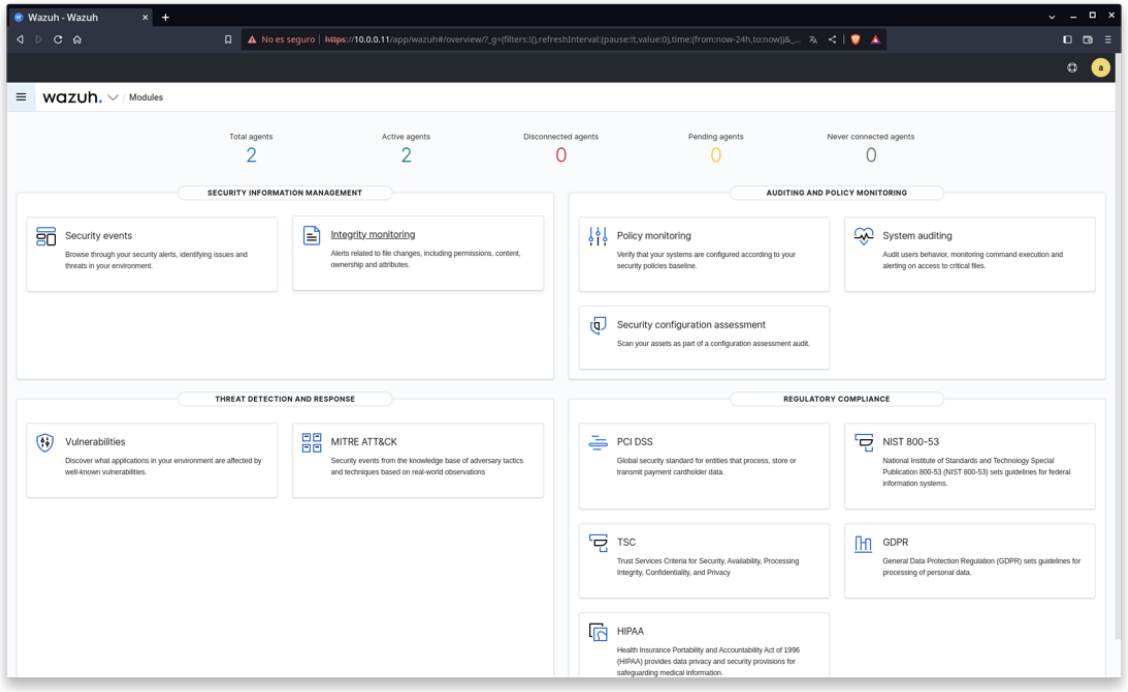

Ilustración 18 - Menú principal de Wazuh

Esta sería la página principal de la aplicación, donde podemos ver los diferentes módulos de Wazuh y un listado de los agentes desplegados y su estado actual. Por defecto no están habilitados todos los módulos, pero es algo que se puede modificar en la propia configuración de la herramienta.

En este caso, vemos que tenemos dos agentes desplegados. Uno es el agente que se encuentra instalado en el Pfsense y el otro es un agente desplegado en una máquina de Ubuntu que hace el papel de cliente.

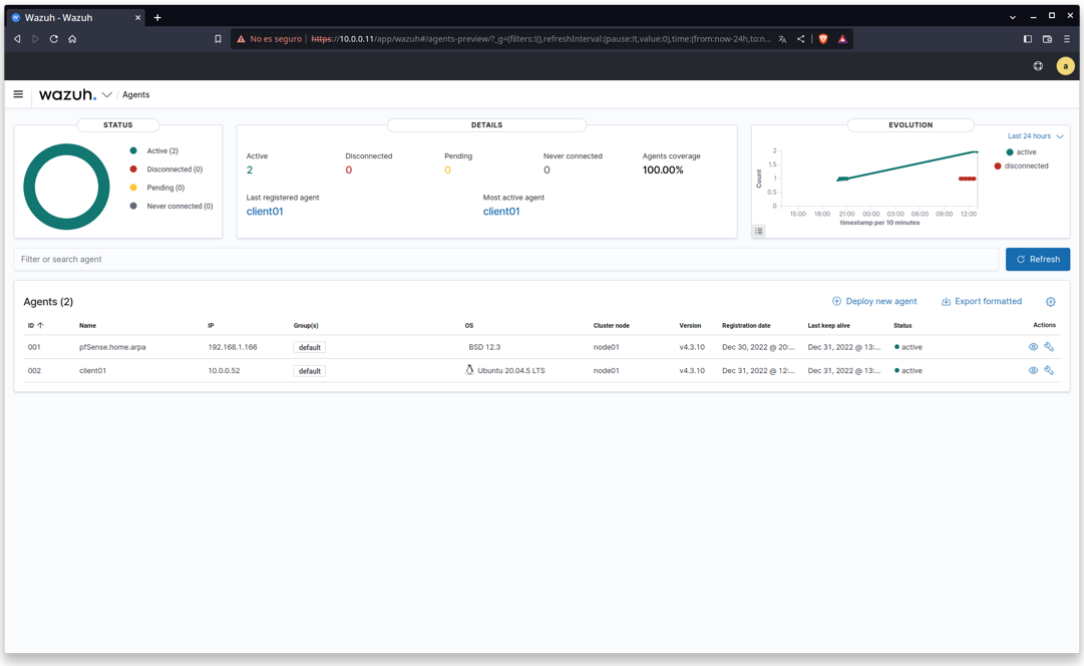

Ilustración 19 - Listado de agentes

En el listado de agentes podemos ver que nos reporta información como el hostname de la máquina, su dirección IP, sistema operativo y versión de este, el nodo del cluster con el que se comunica el agente, la versión del agente, la fecha de registro de dicho agente, o el estado, si está activo o no. Desde la pantalla de cada uno podemos ver la información con más detalle y acceder y poder modificar la configuración de agente

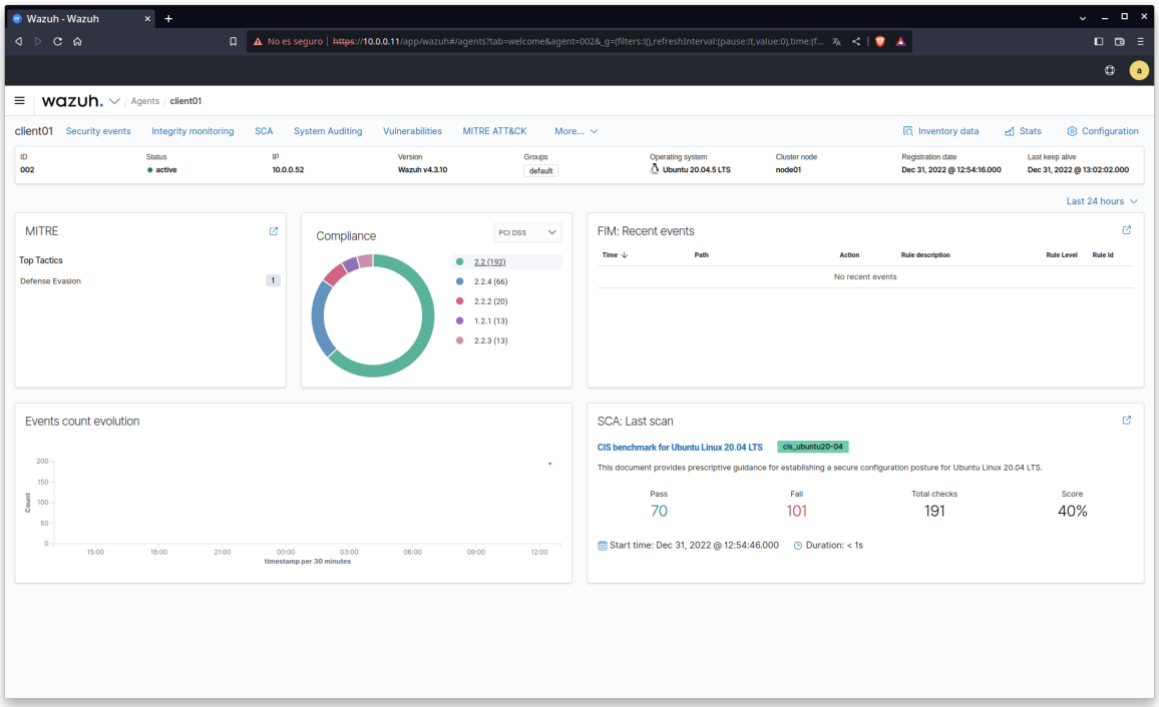

Ilustración 20 - Pantalla con la información de un cliente

Desde aquí también visualizar otro tipo de información como el hardware de la máquina, tarjetas de red, o información sobre el software instalado, así como de los procesos en ejecución.

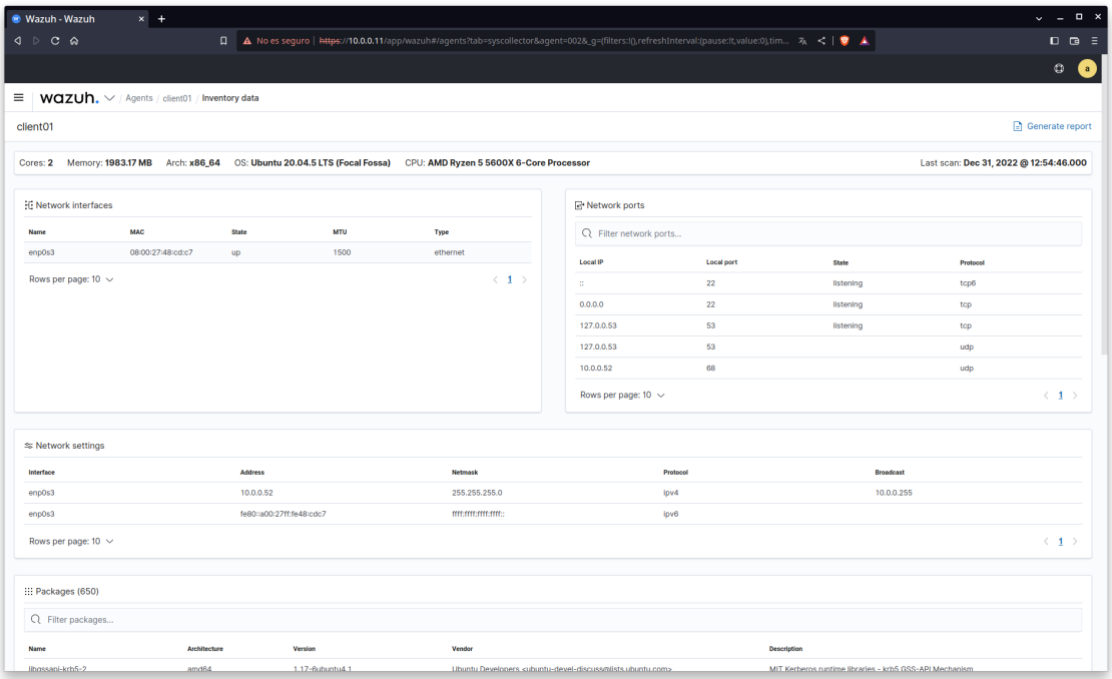

Ilustración 21 - Información sobre la máquina cliente

En la parte de “Security Events”, podemos ver la información que el agente ha recopilado de los diferentes eventos de seguridad que ocurren en la máquina donde está desplegado. Si nos vamos a nuestro Pfsense donde, además, tenemos instalado el IDS de Suricata podemos ver los distintos eventos que se registran, tanto por Suricata como por el propio Firewall de Pfsense:

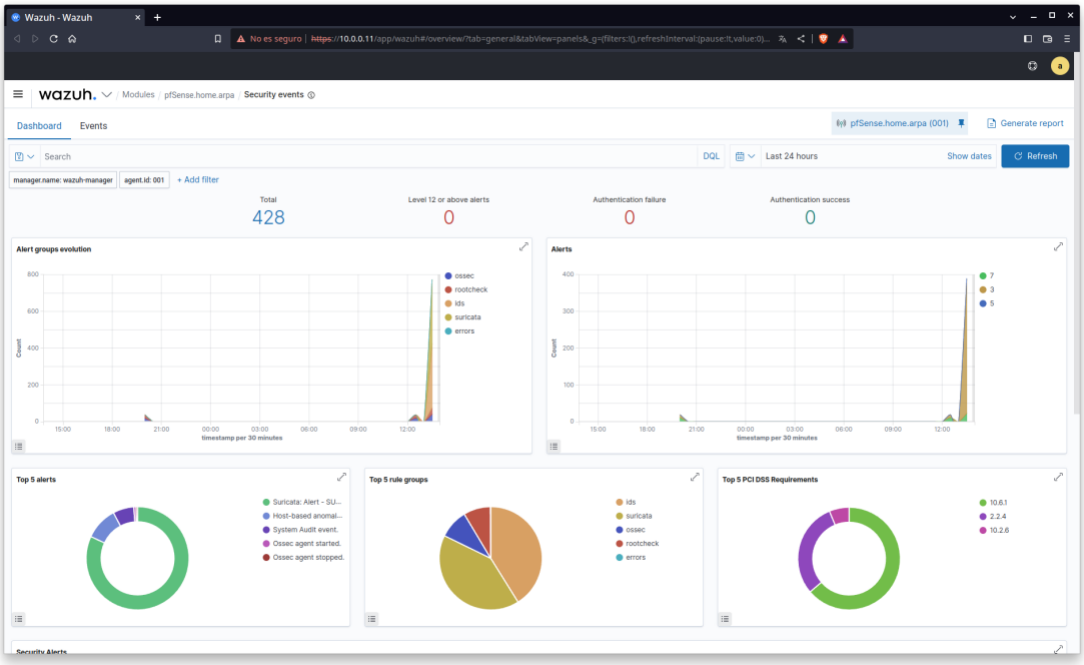

Ilustración 22 - Eventos de Seguridad reportados por el agente de Pfsense

Justo debajo podemos el registro de mensajes. Podemos desplegar cada uno de los mensajes para ver la información con más detalle. Podemos verla en tres formatos distintos, en modo tabla, en json y en modo regla que en este caso se refiere a las reglas de Suricata.

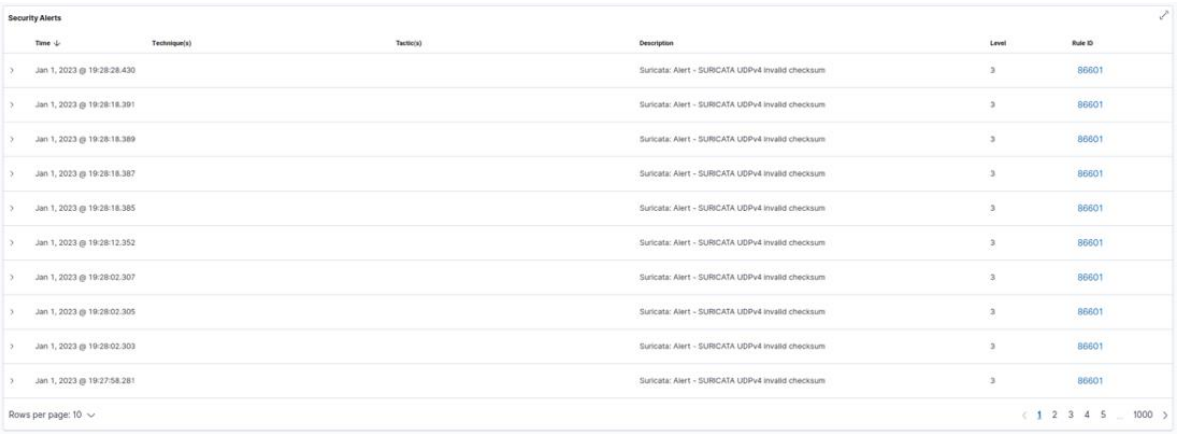

Ilustración 23 - Logs de seguridad de Pfsense

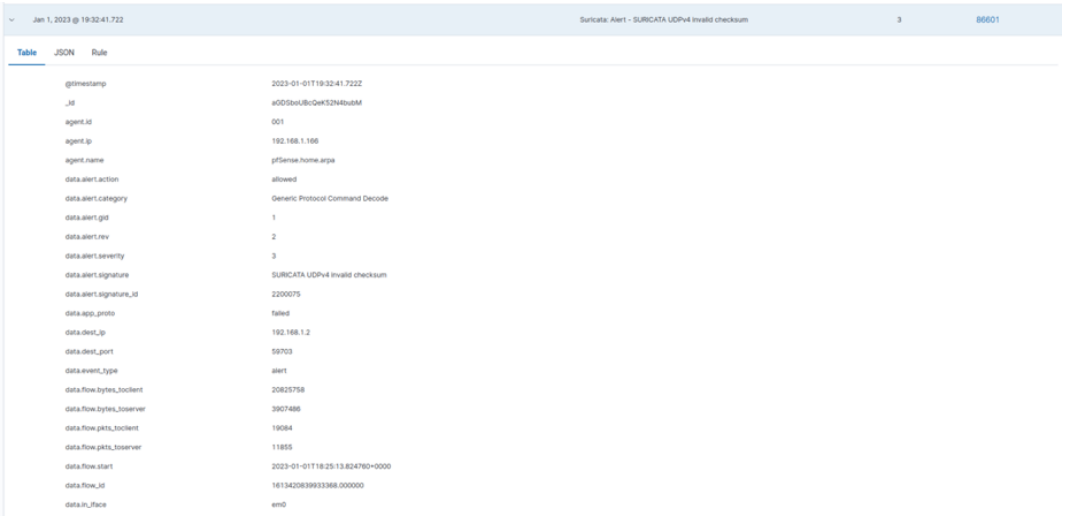

Ilustración 24 - Información detallada de un mensaje

En este caso en concreto vemos alertas de Suricata sobre un checksum inválido para UDPv4. Leyendo por Internet, mucha gente tiene problemas con esta alerta, porque mete demasiado ruido, llenando el listado de mensajes. Es por ello por lo que recomiendan suprimir la regla asociada a esta alerta. Esto lo podemos hacer desde la interfaz web de Pfsense, a través del listado de servicios, en suricata. Hecho esto, podemos ver que el listado de mensajes se limpia, pudiendo centrarnos en los mensajes importantes.

4.2. The Hive

Para la instalación de TheHive tenemos un script de instalación que nos instala todos los componentes de la herramienta, que son la base de datos de Cassandra, un Elasticsearch y el propio TheHive [41].

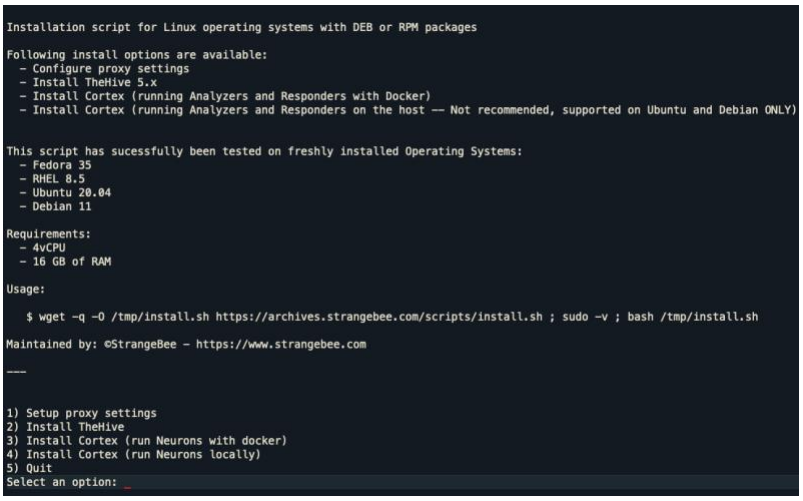

Ilustración 25 - Script de Instalación de TheHive/Cortex

Una vez instalado, podemos acceder a la herramienta:

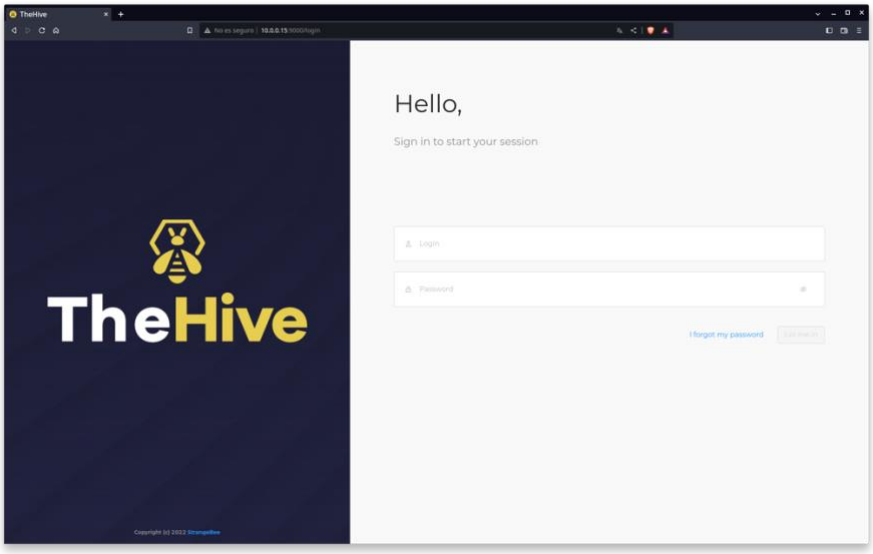

Ilustración 26 - Menú de login de TheHive

Al finalizar la propia instalación, se nos muestra que debemos acceder utilizando el usuario “admin” y una contraseña por defecto.

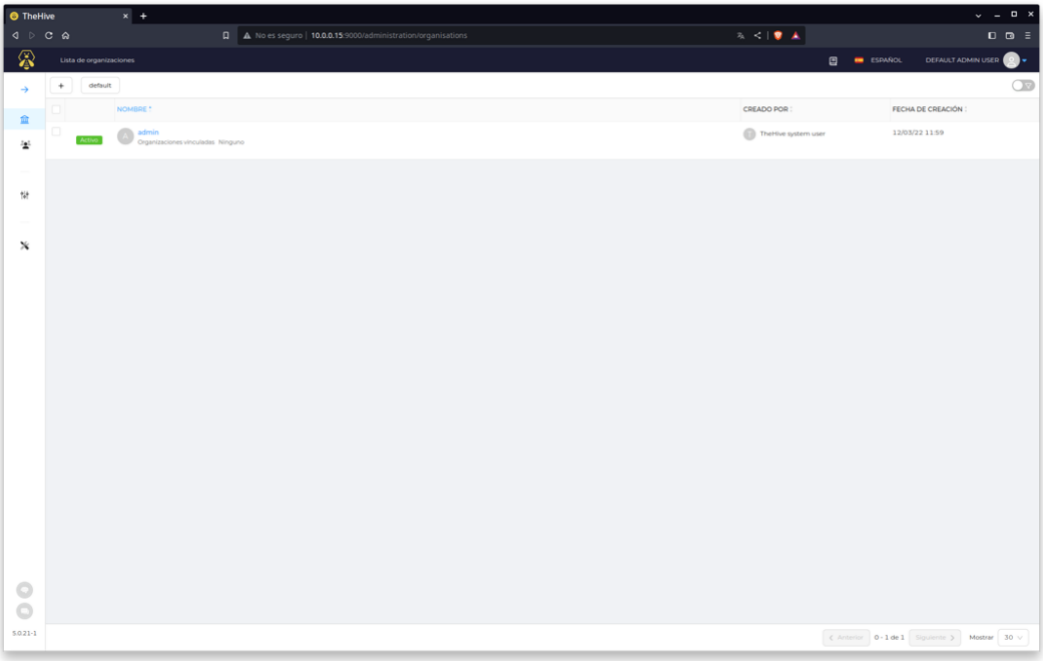

Ilustración 27 - Pantalla principal del usuario "admin"

En este caso, vemos que no tenemos una conexión SSL configurada, como si ocurría para Wazuh. En la documentación oficial explican cómo realizar la configuración para utilizar NGINX como proxy para utilizar SSL. Así que instalamos y configuramos NGINX con un certificado autofirmado, dado que esto es un entorno de pruebas. En un entorno de producción se debería utilizar un certificado válido.

Hecho esto, ahora tenemos que crear una nueva organización. Por defecto solo tenemos una, la de “Administración” de la que depende el usuario administrador y no vamos a utilizar esta para trabajar. Vamos a crear una para nuestro entorno de pruebas.

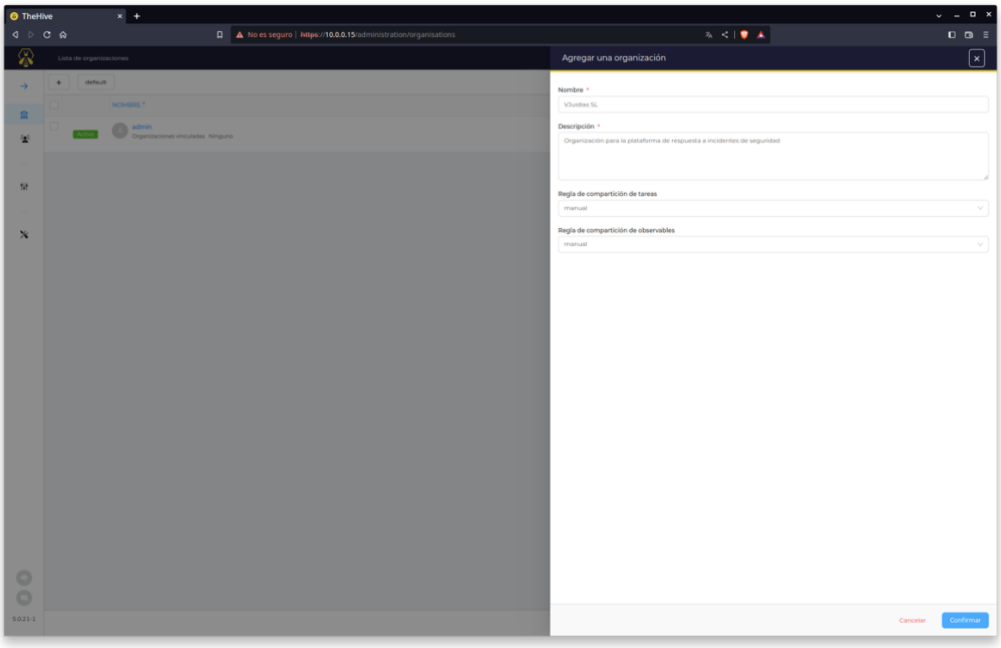

Ilustración 28 - Crear una nueva organización en TheHive

Una vez creada nuestra organización ficticia, vamos a crear usuarios dentro de ella.

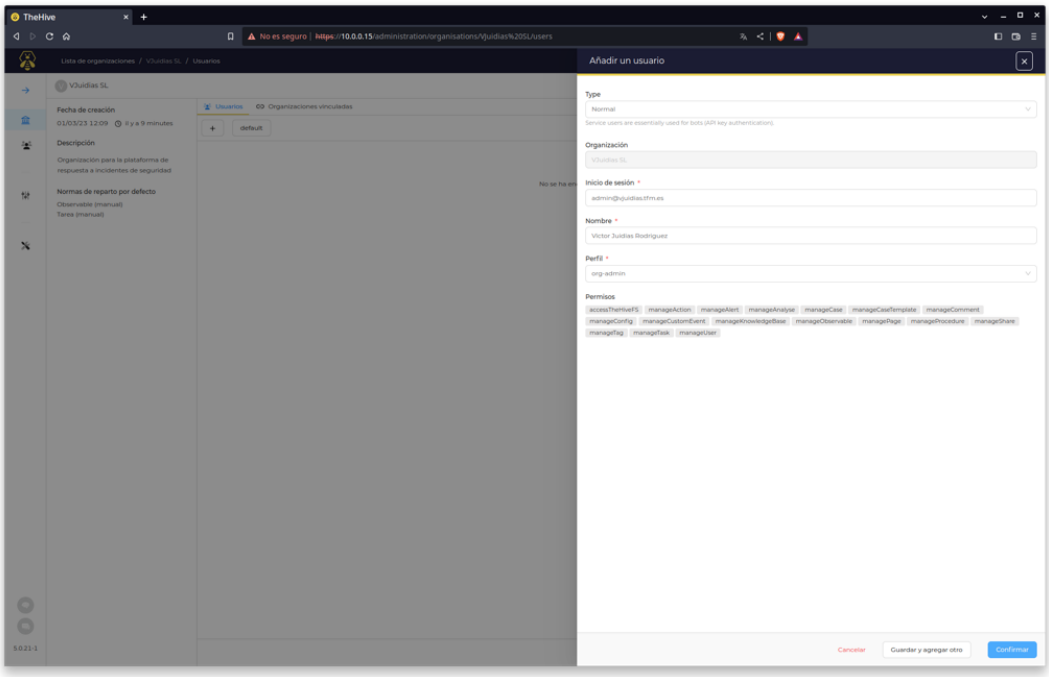

Ilustración 29 - Crear nuevos usuarios para la organización

Para este caso, vamos a crear dos usuarios, uno que haga de administrador de la Organización y otro usuario para Cortex, con permisos de solo lectura para que, mediante una API Key, Cortex se pueda conectar a TheHive.

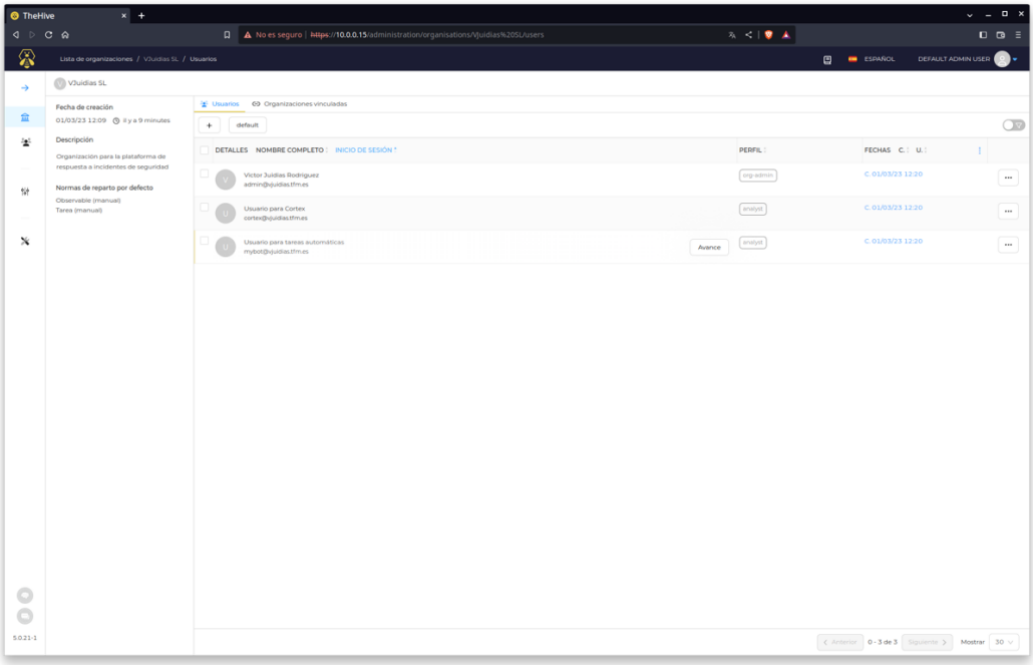

Ilustración 30 - Listado de usuarios de la organización

Si esto fuera un entorno real, deberíamos tener un usuario por analista o persona que estuviera involucrada en la resolución de los casos que puedan surgir, cada uno con un perfil adecuado.

Una vez creados los usuarios, cerramos sesión con el usuario administrador y entramos como el usuario que hemos definido como administrador de la organización.

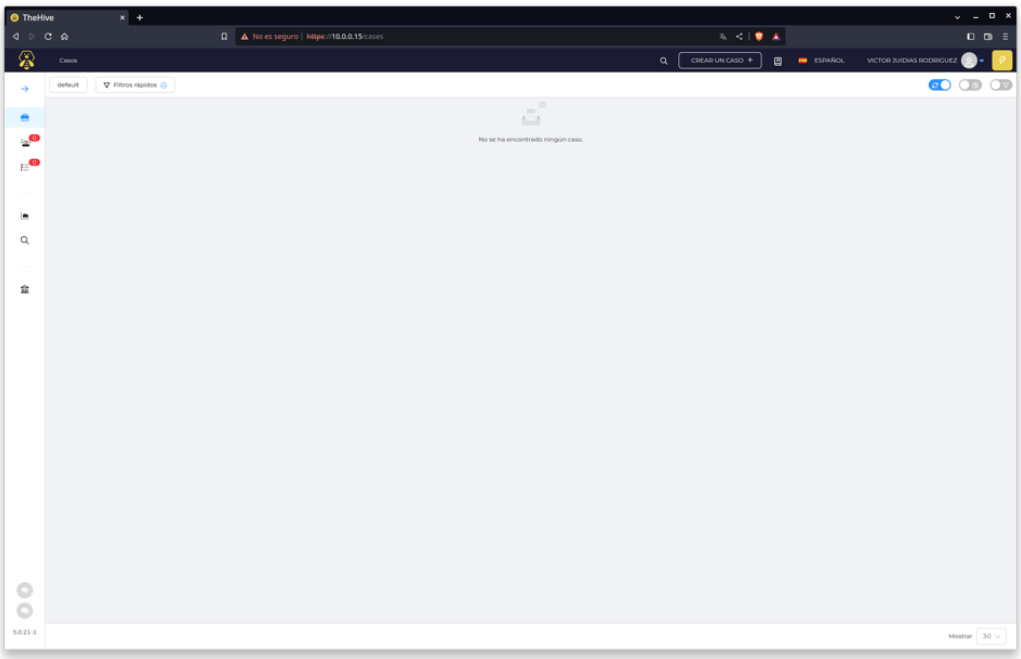

Ilustración 31 - Pantalla principal del usuario adminsitrador de la organización

La pantalla que se nos presenta con este usuario es bastante diferente, donde se nos presenta la lista de casos y donde, a la derecha, encontramos también las Alertas y Tareas. Para cada una de estas opciones tenemos la posibilidad de filtrar la información como más nos convenga. También tenemos la opción de crear dashboard o importarlos.

A partir de aquí podemos crear un caso nuevo, bien desde cero o desde una plantilla ya predefinida:

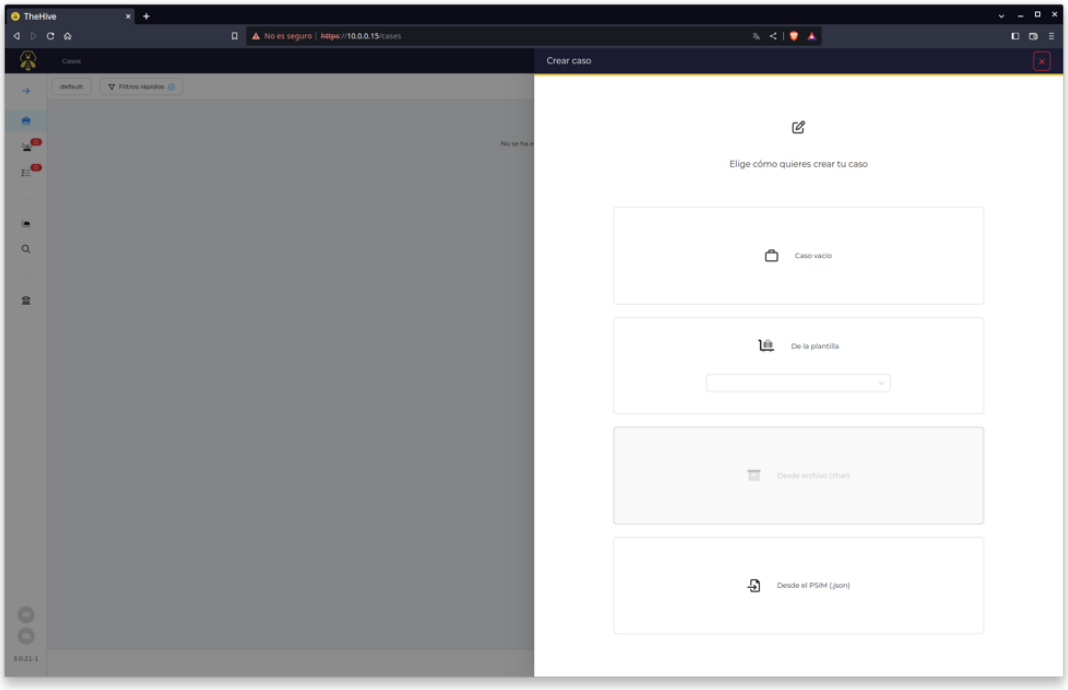

Ilustración 32 - Crear un nuevo caso

También podemos crear nuestras propias plantillas con campos y tareas personalizados, ajustados a nuestras necesidades.

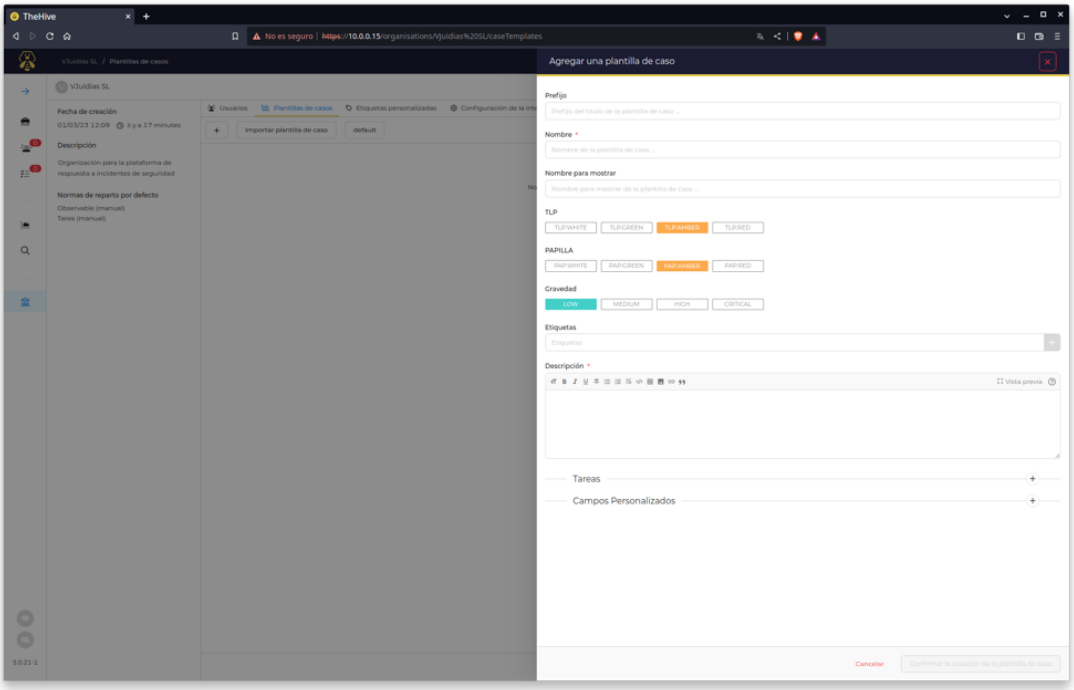

Ilustración 33 - Crear una plantilla para un caso

Lo último que nos queda por ver es que, si nos fijamos en la esquina inferior izquierda, vemos dos símbolos, el de Cortex y el de MISP. Éstos nos indican si TheHive está conectado con ellos o no.

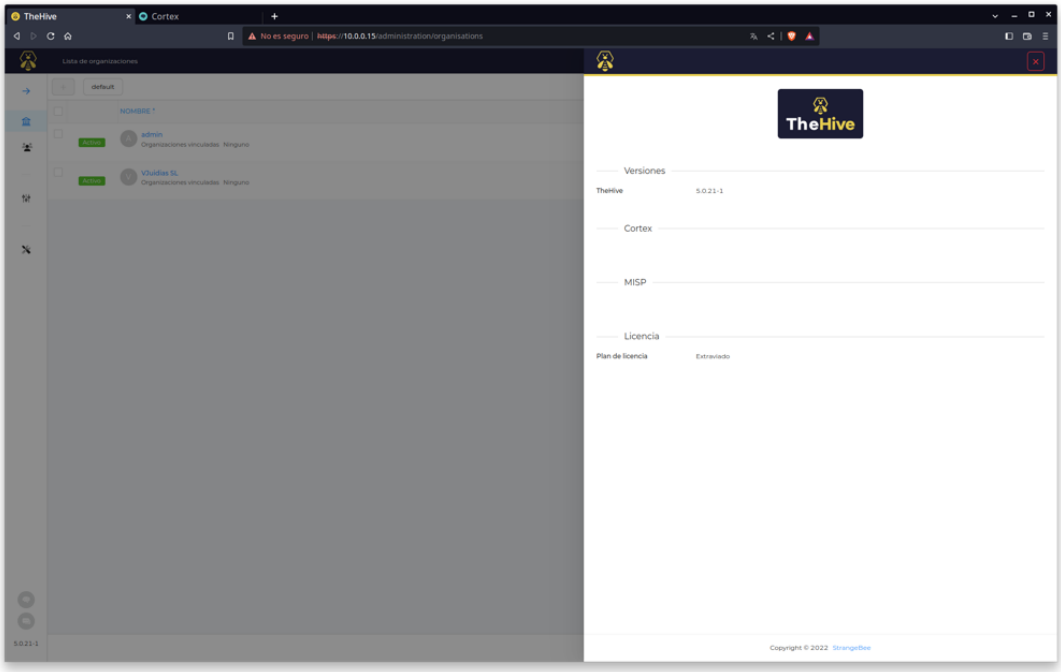

Ilustración 34 - Integración con Cortex & MISP

Para realizar la conexión tenemos que, utilizando el usuario administrado, ir a los ajustes y añadir los servidores. Tenemos que crear un usuario en las otras herramientas que debe tener una API Key para conectarse.

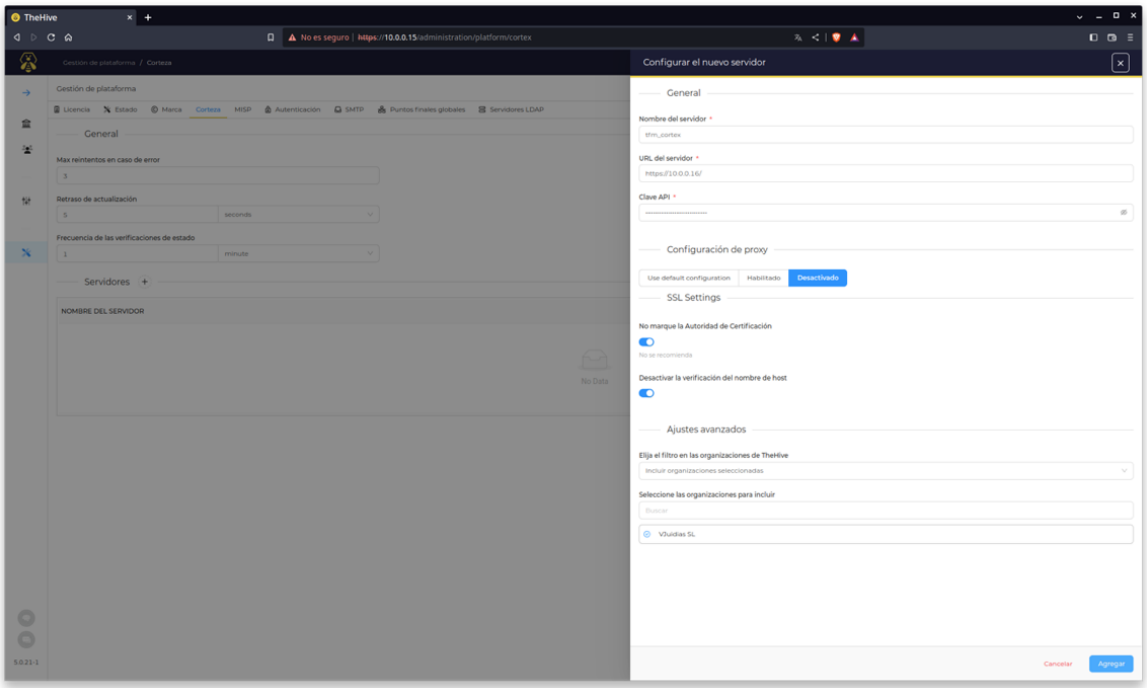

Ilustración 35 - Integrar Cortex con TheHive

4.3. Cortex

Para instalar Cortex podemos utilizar el mismo script que utilizamos para TheHive, ya que éste nos da la opción de instalar también Cortex, ya sea mediante Docker o en local.

En este caso ocurre lo mismo que con TheHive, no tenemos configurada una conexión SSL. Al igual que se hizo en el caso anterior, configuramos un servidor NGINX que actúe como proxy inverso y siguiendo la documentación oficial, lo configuramos para que funcione con Cortex.

Una vez hecho todo esto, podemos acceder a la interfaz web de Cortex. Tras la instalación se nos indica que debemos utilizar el usuario “admin” para acceder y una contraseña para el acceso.

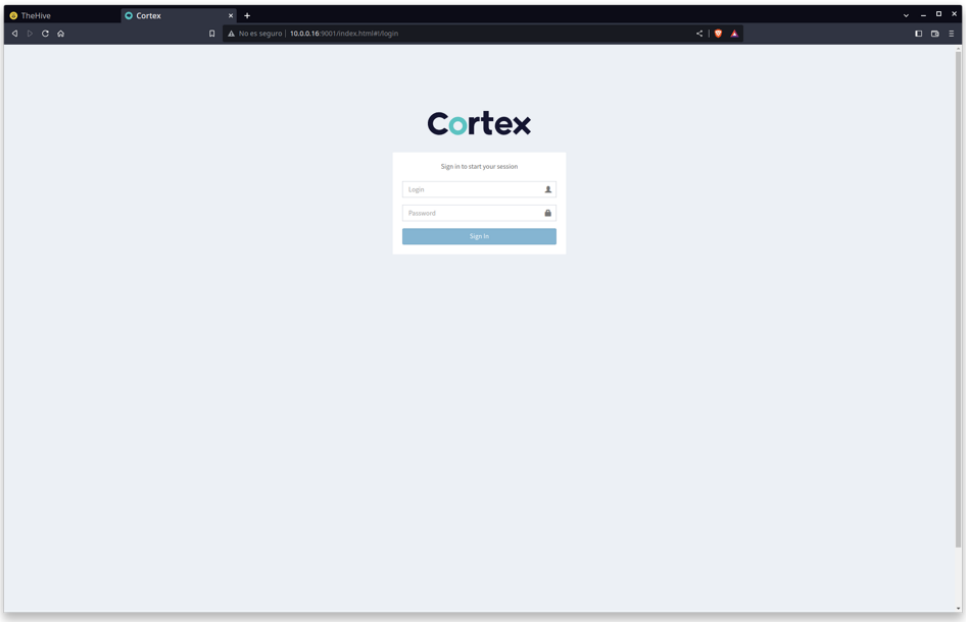

Ilustración 36 - Menú de Login de Cortex

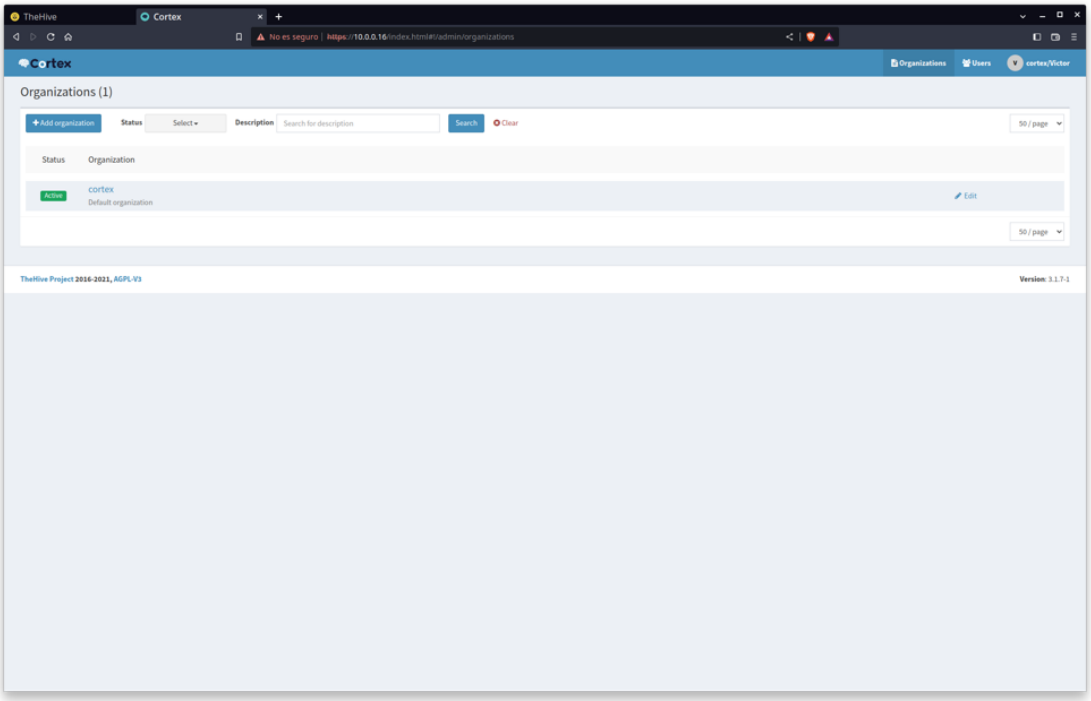

Ilustración 37 - Pantalla principal de Cortex

Al igual que hicimos con TheHive, para poder trabajar tenemos que crear una nueva organización, que será la misma.

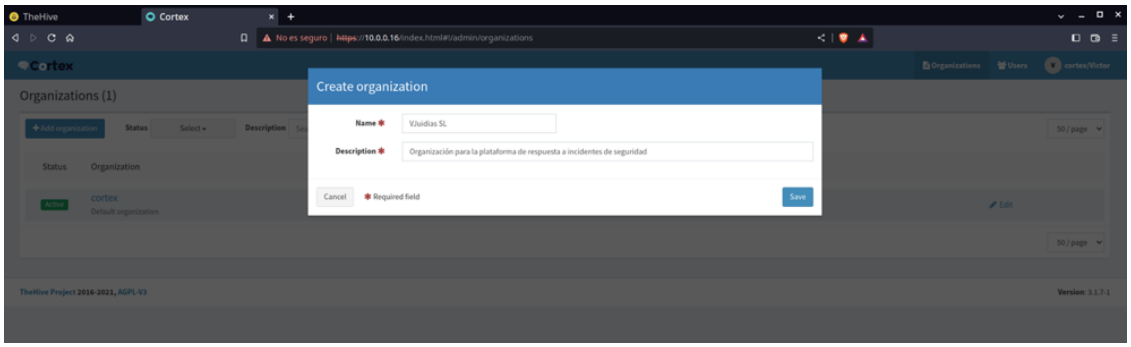

Ilustración 38 - Crear una nueva organización en Cortex

Una vez creada, tenemos que crear usuario para ella. Uno de ellos será un usuario con el rol de administrador y con el que trabajaremos en la interfaz de Cortex. También crearemos un segundo usuario, pero esta vez con el rol de “read” y “analyze”, para que nuestro servidor de TheHive se pueda conectar a Cortex [42].

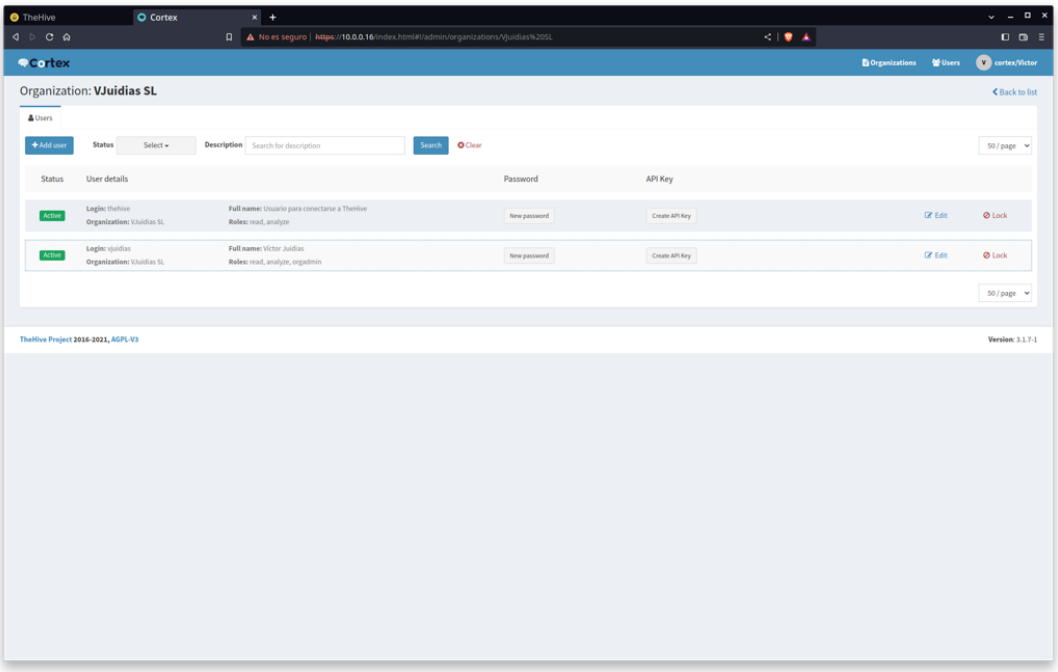

Ilustración 39 - Listado de usuarios de la organización

En este caso también habrá que generale una API Key para que la conexión se haga sin contraseña. Es la misma API Key que añadimos a la configuración de la conexión de Cortex en TheHive.

Una vez tenemos los dos usuarios, cerramos sesión como el usuario administrador y entramos como el usuario administrador de la organización que creamos.

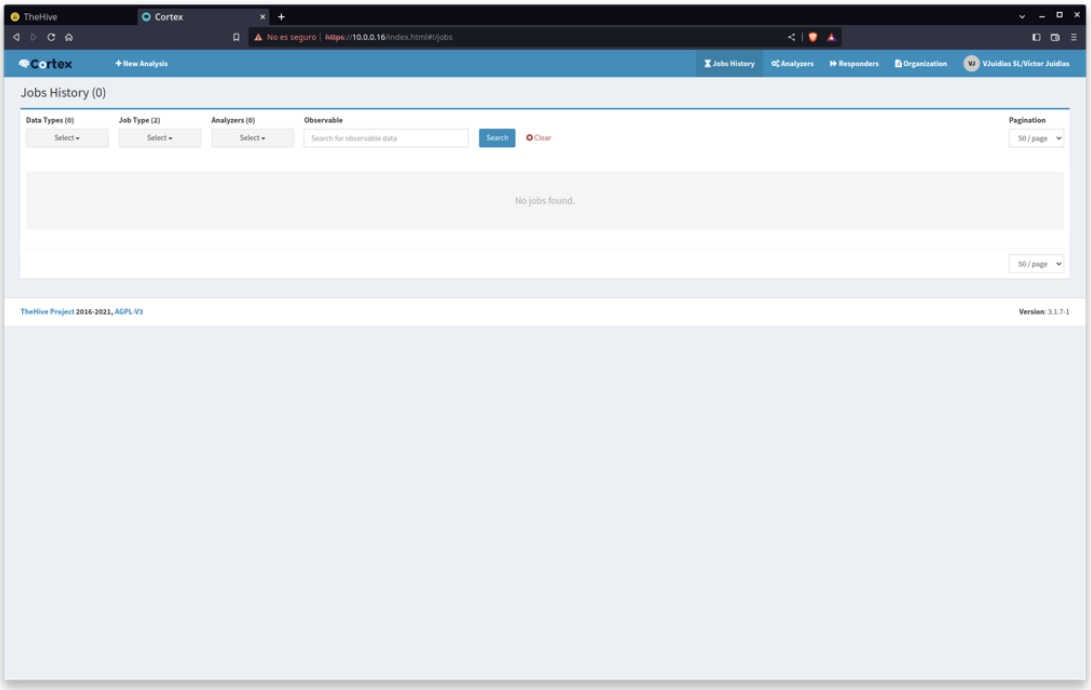

Ilustración 40 - Pantalla principal del administrador de la organización

La pantalla que se nos muestra es la pantalla de los “Jobs”, de los que hay dos tipos, los “analyzers” y los “responders”. Los analyzers pueden ser ejecutados contra los observables para obtener una información más detallada, mientras que los responders se utilizan con casos, tareas, observables, alertas, para ejecutar ciertas acciones durante la investigación de un incidente [43]. Para cada uno de ellos tenemos una larga lista de servicios que tienen que ser configurados debidamente para poder utilizarlos. Algunos de ellos son gratuitos, previo registro, otros son de pago [44]. Como es de suponer, los gratuitos tienen limitaciones como, por ejemplo, el número de peticiones que se pueden realizar al día. Cada uno de ellos tendrá que ser configurado debidamente para que funcione. En la mayoría de los casos, con introducir la API Key correspondiente será suficiente. Cuántas más fuentes de inteligencia tengamos disponibles, de mejor calidad y más fiable será la información que podremos obtener de los observables de nuestros casos.

Para acceder a la lista de analyzers y responders, tenemos que ir “Organization”. Una vez ahí, podremos ver la lista de usuarios disponibles en la organización y acceder también a la parte donde configuramos y habilitamos los analyzer y responders. Como se puede observar en la imagen, tenemos disponibles 106 configuraciones, puesto que varios analyzers pueden compartir a configuración.

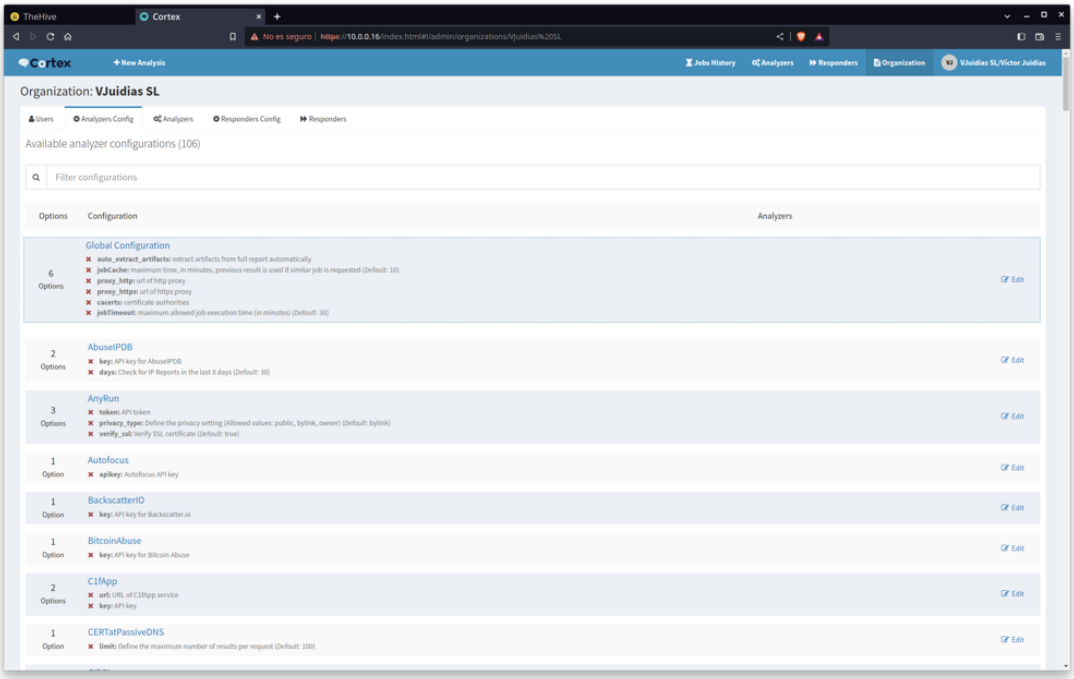

Ilustración 41 - Listado de configuraciones disponibles

Antes de habilitar un analyzer tenemos que editar la configuración de este.

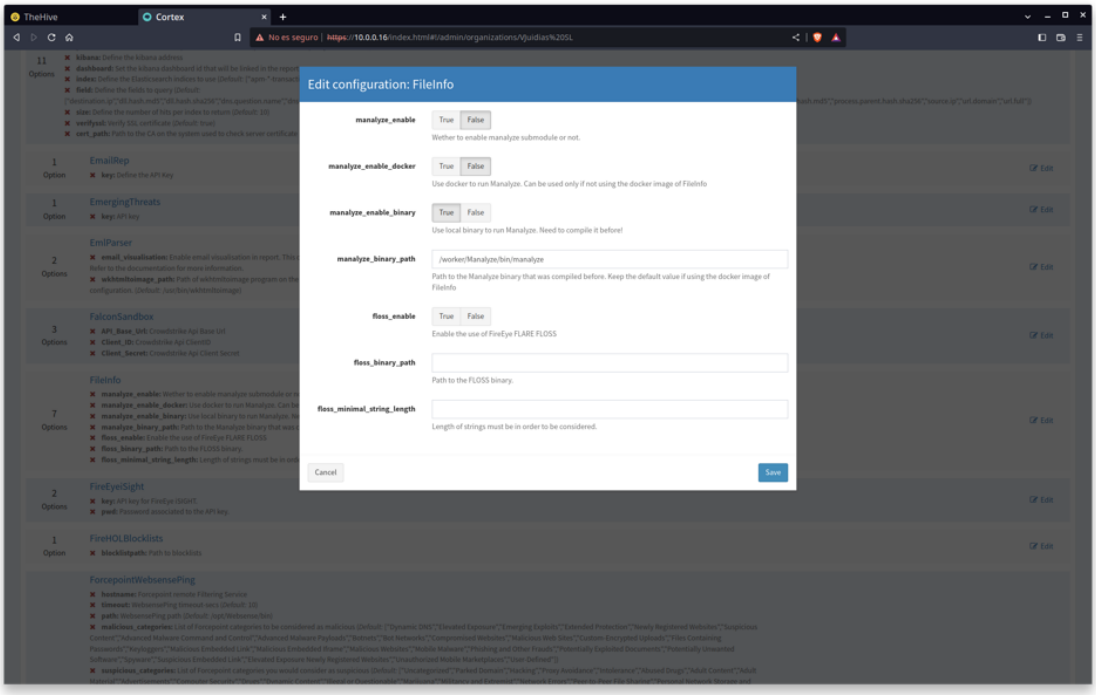

Ilustración 42 - Configuración de un analyzer

Este analyzer, por ejemplo, es uno que viene incluido dentro de Cortex, no tenemos que recurrir a ningún servicio externo. Simplemente editamos los campos como deseemos y guardamos la configuración, para ya poder habilitarlo en la siguiente pestaña. Entre los distintos analyzer disponibles también podemos configurar nuestro servidor MISP. De esta manera podemos hacer la conexión entre ambos servicios.

Una vez configurados los analyzer que queramos, podemos ir a la lista de todos ellos para habilitarlos.

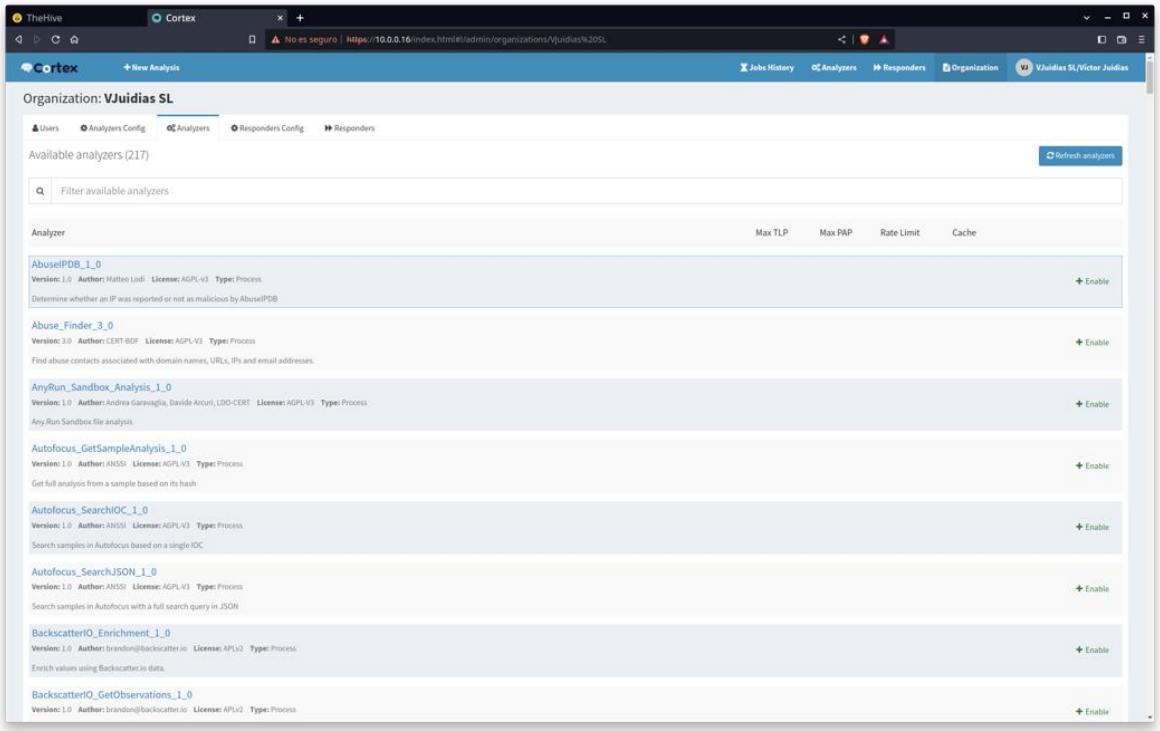

Ilustración 43 - Listado de analizadores para habilitar

Como se puede observar, tenemos disponibles 217 analyzers, que utilizar algunas de las 106 configuraciones anteriores.

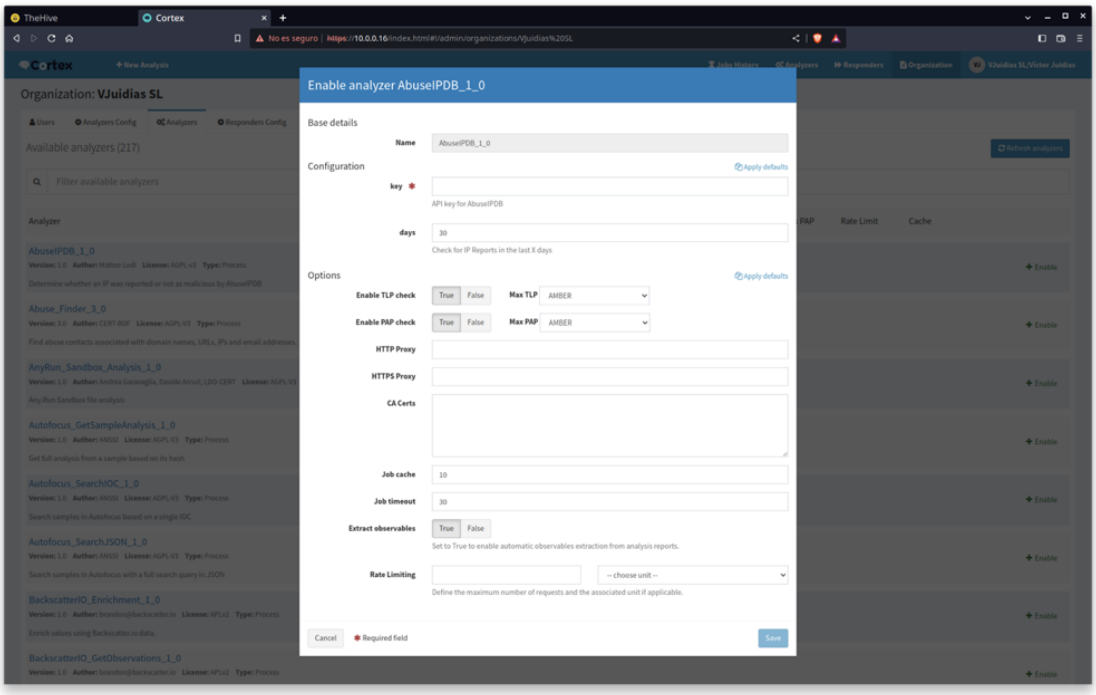

Ilustración 44 - Analizador de AbuseIPDB

En este ejemplo vemos el analyzer correspondiente al servicio “AbuseIPDB”, un proyecto dedicado a recopilar direcciones IP que se han relacionado con actividades maliciosas. En este caso sí que necesitamos una API Key para poder utilizarla, aunque para la captura de la imagen, la haya borrado.

En el momento de redactar el presente documento, tengo habilitados los siguientes analyzers:

- AbuseIPDB
- FileInfo_8_0
- MISP_2_1
- Shodan_DNSResolve_1_0
- Shodan_Host_1_0
- Shodan_Host_History_1_0
- Shodan_InfoDomain_1_0
- Shodan_ReverseDNS_1_0
- Shodan_Search_2_0
- TalosReputation_1_0
- ThreatResponse_1_0
- URLhaus_2_0
- Urlscan_io_Scan_0_1_0
- VirusTotal_DownloadSample_3_1
- VirusTotal_GetReport_3_1
- VirusTotal_Rescan_3_1
- VirusTotal_Scan_3_1

Los analizadores se especializan cada uno en temas diferentes, como puede ser la reputación de una IP, nombres DNS, investigar URLs, analizar ficheros, etc. También tenemos la posibilidad de crear nuestros propios analizadores en Python.

De la misma manera que configuramos y habilitamos los analizadores, con los responders hacemos lo mismo, tenemos que configurarlos primero y luego habilitarlos.

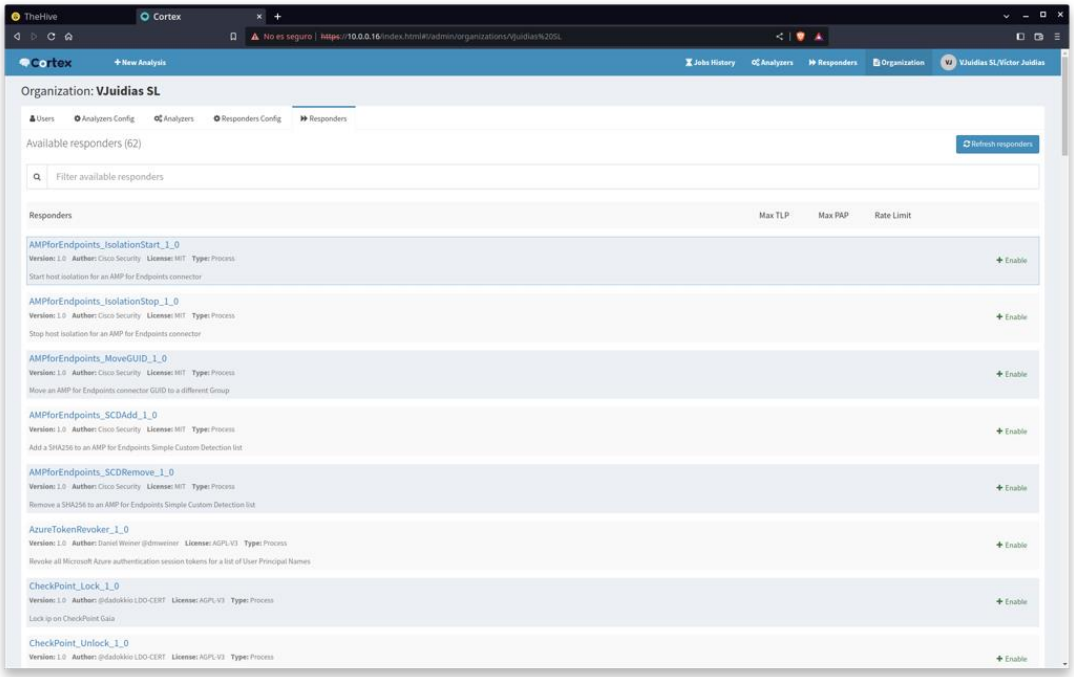

Ilustración 45 - Listado de responders disponibles

Como se puede observar, tenemos muchos menos responders que analyzers. En total tenemos disponibles 62 responders que utilizan unas 28 configuraciones. También es normal que pueda haber menos responders que analyzer, puesto que éstos son más proactivos que los otros. Mientras los analyzer analizan un observable para obtener más información, los responders están pensados para realizar acciones sobre casos, tareas, alertas, etc. Es decir, necesita realizar acciones en otra herramienta, por tanto, debe ser capaz de soportar otros productos.

Si exploramos la lista vemos que tenemos disponibles responders para interacturar con firewall PaloAlto, con Redmine, con Microsoft Defender, con Gmai, con Azure, etc. Por tanto, dependerá de la infraestructura de la que se disponga, podremos tener más o menos responders.

De entre todos los servicios disponibles, se han habilitado los siguientes responders:

- Virustotal_Downloader_0_1
- Wazuh_1_0

El primero es porque ya teníamos la API Key para utilizar los analizadores de VirusTotal. Este responder nos permite descargarnos desde VirusTotal una muestra de malware enviando un hash. El segundo es nuestro servidor de Wazuh. Según la documentación oficial, nos permite bloquear un IP de un observable a través de Wazuh. Para esto hemos creado un nuevo usuario en Wazuh. Para comprobar que funcione, le hemos asignado el rol de “agent_admin”, visto que tiene que realizar ciertas acciones sobre los agentes, aunque lo ideal sería crear un rol específico.

En este caso no generamos ninguna API Key, sino que Cortex nos pide el nombre de usuario y la contraseña.

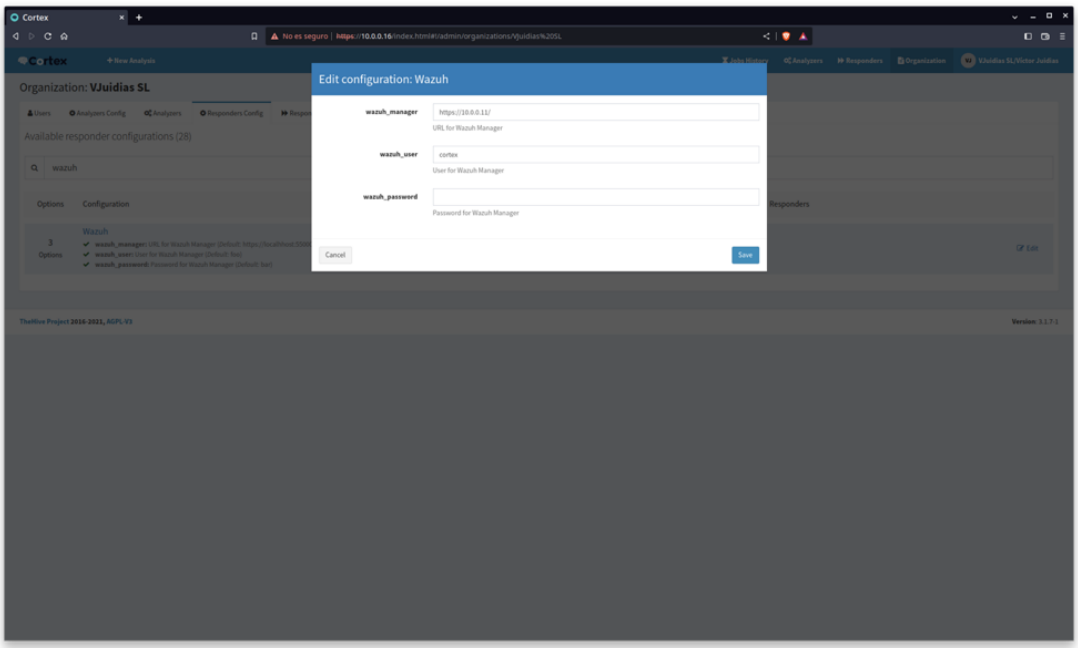

Ilustración 46 - Habilitar el responder de Wazuh

4.4. MISP

Seguimos la documentación oficial para instalar la herramienta en una máquina virtual, pero esta vez no en VirtualBox sino en el Proxmox. Al finalizar la instalación podemos acceder a la interfaz web con el usuario admin@admin.test, que es el usuario que nos proporciona MISP tras la instalación.

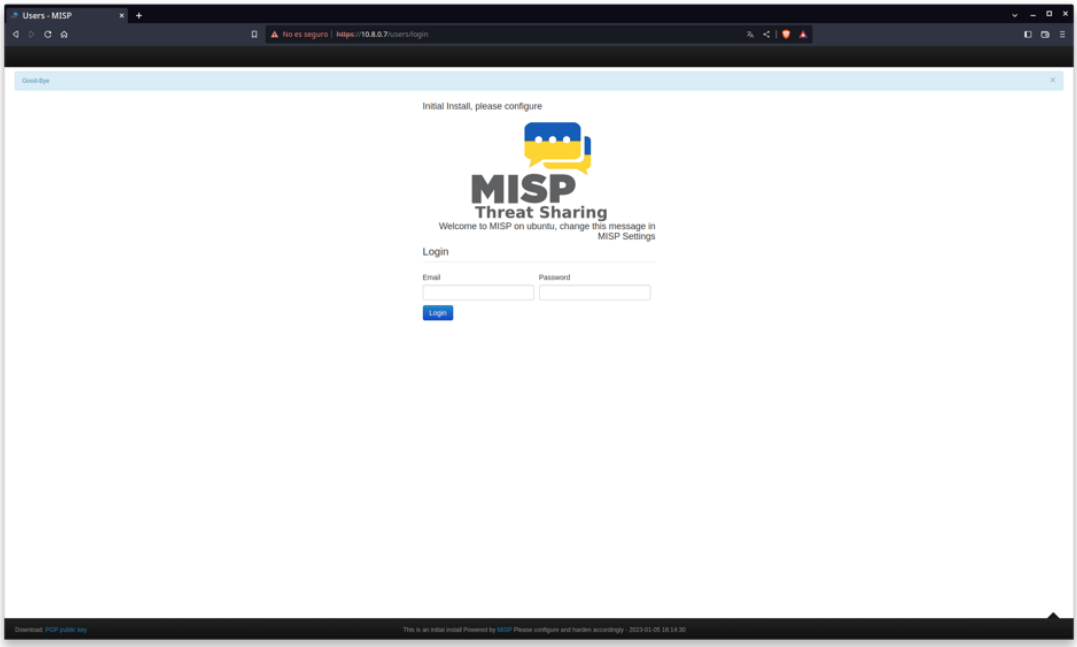

Ilustración 47 - Pantalla de login de MISP

Como se puede observar, la herramienta viene configurada con un certificado autofirmado para la conexión SSL. Al iniciar sesión se nos pide que cambiemos la contraseña por defecto.

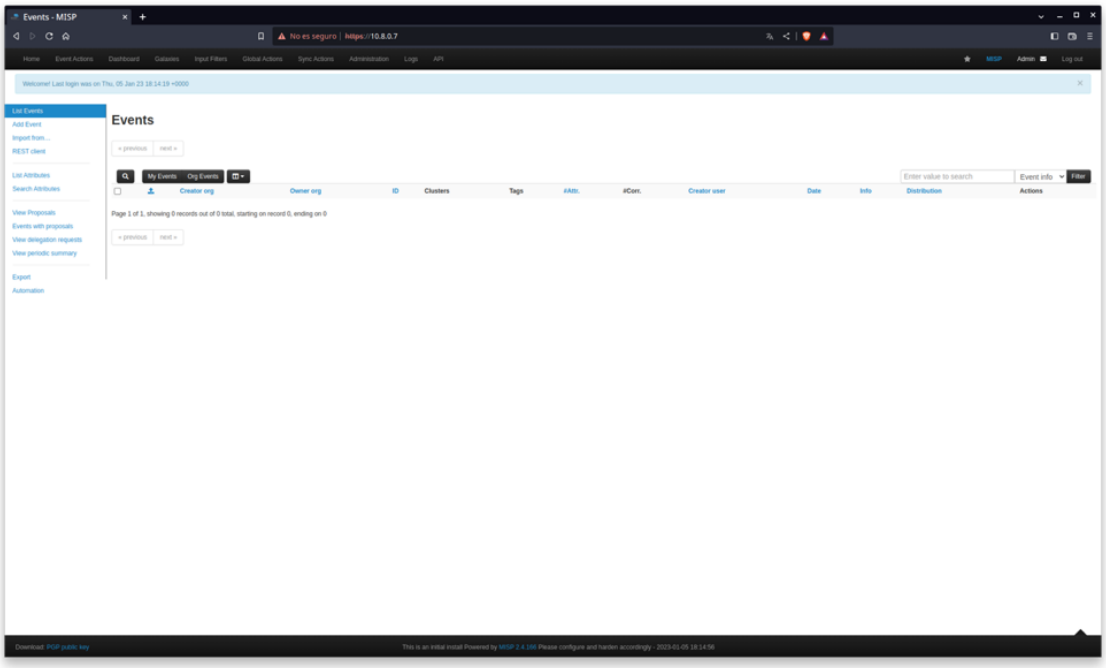

Ilustración 48 - Listado de Eventos

Al igual que hemos hecho en el resto de las herramientas, tenemos que crear una nueva organización y usuarios para la misma. Para hacerlo tenemos que ir al menú de “Administration”, “Add Organizations”. Cumplimentamos el formulario que se nos presenta para añadir nuestra organización ficticia a MISP.

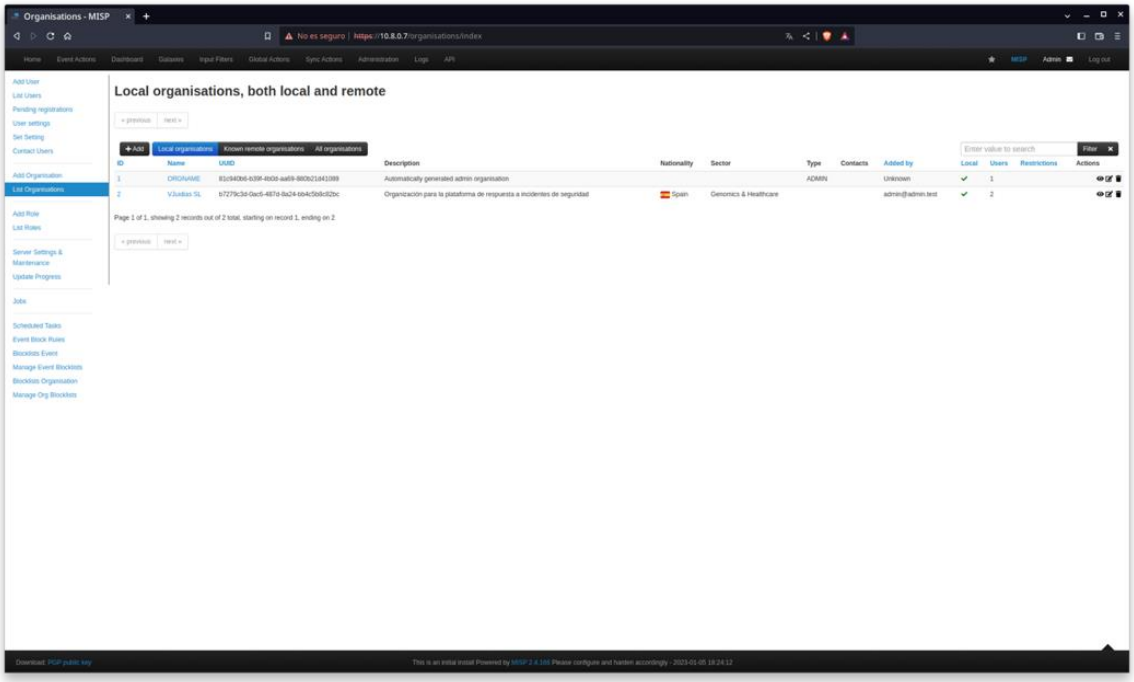

Ilustración 49 - Nueva organización en MISP

Tras la creación de la organización, creamos los usuarios para la misma. Al igual que en los casos anteriores, vamos a crear dos usuarios, uno con el rol de administrador y otro para Cortex, con permisos de solo lectura. Como los usuarios tienen que ser con correo electrónico, seguimos el mismo criterio para el dominio que con el resto de las herramientas. Para el usuario de Cortex, le creamos una API Key.

Una vez creados los usuarios, iniciamos sesión con el usuario administrador. Como no hemos hecho nada aún no tendremos ningún contenido en la aplicación, como podemos ver en nuestra lista de “Feeds”.

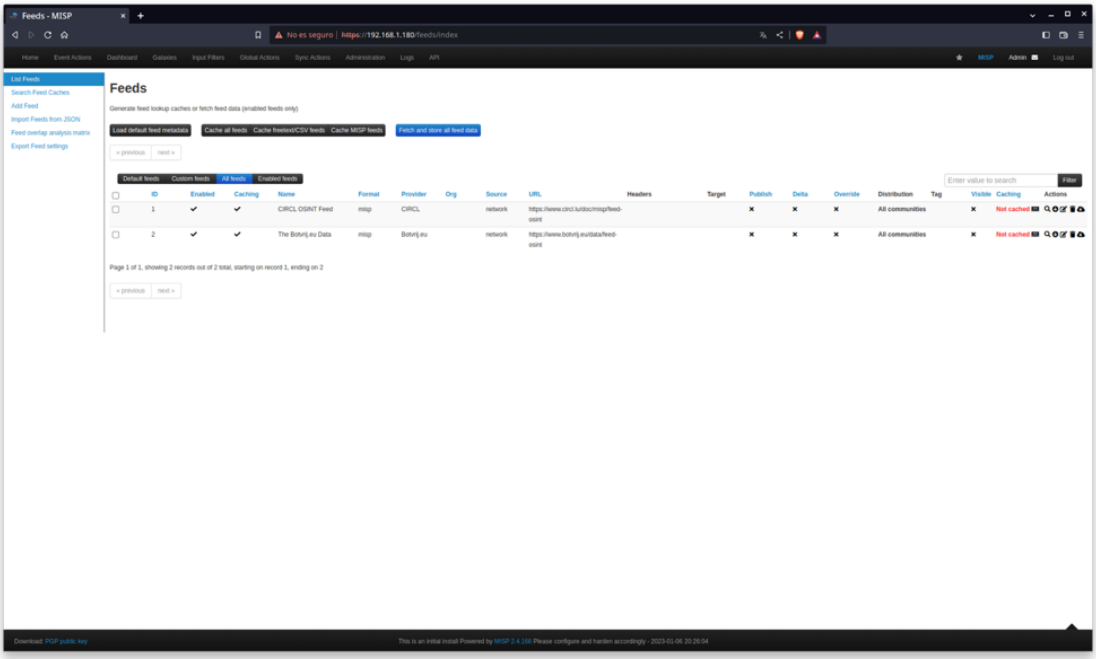

Ilustración 50 - Listado de Feeds

Por defecto solo tenemos estos dos, que vienen con la instalación de MIPS. Por defecto viene desactivados, así que hay que habilitarlos para poder utilizarlos. A través de los iconos de la derecha podremos, entre otras cosas, editar los “Feeds”, donde podremos habilitarlos y habilitar también el caché de su información.

Si necesitamos más Feeds, tenemos disponible la página de MISP Default Feeds [45], donde encontraremos una larga lista de Feeds que podremos añadir a nuestra herramienta. Simplemente tenemos que añadir la URL que aparece en la lista. A parte de estos Feeds, deberíamos tener también tener información de organizaciones u organismos oficiales de nuestro entorno o país.

Una vez los estén habilitados, podremos descárgalos. Para hacerlo, de entre los iconos de la derecha tenemos uno que dice “Fetch all events”. Pero lo ideal sería que se actualizara regularmente la información. Para ello, bajo la pestaña “Administration” tenemos “Scheduled Tasks”.

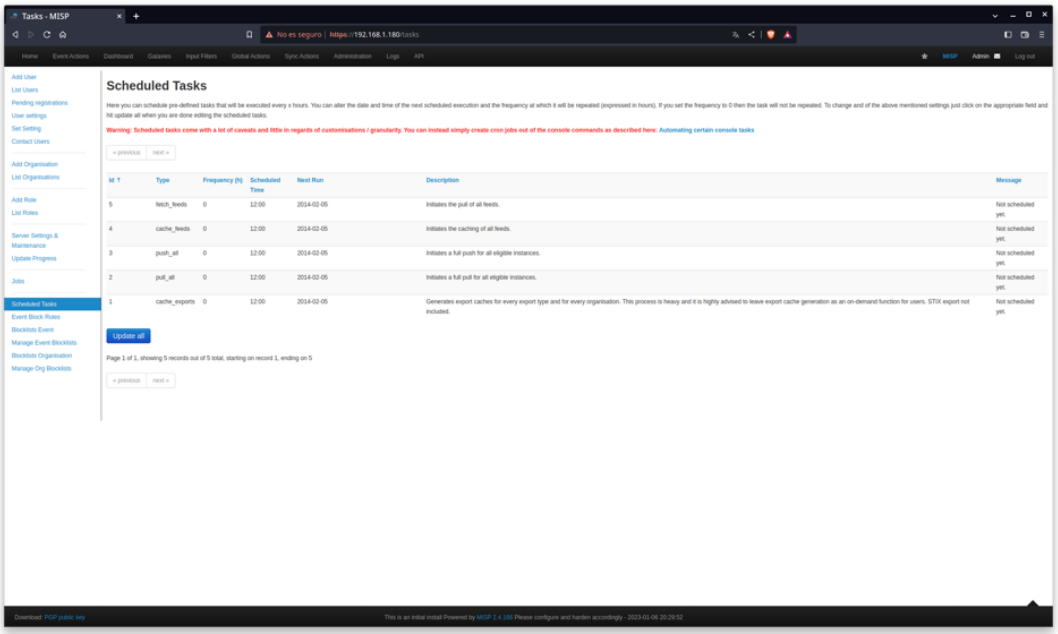

Ilustración 51 - Descargar los Feeds

Desde aquí, podremos gestionar como queremos que se haga la actualización de la información. Como se puede observar, la fecha de la próxima descarga está algo anticuada, pues está establecida para el 5 de febrero de 2014. Para modificar esta configuración, simplemente clickamos en los campos que queramos, como la hora para la descarga o cada cuanto tiempo se debe descargar información nueva de los feeds. Para el propósito que nos ocupa, actualizaremos el campo de “Next Run” a la fecha actual para tener información con la que trabajar. Si vamos a la pestaña “Jobs” que está a la izquierda, podremos ver el estado de las descargas.

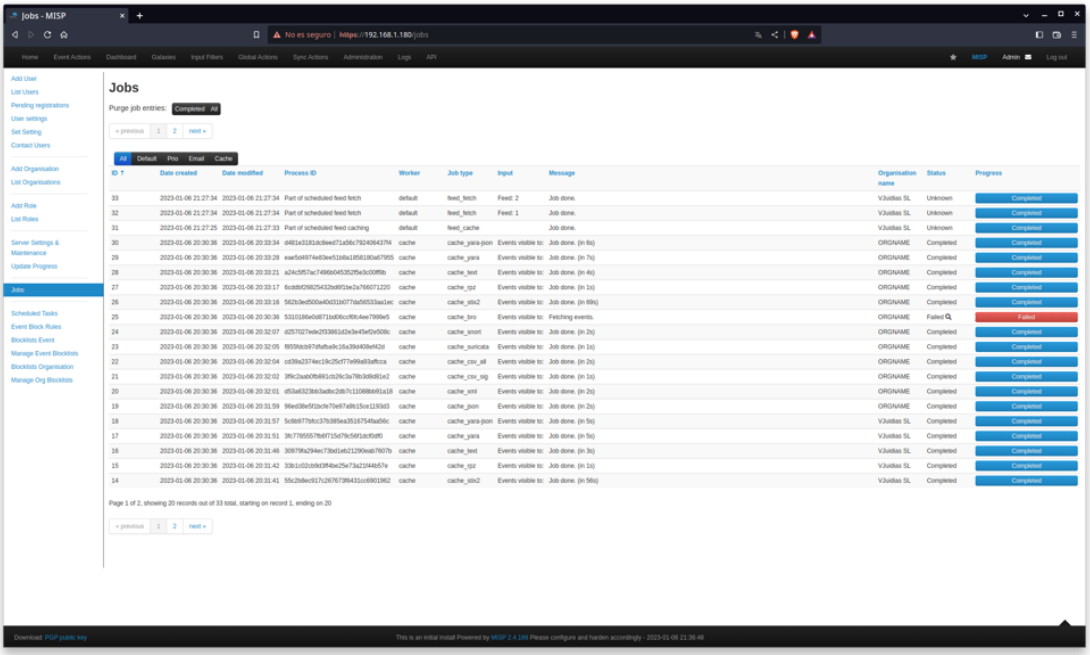

Ilustración 52 - Listado de trabajos

Si vamos a la pestaña “Events Actions”, “List Events”, podemos ver todos los eventos que se ha descargado y con los que podemos trabajar.

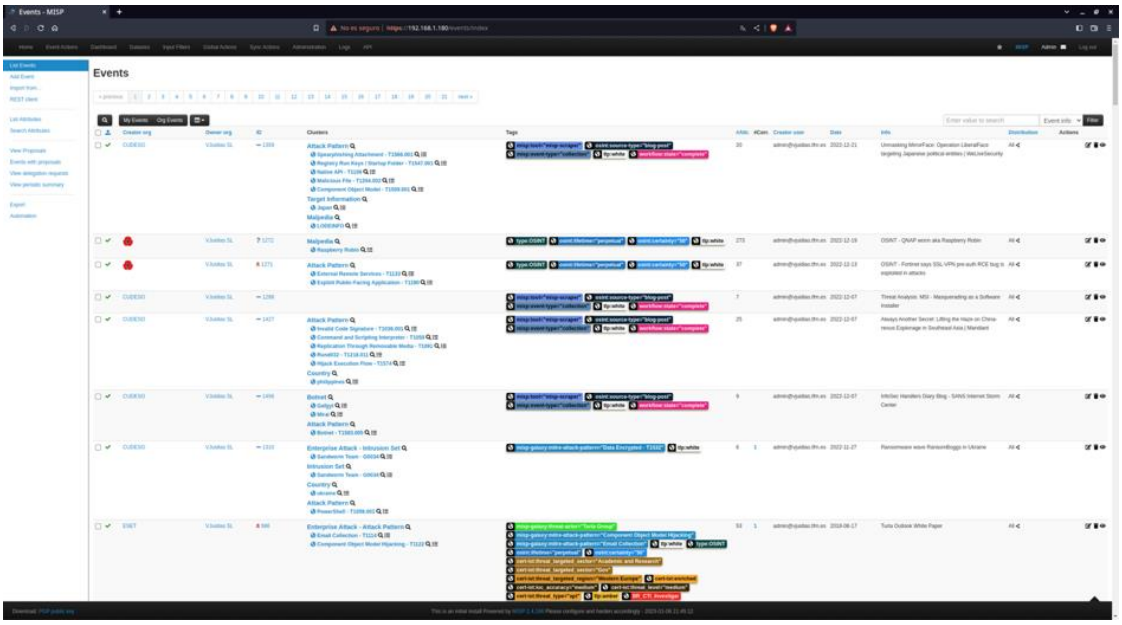

Ilustración 53 - Listado de eventos actualizados

Como podemos ver, los eventos están etiquetados, asignadas por terceros. Cabe mencionar que, al ser MISP una herramienta de compartición de información, nosotros podemos también publicar nuestros propios eventos y utilizar tanto las etiquetas ya disponibles como crear nuevas.

4.5. Integración Wazuh - TheHive

Ahora tenemos casi todos los elementos de nuestra plataforma de respuesta instalados, configurados y listos para ser utilizados, lo único que nos falta es integrar nuestro sistema de alertas, Wazuh, con TheHive, nuestra plataforma de gestión de casos de incidentes de seguridad. Para ello, la propia gente de Wazuh a través de su blog, tiene publicado una guía sobre como integrar ambas herramientas [46]. La integración pasa por realizar lo siguiente:

- En TheHive, crear un nuevo usuario para nuestra organización para Wazuh, con el rol de analista, al que le generaremos una API Key.
- En la máquina de Wazuh, instalar el módulo de Python para TheHive.
- Crear un script de integración, copiando el que se nos proporciona desde la documentación. Aquí podemos modificar, a través de una variable delscript, el nivel de alerta mínimo para que se envíen las alertas a TheHive.
- En Wazuh, crear un script en bash, también proporcionado por la documentación, que será el que ejecute a su vez, el script anterior de Python.
- Asignar los permisos adecuados a ambos scripts
- Añadir al fichero de configuración de Wazuh, la integración, especificando el nombre del scritp de bash, la URL de TheHive y la API Key para conectarse.

- Reiniciar el servicio para que funcione.

Si todo ha ido bien, podremos ver como las alertas de Wazuh se transforman a su vez en alertas en TheHive. Para probarlo, podemos probar a conectarnos a nuestra máquina de Pfsense por ssh introduciendo datos erróneos para forzar una alerta. Para ello, primero tenemos que habilitar el servicio en Pfsense, ya que entiendo que por seguridad viene deshabilitado por defecto.
Probamos varias veces a conectarnos con el usuario “root” e introducimos una contraseña errónea varias veces hasta que nos echa fuera. Comprobamos que efectivamente en Wazuh, en “Security Events”, vemos las alertas por “authentication_failed” e “invalid_login”, entre otras. El problema viene que en TheHive no vemos que se estén creando nuevas alertas como se debería estar haciendo si hemos realizado bien la integración entre ambos servicios. Revisamos que hemos hecho las cosas bien leyendo de nuevo la guía y comprobando que esté todo como debería estar, pero nada, seguimos forzando alertas, pero no vemos que eso se traduzca también en alertas en TheHive. Revisamos los logs de Wazuh y vemos que hay un fichero de log específico para las integraciones, “integrations.log” y vemos lo siguiente:

Ilustración 54 - Error SSL entre Wazuh y TheHive

Como hemos utilizado un certificado autofirmado, no es válido y Wazuh no puede hacer la petición a TheHive para que cree las alertas correspondientes. Pero esto tiene fácil solución, porque estamos trabajando en un entorno de pruebas, en un entorno real deberíamos utilizar un certificado válido.

Si no fijamos, el error aparece en al ejecutar la línea 909 del fichero “api.py” que está incluido dentro del módulo de Python de TheHive. En esa línea de ejecuta un “request” hacia la URL de TheHive y tiene específicado el parámetro “verify” para el certificado. Si cambiamos “verify=self.cert” por “verify=False”, evitará comprobar la autenticidad del certificado. Son varias líneas que hay que modificar, pero al hacerlo y forzar de nuevo las alertas, ahora sí, vemos como se crean las alertas correspondientes en TheHive.

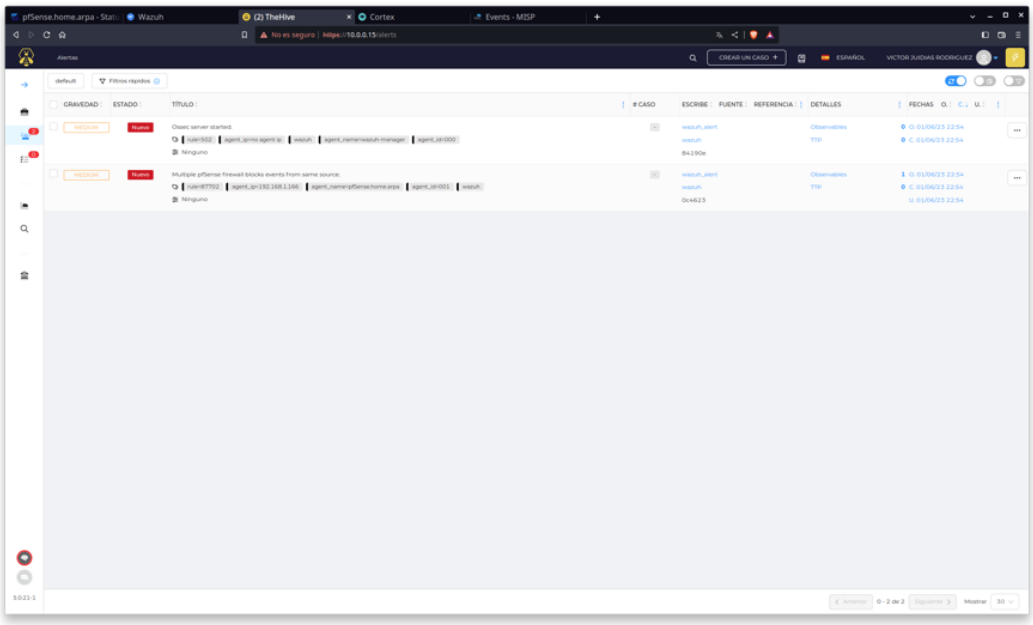

Ilustración 55 - Alertas creadas en TheHive por Wazuh

Podemos también ver los observables de la alerta, que en este caso es la IP desde la que se intentó acceder por ssh a Pfsense.

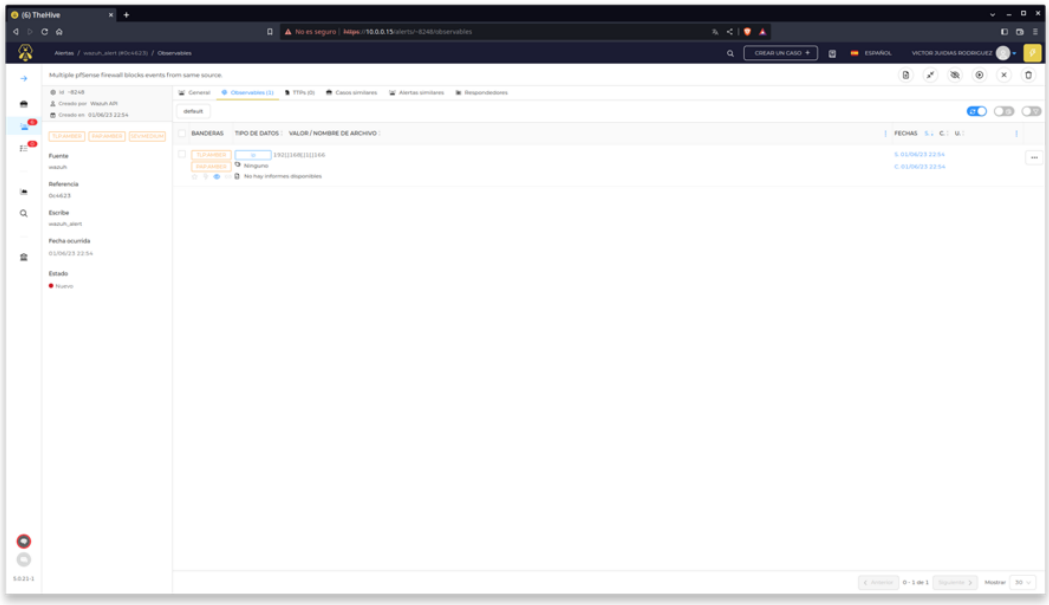

Ilustración 56 - Observables de la alerta creada por Wazuh

Con esto, ya tendríamos integrado Wazuh con TheHive y a su vez, TheHive con Cortex y MISP.

4.6. n8n

La herramienta tiene varios modos de instalación, se puede utilizar a través de Docker o instalándola vía “npm” el gestor de paquetes de NodeJS. También tienen documentado la instalación para plataformas específicas como Digual Ocean, Heroku o utilizando kubernetes en AWS, Azure o Google cloud. En nuestro caso, se instalará en una máquina virtual. Tenemos dos opciones,

podemos ejecutar n8n sin necesidad de instalar nada, utilizando “npx”, o instalarlo globalmente utilizando “npm” [47]. En este caso, con “npx” nos es más que suficiente para poner en marcha el servidor y probar las funcionalidades que ofrece n8n e integrarlo en el workflow.

Una vez instalado, podemos empezar a utilizar n8n a través de la interfaz web:

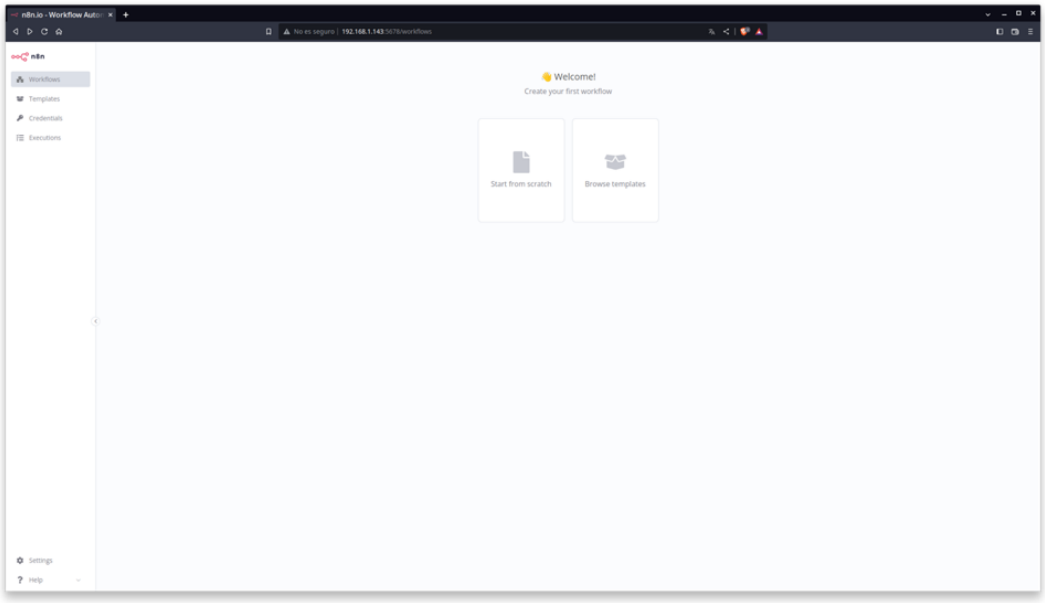

Ilustración 57 - Pantalla principal de n8n

En este caso no he configurado una conexión SSL porque he visto en los foros que había bastante gente con problemas para configurarlo adecuadamente y que funcionase, sobretodo a la hora de usar “webhooks”. Para el caso que nos ocupa, he preferido dejarlo sin SSL para probar la herramienta e integrarla con las demás, pero sobra decir que, en un entorno real, hay que utilizar siempre conexiones cifradas.

Como se puede ver, no tenemos todavía ningún workflow, así que, o podemos crearlo de cero o importar una plantilla y trabajar a partir de ésta. En nuestro caso, n8n tiene integraciones para trabajar con TheHive y Cortex. En su página oficial podemos revisar la lista completa de integraciones que soporta actualmente n8n.

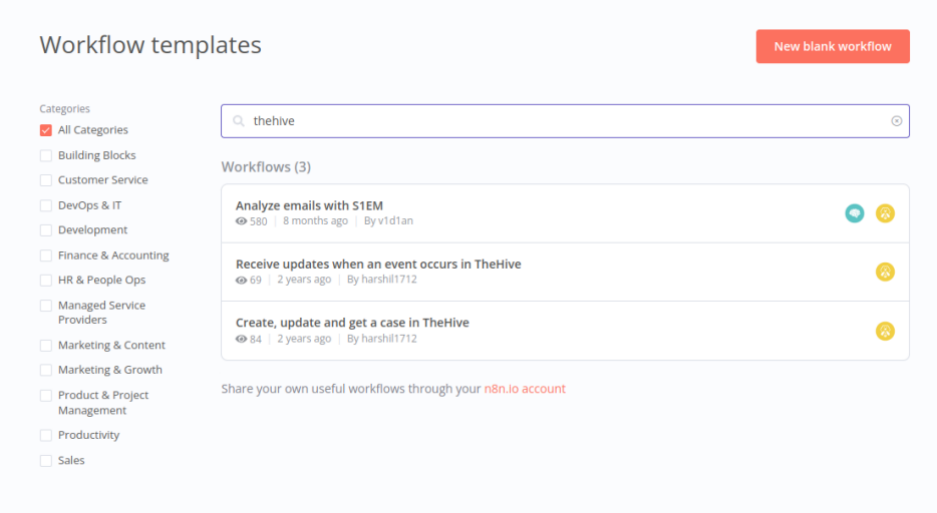

Ilustración 58 - Plantillas para n8n

Vemos que tenemos a nuestra disposición varias plantillas para TheHive en la que una de ellas también utiliza Cortex. Podemos visualizar cada una de estas plantillas antes de importarlas.

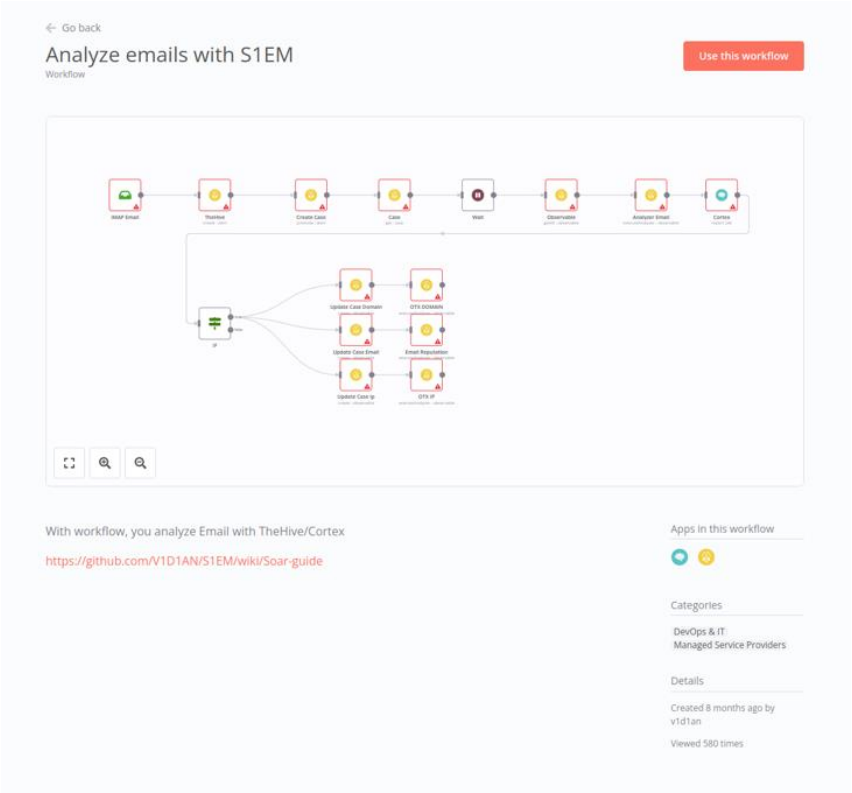

Ilustración 59 - Workflow de TheHive en n8n

En este caso, este workflow analiza una dirección de correo electrónico sospechosa. Como vemos, va pasando por los diferentes nodos del workflow que interactúan con TheHive, a partir de un email reportado, crea una alerta, que a su vez eleva a un caso, para luego pasarle los observables a Cortex para que los analice y según los resultados del análisis se actualizará el caso debidamente.

Esto es solo un ejemplo de lo que podemos llegar a hacer utilizando esta herramienta. Como se puede ver, tenemos nodos condicionales, los clásicos “If” de cualquier lenguaje, que, en base a unos datos de entrada procedentes del nodo anterior, podemos derivar el flujo del workflow a otros nodos. En resumen, las posibilidades son inmensas, así que dependerá de nuestras necesidades, el crear un workflow más o menos complicado.

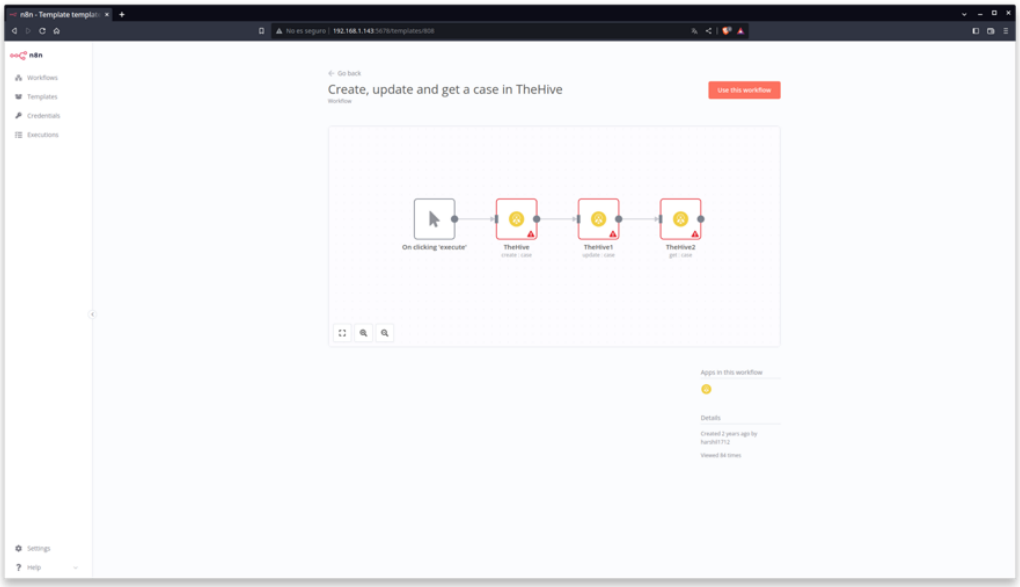

Ilustración 60 - Workflow que crea y actualiza un caso en TheHive

Los nodos los podemos ejecutar por separado para comprobar su funcionamiento y ver, entre otras cosas, la salida que generan para saber con qué tenemos que trabajar en el siguiente nodo

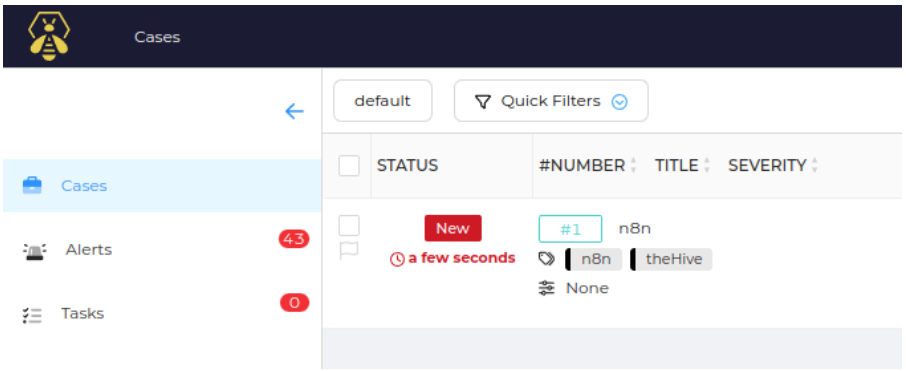

Ilustración 61 - Caso creado a partir de n8n

En este caso, hemos visto un nodo que es capaz de interactuar con TheHive, a través de su API, para gestionar alertas, casos, observables, etc. Para que esto funcione, debemos previamente haber creado un usuario de tipo analista en nuestra organización y haberle creado una API Key que importaremos a n8n.

Pero también tenemos otro tipo de nodo para trabajar, el llamado “trigger”. En este caso, es un webhooks que se quedará a la espera de que sea llamado para empezar el flujo de trabajo. Para esto, en TheHive tenemos que configurar dos cosas, un “Endpoint” y una notificación. Las notificaciones se ejecutarán

cuando un determinado evento ocurra, como una alerta o un caso que se ha creado, un caso que se actualiza, etc.

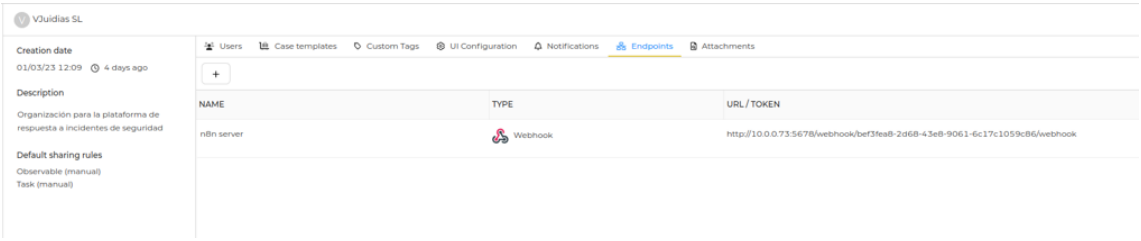

Ilustración 62 - Crear un webhook en TheHive

En nuestro caso, para probar la funcionalidad de la herramienta y el potencial de la integración entre ambas herramientas, podemos crear una notificación cuando una alerta nueva se cree, en este caso proveniente de Wazuh. Dicha notificación, hará una llamada a un webhook, que será nuestro trigger en n8n y a partir de aquí, podemos crear un caso a partir de dicha alerta e incluso probar a analizar algún observable con Cortex.

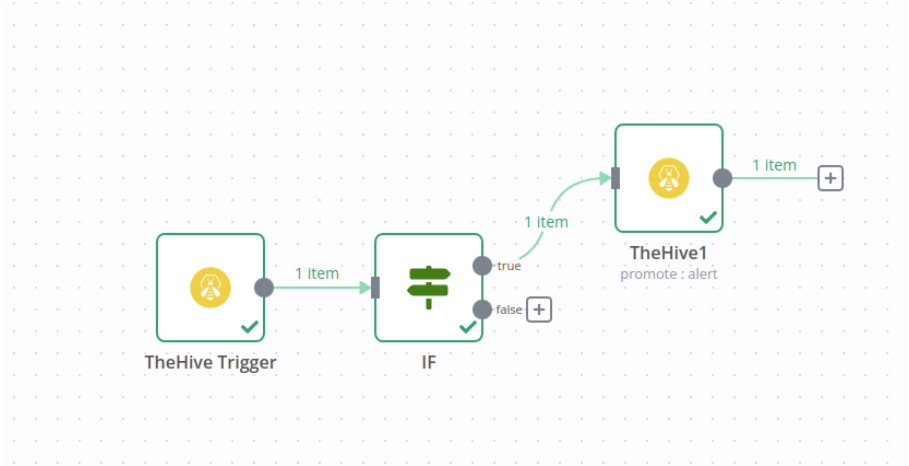

Ilustración 63 - Workflow usando un trigger

En este ejemplo, a partir de una alerta generada en TheHive, ejecutamos el webhook en n8n. A partir de aquí, filtramos el contenido de la alerta para procesar solo las que reportan un fallo en la autenticación por ssh, ya que es algo fácil de generar. Si la alerta es cumple dicha condición, creamos un caso a partir de la información de la alerta.

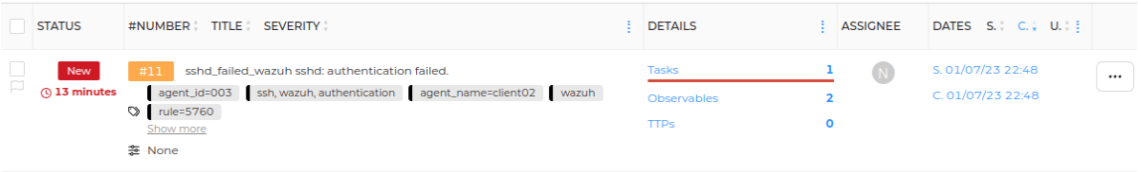

Ilustración 64 - Caso creado a partir de un trigger en n8n

Como hemos dicho, esto es solo una muestra del potencial que nos ofrece n8n para automatizar tareas. Para el caso que nos ocupa, podríamos crear un workflow que, al igual que en el ejemplo, crease casos automáticamente a partir de una alerta, extrajese los observables, los analizase con Cortex, pudiendo realizar otro tipo de procesamiento también, y a partir de la información generada

por Cortex, actualizar el caso debidamente, ya sea, por ejemplo, dándolo por cerrado o asignándoselo a algún analista. Como vemos, podríamos automatizar todas las alertas y casos que tengamos para dedicar tiempo a los casos más complejos, o que de especial gravedad.

4.7. Monitorización

Para implementar la plataforma de respuesta incidentes de seguridad he utilizado máquinas virtuales que se ejecutan tanto en mi ordenador como en un Intel NUC. Además, he instanciado algunas máquinas con recursos ligeramente por debajo a lo que especificaban en la documentación oficial de dichas herramientas. Por ello, era importante tener algún tipo de monitorización más allá de los clásicos comandos de Linux, algo que pueda visualizar el consumo de los recursos en el tiempo. Para este caso he optado por el tándem Prometheus/Grafana. Por un lado, Prometheus, a través de sus agentes llamados exportes e instalados en las máquinas clientes, se encarga de recopilar, almacenar y organizar las métricas de cpu, memoria, red, disco, etc. Y, por otro lado, con Grafana podemos visualizar dicha información a través de los paneles o “dasboard” que tenemos a disposición.

Ilustración 65 – Monitorización de las VMS

En este caso tenemos a la máquina de TheHive, la cual está dimensionada con menos memoria ram de lo que requieren, 12GB frente a 16GB. Como el caso que nos ocupa es plantear una plataforma de respuesta a incidentes desde un punto de vista didáctico o como prueba de concept, el nivel de usuarios, peticiones e información iba a ser bastante bajo, razón por la cual me tomé la libertad de reducir la cantidad de memoria ram, algo que también hice con la máquina de Cortex y MISP. Como podemos ver, llega a consumir unos 9GB-10GB de forma sostenida, que es lo que he ido observando. La razón se debe al uso de Elasticsearch y Java, que tienen unos requerimientos de

memoria alto. Pero en cuanto a CPU, el consumo no ha sido tan alto. Con la máquina de Cortex el comportamiento es similar.

Durante el trabajo de puesta en funcionamiento de las distintas herramientas, aunque he bajado los recursos de algunas máquinas, no he notado ningún tipo de problema, ni ralentización ni nada, por lo que, al final, no ha sido mala idea optimizar los recursos disponibles. Es posible que hubiera podido ajustar más la cantidad de memoria, pero lo veía innecesario. Está bien saber que las herramientas pueden llegar a funcionar un poco por debajo de estos requerimientos, si alguna vez, en su puesta en producción, no se pueden cumplir los requisitos mínimos.

5. Caso de uso

Para una plataforma de estas características hay multitud de casos de uso, puesto que la seguridad informática abarca muchos ámbitos de la infraestructura de una organización. Para este trabajo nos centraremos en uno que involucre todos los componentes de nuestro entorno de simulación.

- Eventos de seguridad detectados por los agentes de Wazuh y enviados al servidor de Wazuh para su almacenamiento y su procesamiento posterior.
- Una vez Wazuh detecte el evento de seguridad, gracias a su integración con TheHive, cree una nueva alerta en esta plataforma con toda la información disponible, como tipo de evento de seguridad, dirección IP si procede, máquina o entorno afectado, usuario afectado, etc.
- Una vez la alerta es creada en TheHive, la podemos tratar como un nuevo caso.
- Podemos enriquecer la información de los observables del caso (direcciones IP, nombres de dominio, ficheros, etc.) utilizando Cortex y sus analizadores. Esta información extra nos ayudará a determinar si la alerta es real, cuán grave puede ser o si es un falso positivo.
- En función de la información que ya tengamos en este punto, podemos decidir si utilizar los módulos de Cortex como los “responders” (Wazuh, por ejemplo) para dar respuesta a la incidencia o bien, utilizar otras herramientas para dar por finalizada la incidencia.
- En función de lo anterior, actualizamos el caso en TheHive, documentamos lo ocurrido y damos por cerrado el caso.
- Como elemento innovador, podemos incluir la herramienta de n8n en nuestra plataforma para que gestione automáticamente las alertas, casos, observables, etc., pudiendo gestionar una parte de las alertas, como las catalogadas de bajo riesgo, por ejemplo, de forma automática sin necesidad de tener que dedicar personal o tiempo a gestionarlas.
- De la misma forma, con n8n podemos crear otros workflows que nos ayuden a automatizar determinadas tares que nos ayuden a ser más ágiles en nuestro trabajado diario.
- Una vez se dé por cerrada la incidencia, podemos utilizar MISP para compartir los resultados obtenidos o si aún está abierta, consultar la información disponible para ver si encontramos algún evento que ya haya ocurrido que esté relacionado con el nuestro.

Con esta situación que hemos planteado, estaríamos utilizando todos los elementos de nuestra plataforma y no sería muy distinto de un caso al que se pudiera enfrentar un analista en una situación real en cualquier empresa u organización.

5.1. Test de un caso de uso

Una vez hemos validado que todas las herramientas funcionan como es debido y pueden trabajar unas con otras, podemos pasar a simular un caso de uso.

Como se ha explicado en apartados anteriores, hemos creado un entorno simulado de máquinas virtuales, que están detrás de una máquina Pfsense que hace de router/firewall, simulando una infraestructura básica que puede tener cualquier organización o empresa. Durante el presente trabajo, he ido probando las distintas funcionalidades de las diferentes herramientas utilizando las alertas generadas por Pfsense, ya sea por su firewall integrado o por Suricata, el IDS que instalamos. La información de ambos es recogida por el agente de Wazuh instalado en el propio Pfsense y envidad al servidor. Pero para el caso de uso he querido probar otro escenario. Para el caso de uso vamos a probar con intentos de acceso por ssh a una máquina que haga de cliente. Las alertas por intentos de acceso por ssh son fáciles de generar y, además, son muy habituales. La idea de hacerlo así es que normalmente los sistemas tipo Pfsense están muy bien configurados, son robustos y son seguros, y en la inmensa mayoría de los casos no tienen habilitado el servicio ssh. Pero un ordenador cualquiera, ya sea Linux o Windows o incluso macOS, no están tan bien securizado, puesto que su propósito es otro. El caso de uso planteado es el de una máquina cliente, una Ubuntu, instanciada en el proxmox del Intel NUC y que se conecta al Pfsense a través de una vpn. La idea es replicar una situación real. Hoy en día, el trabajo remoto está, o parece que está, cada vez más implantado y en donde los trabajadores normalmente se suelen conectar a las redes empresariales utilizando una vpn. Muchas veces utilizan equipos cedidos por la propia empresa y otras, ordenadores personales y en ambos casos, pueden no estar bien securizados. Además, los trabajadores se conectan desde sus casas o desde donde sea, y es probable que las redes no estén tampoco muy bien protegidas, además de compartir la red con otros dispositivos y es ahí donde planteamos este caso de uso.

Esta máquina cliente está conectado a la red de mi casa, por lo que tiene una IP en 192.168.1.0/24, y luego realiza una conexión vpn hacia el Pfsense. Las máquinas de la plataforma están detrás del Pfsense y tienen una IP en la red que gestiona Pfsense, una 10.0.0.0/24, aislada de mi red doméstica. Por otro lado, instanciamos otra máquina que hará de atacante, también en el proxmox, que solo estará conectada a la red doméstica y, por tanto, tiene conectividad con la máquina cliente, por lo que puede atacarla.

Este escenario me pareció interesante porque creo que es uno al que se enfrentan a diario muchos administradores de sistemas y analistas de seguridad. Hoy en día, muchos ataques informáticos tienen su origen en la infección de una máquina interna de la organización. Así que planteamos una situación parecida, podemos intentar atacar una máquina que sería la de un empleado, en donde él podría no ser consciente de dicho ataque, pero que si llegaría de los responsables de sistemas y seguridad a través de la plataforma, pudiendo éstos notificar al empleado para que tome las medidas oportunas para proteger su equipo o su red doméstica.

Pasos que se han seguido:

1. Conectamos la máquina cliente a la vpn de Pfsense y comprobamos que tiene conectividad con el resto de las máquinas de la red.

2. Por otro lado, preparamos la máquina atacante. Comprobamos que tiene conectividad con la máquina cliente. Para realizar el ataque se puede hacer de manera automatizada utilizando muchas de las herramientas disponibles en Internet, e incluso, podríamos haber utilizado una distribución como Kali Linux para que hiciera de atacante, ya que incluye un gran elenco de herramientas para realizar ataques.
3. En este caso vamos a hacerlo manual para controlar que cuando intencionadamente introducimos mal las credenciales de acceso, se nos genera una alerta en Wazuh y esta a su vez en TheHive. Al crearse en TheHive, vemos que se notifica al webhook de n8n y que éste crear un caso nuevo, extrae observables como la IP y la envía a Cortex para su análisis.
4. Para este caso, probamos a crear una plantilla para el caso en TheHive y así probamos otra funcionalidad que tenemos disponible.

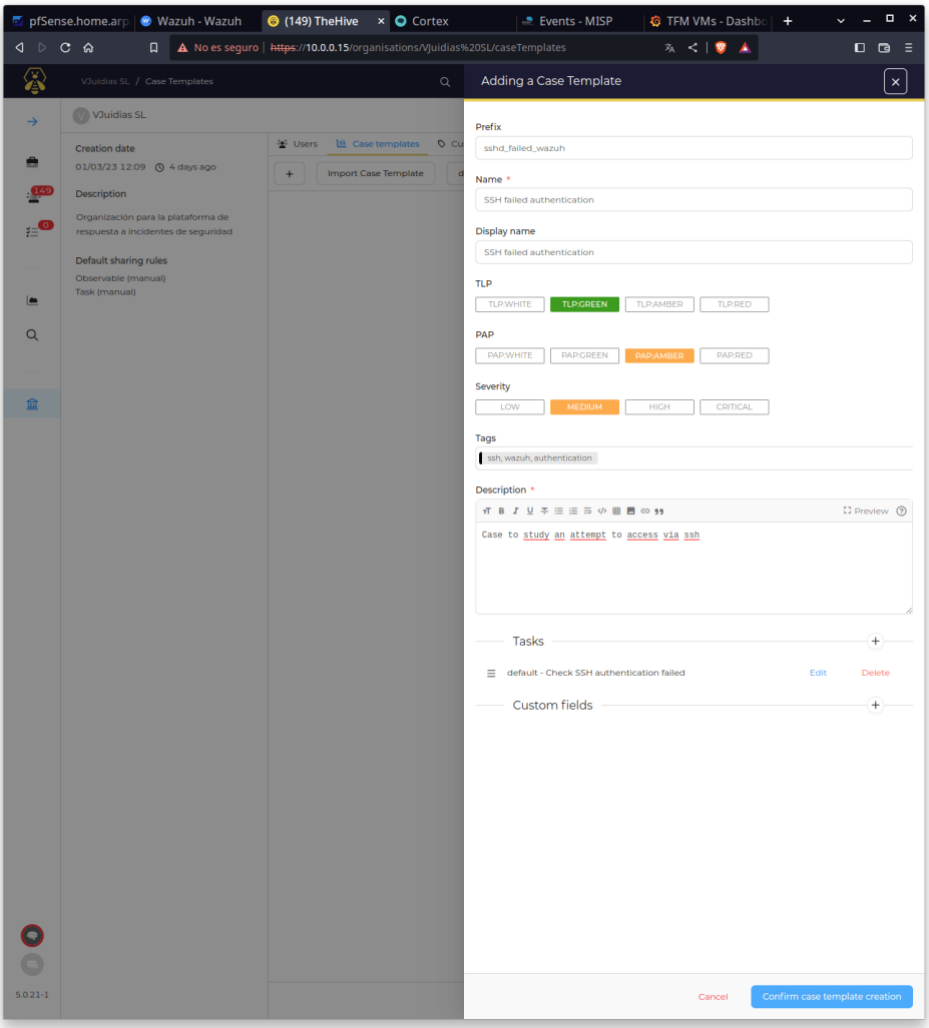

Ilustración 66 - Crear una plantilla para un caso en TheHive

5. Este será la plantilla que usaremos en n8n para generar el caso. Configuramos el workflow para que solo nos procese las alertas se “sshd authentication failed”

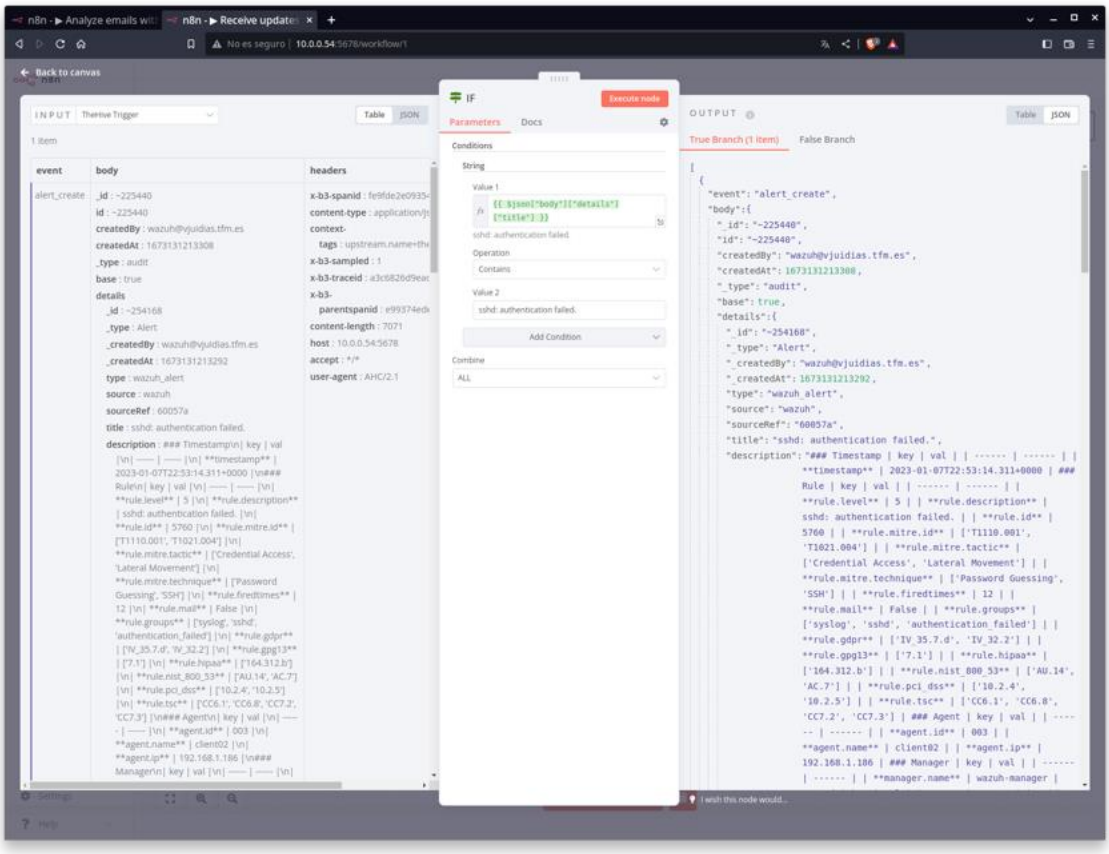

Ilustración 67 - Input & Oputput entre dos nodos en n8n

6. Con esto, al generar la alerta, se nos creará un nuevo caso en TheHive

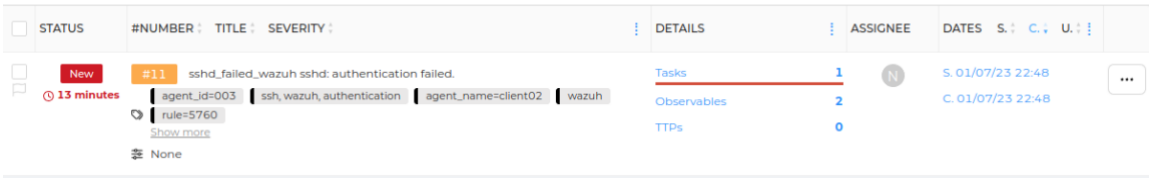

Ilustración 68 - Caso creado de un intento fallido de acceso por ssh

7. Podemos procesar los observables con los analizadores de Cortex desde la propia interfaz de TheHive, pero lo vamos a hacer utilizando n8n de forma automática.

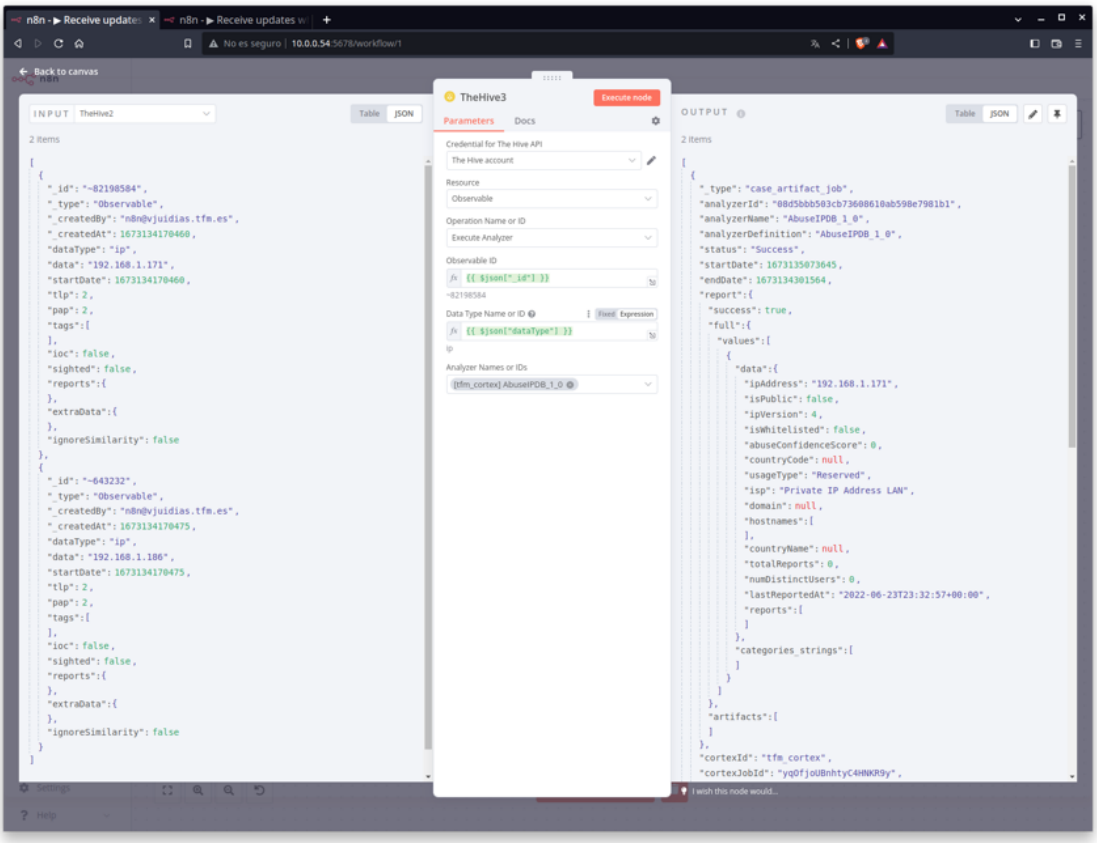

Ilustración 69 - Extraer y analizar los observables de un caso en Cortex

En este caso, vemos el input de entrada y de salida del nodo. Como input, tenemos todos los observables del caso en cuestión, del cual extraemos el observable que queramos. En este caso, sabemos cuál es la IP del atacante, así que es el que usamos para analizarlo en Cortex. Por el lado del output, vemos el resultado del análisis de Cortex utilizando el analizador de “AbuseIPDB” como ejemplo. En el mismo cuadro donde añadimos el analizador, podemos añadir todos los que queramos, ya que permite una selección múltiple.

Cuando creamos la conexión en n8n para TheHive teníamos la posibilidad de omitir la verificación del certificado, para Cortex no tenemos esa opción, por lo que, por desgracia, no podemos utilizar este nodo para lo que queríamos. Al menos no si utilizamos conexiones SSL con un certificado que no sea válido.

8. Si queremos, una vez realizadas las tareas correspondientes, podríamos ejecutar Wazuh como Responder desde TheHive para tomar medidas contra la IP del atacante.

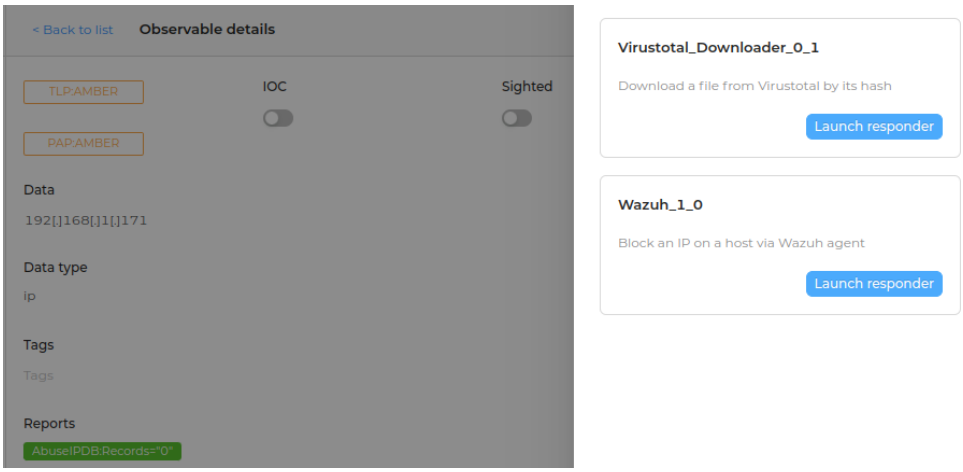

Ilustración 70 - Ejecutar el responder de Wazuh en TheHive

9. Una vez tomadas las acciones oportunas, que bien podrían ser otras, podemos documentar la incidencia y dar por cerrado el caso en TheHive.

En un caso como este tanto la respuesta como la actividad post-incidente puede variar. Es obvio que si tenemos múltiples intento de acceso por ssh no autorizado, lo lógico es bloquear esa IP para parar el ataque, pero eso puede ser solo una parte de la respuesta, ya que esta dependerá de cada organización. Con la actividad post-incidente ocurre algo similar, en vista de lo ocurrido se puede decidir tomar medidas para, como es este caso, proteger más los equipos de los trabajadores que están en remoto o limitar el acceso de éstos a recursos de la empresa.

6. Conclusiones

Durante el inicio de este trabajo se plantearon una serie de objetivos a realizar para culminar con éxito el trabajo. Por una parte, hubo una serie de objetivos más orientados a la parte teórica, dedicada a la investigación y al estudio de la bibliografía existente sobre la detección y la gestión de incidentes. Se llevó a cabo un estudio sobre las diferentes herramientas existentes para cada ámbito y se hizo una selección de las que eran más adecuadas para desarrollar lo que se había planteado en un inicio. Con las herramientas en mano y con el objetivo claro de que es lo que ser quería lograr con ellas, hubo que plantear un diseño realista para la plataforma que pudiera dar servicio a una empresa u organización, planteando donde estarían ubicados cada uno de los elementos y como se integrarían unos con otros. También se planteó que papel jugaba cada una de las herramientas en un ciclo de vida de un incidente de seguridad.

Una vez hecho esto, pasamos a la parte más práctica del trabajo. Para ello, una vez seleccionadas las herramientas y como sería el diseño de la plataforma, hubo que plantear un caso de uso donde poner en práctica todo lo anterior. Se planteó como se implementaría teniendo en mente un caso de uso práctico y los recursos con los que se contaba para lograrlo. Se decidió hacerlo sobre máquinas virtuales, ya que no había recursos físicos suficientes para implementar cada herramienta por separado, y que dichas máquinas correrían sobre dos máquinas físicas.

Entre los objetivos planteados era que la solución planteada debía ser escalable y automatizada. Todas las herramientas cuentan con la posibilidad de crear un clúster de X nodos para escalar de manera horizontal si fuera necesario. Y todas ellas tienen documentado su instalación y configuración de manera automatizada, utilizando herramientas como Ansible o Puppet.

Aunque en el caso de uso planteado no se hayan podido probar algunas de estas funcionalidades, como la escalabilidad o la alta disponibilidad, las herramientas seleccionadas sí que disponen de todas esas posibilidades de forma nativa. A fin de cuentas, la idea tras este proyecto era que este trabajo pudiera servir como prueba de concepto e incluso tener un carácter pedagógico.

Por tanto, hemos podido cumplir con los objetivos que se plantearon al inicio de los objetivos, desarrollar, utilizando herramientas *opensource*, una Plataforma de Respuesta a Incidentes de Seguridad con capacidad de detección, análisis, gestión de los casos y respuesta a los mismo, que es escalable, con alta disponibilidad y automatizable y, por supuesto, perfectamente capaz de implantarse en un entorno real.

La idea tras este proyecto era que este trabajo pudiera servir de objetivo pedagógico o como prueba de concepto para que una empresa u organización, sin importar su tamaño, pudiera plantearse ponerla en marcha para ayudar a proteger sus sistemas. Como hemos dicho, las herramientas escogidas permiten que la plataforma pueda escalar sin problemas y se pueda disponer de alta

disponibilidad para no tener una pérdida del servicio. El único requisito es el hardware disponible, ya sea físico o en la nube.

Otra parte importante es que solo hemos rascado la superficie de las herramientas seleccionadas. Cada una de estas herramientas dispone de amplio abanico de funcionalidades, permitiendo que puedan adaptarse a muchos escenarios o necesidades, aunque, por desgracia, en este trabajo por falta de tiempo y recursos, no se hayan podido ver y explicar cada una de ellas.

6.1. Trabajo Futuro

Una vez concluido el trabajo hay una serie de aspectos que se podrían implementar o mejorar en un futuro:

- Utilizar conexiones SSL con certificados válidos. En este trabajo, hemos tenido que crear certificados autofirmados para poder tener conexiones cifradas entre las distintas herramientas. Esto ha supuesto una merma en la seguridad porque para que todo funcione como es debido, se han tenido que deshabilitar ciertas protecciones enfocadas a garantizar el uso de certificados válidos. De hecho, para n8n existe una integración con Cortex que no hemos podido implementar porque precisamente no tiene opción para deshabilitar esta protección, como si la tiene la integración con TheHive y, por tanto, da error al conectarse.

- Modificar las contraseñas por defecto de todas las herramientas. Hay algunas, como Wazuh, que generan una contraseña robusta y segura para el usuario “admin” nada más terminar la instalación y que podemos seguir utilizando sin problemas. Otras herramientas como TheHive, no, dan una contraseña por defecto muy sencilla que no sería viable utilizar en un entorno real.

- Por defecto n8n no implementa un sistema de autenticación de usuarios, al menos no en la versión *selfhosted*. A través de variables de entorno se puede configurar una autenticación y otros parámetros de seguridad, pero lo ideal sería que, al igual que el resto de las herramientas, vienese con un sistema de gestión de usuarios. En caso de no ser posible, siempre se puede implementar una autenticación básica a través Apache o Nginx.

- En una empresa u organización lo normal es tener un sistema centralizado de usuarios y grupos, un directorio activo, por ejemplo. Una mejora para considerar sería implementar la autenticación de usuarios a través de uno de estos sistemas, así no habría que crear el mismo usuario en varios sitios, lo cual, con el tiempo y si aumenta el número de usuarios, se vuelve algo inmanejable y un problema de seguridad.

- Implementar un sistema de HA. Todas y cada una de las herramientas soportan alta disponibilidad, por tanto, lo ideal sería implementar algo

básico para garantizar el servicio si alguno de los servidores o máquinas se cae.

- Al hilo de lo anterior, también sería interesante implementar cada una de las herramientas en su versión de cluster, con el mínimo indispensable, pero preparado por si en el futuro la plataforma necesitase crecer.

- Automatizar por completo toda la plataforma. En este trabajo no se ha podido abarcar por completo este aspecto, pero dado que todas las herramientas tienen documentado este tema, sería cuestión de implementarlo con alguna de las opciones disponibles.

- Utilizar analizadores de pago junto con los servicios gratuitos en Cortex. Hemos podido probar analizadores gratuitos, pero éstos suelen estar bastante restringidos en cuanto al uso diario permitido o la información accesible. No sería mala idea plantearse utilizar alguno de los servicios gratuitos en su modalidad de pago por uso, sobre todo si vamos a hacer un uso frecuente de ese analyzer o responder.

- En MISP utilizar la información compartida por organizaciones del entorno o el ámbito de la empresa u organización y no solo las que vienen por defecto con la herramienta. Es decir, adaptar la herramienta a las necesidades y el entorno en el que se mueve la empresa/organización.

6.2. Experiencia/aprendizaje que ha supuesto el TFM

Debo decir que ha sido una gran experiencia el desarrollo de este Trabajo Final de Máster. Ha sido un trabajo duro, pero muy gratificante, lo he disfrutado muchísimo.

Antes del desarrollo de este proyecto, tenía la impresión de que este tipo de sistemas eran grandes, complejos y muy caros. Tenía la idea de que eran complejos cuadros de mandos, cuya gestión recaía en gente muy especializada, que tendrían certificaciones acreditadas para trabajar con tales sistemas, porque eran sistemas cerrados. Quizás tenía esa impresión porque lo relacionaba con la gente que se dedica a dar soporte a sistemas como Windows, bases de datos, sistemas como SAP, etc. Y sobre todo porque en los sitios grandes o las empresas de “verdad” utilizan estos sistemas. Casualmente, tenía la misma impresión para los sistemas de monitorización, sistemas complejos, caros y difíciles de utilizar, solo apto para personal especializado.

Después del desarrollo de este trabajo, he podido comprobar que no es así. Como todo hoy día en el mundo de la informática, tenemos soluciones de todo tipo, ya sean *opensource*, de pago o sistemas cerrados. He disfrutado muchísimo haciendo este trabajo porque entre otras cosas, he visto que es posible implementar una plataforma de estas características utilizando herramientas de código abierto y, sobre todo, no ha sido tan complicado como pensaba, al menos no tanto cuando ya estás acostumbrado a implementar herramientas y a pelarte con la documentación para ponerlas en funcionamiento

y lidiar con los posibles problemas que puedan surgir. Me llevo la experiencia de que he sido capaz de llevar a cabo el trabajo planteado y de ver que podría ser capaz sin ningún problema, de implantar esta misma plataforma en mi trabajo actual, en el que soy administrador de sistemas, o en uno futuro. También que podría desempeñar un puesto como analista, por ejemplo, utilizando herramientas como las que aquí se han visto u otras del mismo tipo.

7. Glosario

CERT: son las siglas en inglés de ***Computer Emergency Response Team*** o un Equipo de Respuesta antes Emergencias Informáticas. Hace referencia a un centro o un equipo de personal especializado, provenientes de diferentes áreas que son responsables del desarrollo de tareas preventivas y reactivas ante incidentes de seguridad. También puede haber personal de otras áreas como personal del ámbito legal o de la comunicación.

CSIRT: son las siglas de ***Computer Security Incident Response Team***, Equipo de Respuesta ante Incidentes de seguridad, otro término utilizado para referirse al concepto anterior, un equipo de personas altamente cualificado preparado para hacer frente a incidentes de seguridad.

EDR: son las siglas en inglés de ***Endpoint Detection & Response***. Se corresponde con un software que se instala en equipo, clientes o servidores, para detectar amenazas.

Firewall: también llamado Cortafuegos en español, es un sistema informático destinado a proteger redes privadas, bloqueando el acceso no autorizado.

GDPR: son las siglas en inglés de ***General Data Protection Regulation*** o Reglamento General de Protección de Datos, es el reglamente europeo que garantiza la protección y privacidad de los datos de los usuarios europeos.

HIDS: son las siglas en inglés de ***Host Intrusion Detection System***, un software de detección basado en hosts.

IDS: son las siglas de ***Intrusion Detection System***, un software de detección de intrusos.

IPS: son las siglas de ***Intrusion Prevention System***, un software similar a un IDS, pero con capacidades reactivas, de bloquear un ataque o atacante una vez detectado.

Kanban: un método de gestión de proyectos que permite visualizar de manera fácil y sencilla a todos los miembros del equipo los flujos de trabajo y las tareas.

NIDS: son las siglas en inglés de ***Network Detection System***, un software de detección de intrusos basado en red.

SIRP: son las siglas en inglés de ***Security Incident Response Platform***, es una plataforma de software, compuesta por una o más herramientas cuyo objetivo es gestionar el ciclo de vida de un incidente de seguridad.

SGSI: son las siglas de Sistema de Gestión de la Información. Un conjunto de principios o procedimientos y políticas de seguridad enfocados a proteger los sistemas de una organización.

SIEM: son las siglas en inglés de ***Security Information & Event Managment***, un sistema que recibe información de alertas y eventos relacionados con la seguridad, cuyo objetivo es proporcionar a las empresas u organizaciones una respuesta rápida y precisa para detectar y responder ante amenazas de seguridad sobre sus sistemas.

SOC: son las siglas en inglés de ***Security Operation Center***. Son centros dedicados a prevenir, monitorizar y controlar la seguridad de redes y sistemas informáticos.

VPN: son las siglas en inglés de ***Virtual Private Network***, un tipo de conexión segura entre dos puntos a través de Internet.

8. Bibliografía

[1] «Log4Shell: análisis de vulnerabilidades en Log4j,» 24 02 2022. [En línea]. Available: https://www.incibe-cert.es/blog/log4shell-analisis-vulnerabilidades-log4j. [Último acceso: 01 10 2022].

[2] P. Lanzas, «El negocio del cibercrimen,» 14 07 2016. [En línea]. Available: https://www.elplural.com/economia/el-negocio-del-cibercrimen_84330102.

[3] «Wazuh · The Open Source Security Platform,» [En línea]. Available: https://wazuh.com/. [Último acceso: 04 10 2022].

[4] «¿Qué son y para qué sirven los SIEM, IDS e IPS?,» [En línea]. Available: file:///Users/vjuidias/Zotero/storage/GHDFBMLC/son-y-sirven-los-siem-ids-e-ips.html. [Último acceso: 01 10 2022].

[5] «Snort - Network Intrusion Detection & Prevention System,» [En línea]. Available: https://www.snort.org/.

[6] «Suricata: Observe. Protect. Adapt,» [En línea]. Available: https://suricata.io/. [Último acceso: 01 10 2022].

[7] «Wikipedia - iptables,» [En línea]. Available: https://en.wikipedia.org/wiki/Iptables. [Último acceso: 01 10 2022].

[8] «Wikipedia - nftables,» [En línea]. Available: https://en.wikipedia.org/wiki/Nftables. [Último acceso: 01 10 2022].

[9] «Wikipedia - firewalld,» [En línea]. Available: https://en.wikipedia.org/wiki/Firewalld.

[10] «pfSense® - World's Most Trusted Open Source Firewall,» [En línea]. Available: https://www.pfsense.org/. [Último acceso: 01 10 2022].

[11] «Wikpedia - WAF,» [En línea]. Available: https://en.wikipedia.org/wiki/Web_application_firewall. [Último acceso: 01 10 2022].

[12] «¿Qué es un WAF (Web Application Firewall)?,» [En línea]. Available: https://www.f5.com/es_es/services/resources/glossary/web-application-firewall.html. [Último acceso: 01 10 2022].

[13] «Logstash,» [En línea]. Available: https://www.elastic.co/es/logstash/. [Último acceso: 01 10 2022].

[14] «Logstash - Beats,» [En línea]. Available: https://www.elastic.co/es/beats/. [Último acceso: 01 10 2022].

[15] «Búsqueda gratuita y abierta: Los creadores de Elasticsearch, ELK y Kibana | Elastic,» [En línea]. Available: https://www.elastic.co/es/. [Último acceso: 01 10 2022].

[16] «Kibana: Explora, visualiza y descubre datos | Elastic,» [En línea]. Available: https://www.elastic.co/es/kibana/. [Último acceso: 01 10 2022].

[17] «ElastAlert - Easy & Flexible Alerting With Elasticsearch — ElastAlert 0.0.1 documentation,» [En línea]. Available: https://elastalert.readthedocs.io/en/latest/. [Último acceso: 01 10 2022].

[18] «Wazuh Indexer,» [En línea]. Available: https://documentation.wazuh.com/current/installation-guide/wazuh-indexer/index.html. [Último acceso: 01 10 2022].

[19] «Wazuh Dashboard,» [En línea]. Available: https://documentation.wazuh.com/current/installation-guide/wazuh-dashboard/index.html. [Último acceso: 01 10 2022].

[20] «Wazuh announces version,» [En línea]. Available: https://benheater.com/wazuh-announces-version-4-3/. [Último acceso: 01 10 2022].

[21] «OpenSearch,» [En línea]. Available: https://github.com/opensearch-project/OpenSearch. [Último acceso: 01 10 2022].

[22] «TheHive Project,» [En línea]. Available: https://www.thehive-project.org. [Último acceso: 01 10 2022].

[23] «Cortex: a Powerful Observable Analysis and Active Response Engine,» [En línea]. Available: https://github.com/TheHive-Project/Cortex. [Último acceso: 01 10 2022].

[24] «MISP Open Source Threat Intelligence Platform & Open Standards For Threat Information Sharing,» [En línea]. Available: https://www.misp-project.org. [Último acceso: 01 10 2022].

[25] «Sistemas EDR,» [En línea]. Available: file:///Users/vjuidias/Zotero/storage/JM4CLC76/sistemas-edr-son-y-ayudan-proteger-seguridad-tu-empresa.html. [Último acceso: 01 10 2022].

[26] «OSSEC - World's Most Widely Used Host Intrusion Detection System - HIDS,» [En línea]. Available: https://www.ossec.net. [Último acceso: 01 11 2022].

[27] «OSSEC Versions,» [En línea]. Available: https://www.ossec.net/ossec-downloads/. [Último acceso: 06 11 2022].

[28] «Wazuh agent - Components · Wazuh documentation,» [En línea]. Available: https://documentation.wazuh.com/current/getting-started/components/wazuh-agent.html. [Último acceso: 03 11 2022].

[29] «Wazuh server - Components · Wazuh documentation,» [En línea]. Available: https://documentation.wazuh.com/current/getting-started/components/wazuh-server.html. [Último acceso: 04 11 2022].

[30] «HAProxy - Wikipedia,» [En línea]. Available: https://en.wikipedia.org/wiki/HAProxy. [Último acceso: 04 11 2022].

[31] «Wazuh - Orchestration,» [En línea]. Available: https://github.com/wazuh/wazuh#orchestration. [Último acceso: 04 11 2022].

[32] «Wazuh - OVA,» [En línea]. Available: https://documentation.wazuh.com/current/deployment-options/virtual-machine/virtual-machine.html. [Último acceso: 04 11 2022].

[33] «Open Information Security Foundation,» [En línea]. Available: https://oisf.net. [Último acceso: 05 11 2022].

[34] «TheHivev5,» [En línea]. Available: https://docs.strangebee.com. [Último acceso: 06 11 2022].

[35] «MISP Features,» [En línea]. Available: https://www.misp-project.org/features/. [Último acceso: 06 11 2022].

[36] «n8n Automate without limits,» [En línea]. Available: https://n8n.io. [Último acceso: 02 12 2022].

[37] «Ansible: Automation for everyone,» [En línea]. Available: https://www.ansible.com. [Último acceso: 03 12 2022].

[38] «Wazuh Quuickstart,» [En línea]. Available: https://documentation.wazuh.com/current/quickstart.html. [Último acceso: 29 11 2022].

[39] «From metrics to insight,» [En línea]. Available: https://prometheus.io. [Último acceso: 05 12 2022].

[40] «Grafana Labs,» [En línea]. Available: https://grafana.com. [Último acceso: 05 12 2022].

[41] «TheHive - Installation & configuration guides,» [En línea]. Available: https://docs.strangebee.com/thehive/setup/. [Último acceso: 10 12 2022].

[42] «Cortex - Quick start guide,» [En línea]. Available: https://docs.thehive-project.org/cortex/user-guides/first-start/#step-7-optional-create-an-account-for-thehive-integration. [Último acceso: 29 12 2022].

[43] «The Hive - About Cortex,» [En línea]. Available: https://docs.strangebee.com/thehive/administration/cortex/#introduction. [Último acceso: 02 01 2023].

[44] «Cortex Analyzer & Responder Requirements Guide,» [En línea]. Available: https://github.com/TheHive-Project/CortexDocs/blob/master/analyzer_requirements.md. [Último acceso: 02 01 2023].

[45] «MSIP Default Feeds,» [En línea]. Available: https://www.misp-project.org/feeds/. [Último acceso: 05 01 2023].

[46] «Using Wazuh and TheHive for threat protection and incident response,» [En línea]. Available: https://wazuh.com/blog/using-wazuh-and-thehive-for-threat-protection-and-incident-response/. [Último acceso: 05 01 2023].

[47] «n8n Documentation - Installation Guide,» [En línea]. Available: https://docs.n8n.io/hosting/installation/npm/. [Último acceso: 07 01 2023].

Anexo 1

Recolectar los logs de Suricata y del firewall de Pfsense.

Para hacer esto tenemos que crear un grupo de agentes en Wazuh, que es una funcionalidad destinada a gestionar los agentes mediante grupos. Estos grupos se pueden configurar para que, entre otras cosas, recolecten determinados ficheros de logs, como son los de Suricata y el firewall de Pfsense.

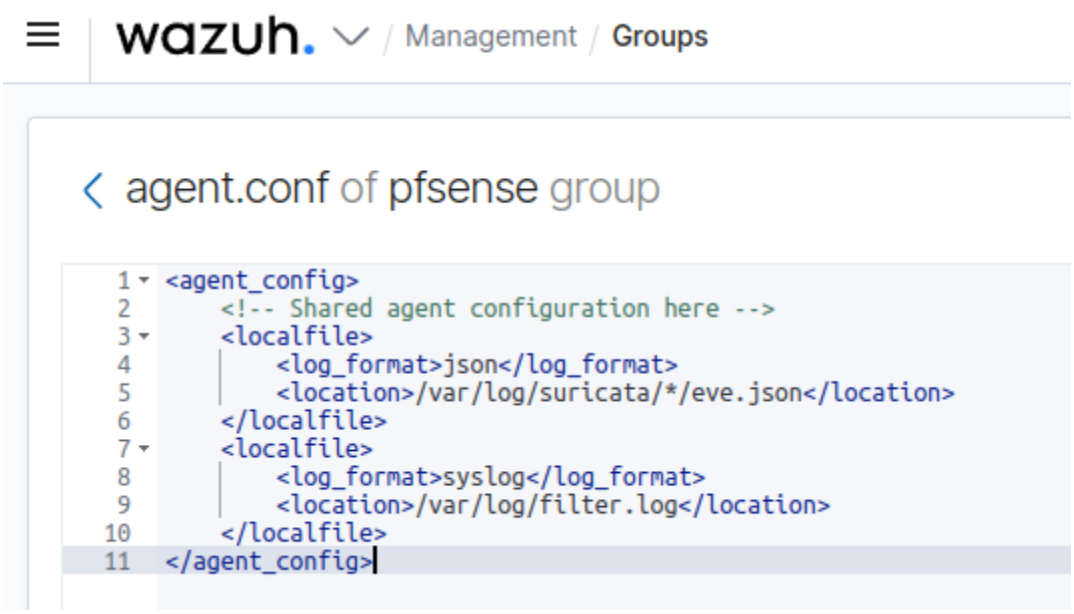

Ilustración 71 - Configuración de un grupo de agentes para Pfsense

Con esta simple configuración, tendremos los logs de estos dos servicios disponibles en Wazuh.

Anexo 2

Monitorización de la plataforma mediante Prometheus y Grafana.

Como se ha dicho, Prometheus es un sistema de monitorización que se ha escogido para monitorizar las máquinas del proyecto y vigilar los recursos que consumen y saber si hemos hecho bien o no en reducirlos. Funciona mediante un fichero YAML donde se definen los hosts donde están desplegados los "*exporters*", los ejecutables que recolectan las métricas de CPU, RAM, red, disco, etc. Estos agentes, vuelcan esta información en el puerto 9100 de la máquina donde están alojados y donde Prometheus irá a recolectar la información cada X tiempo.

root@monitor: /usr/local/bin/prometheus

```
# my global config
global:
  scrape_interval: 15s # Set the scrape interval to every 15 seconds. Default is every 1 minute.
  evaluation_interval: 15s # Evaluate rules every 15 seconds. The default is every 1 minute.
  # scrape_timeout is set to the global default (10s).

# Alertmanager configuration
alerting:
  alertmanagers:
    - static_configs:
        - targets:
          # - alertmanager:9093

# Load rules once and periodically evaluate them according to the global 'evaluation_interval'.
rule_files:
  # - "first_rules.yml"
  # - "second_rules.yml"

# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries scraped from this config.
  - job_name: "prometheus"

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.

    static_configs:
      - targets: ["localhost:9090"]

  - job_name: "tfm"

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.

    static_configs:
      - targets: ["10.0.0.11:9100"]
        labels:
          name: 'wazuh'
          ip: '10.0.0.11'

      - targets: ["10.0.0.15:9100"]
        labels:
          name: 'thehive'
          ip: '10.0.0.15'

      - targets: ["10.0.0.16:9100"]
        labels:
          name: 'cortex'
          ip: '10.0.0.16'

      - targets: ["192.168.1.180:9100"]
        labels:
          name: 'misp'
          ip: '192.168.1.180'

      - targets: ["192.168.1.143:9100"]
        labels:
          name: 'n8n'
          ip: '192.168.1.143'
```

Ilustración 72 - Fichero de configuración de Prometheus

La información que recolectan los exporters la podemos ver en cualquier navegador yendo al puerto 9100.

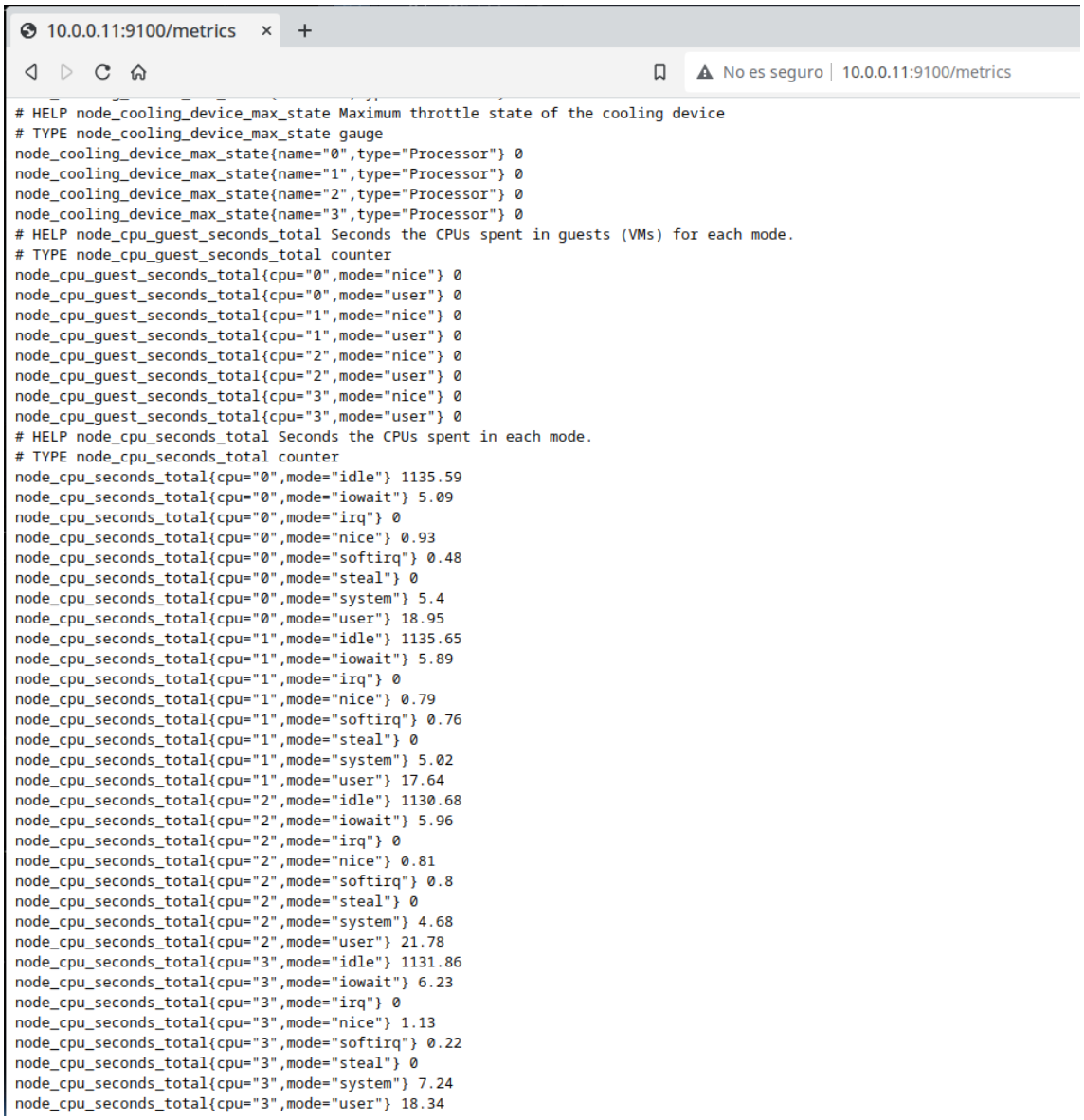

Ilustración 73 - Métricas recolectadas por un exporter

La información sobre las métricas y las máquinas se puede visualizar en la interfaz web de Prometheus. También se pueden hacer consultas.

Ilustración 74 - Máquina con el exporter activo

Esta información la podemos visualizar en Grafana para ver su evolución en el tiempo. Para eso, tenemos que añadir el servidor de Prometheus como una fuente de datos y luego escribir las consultas adecuadas para obtener la información que queremos que se muestre.

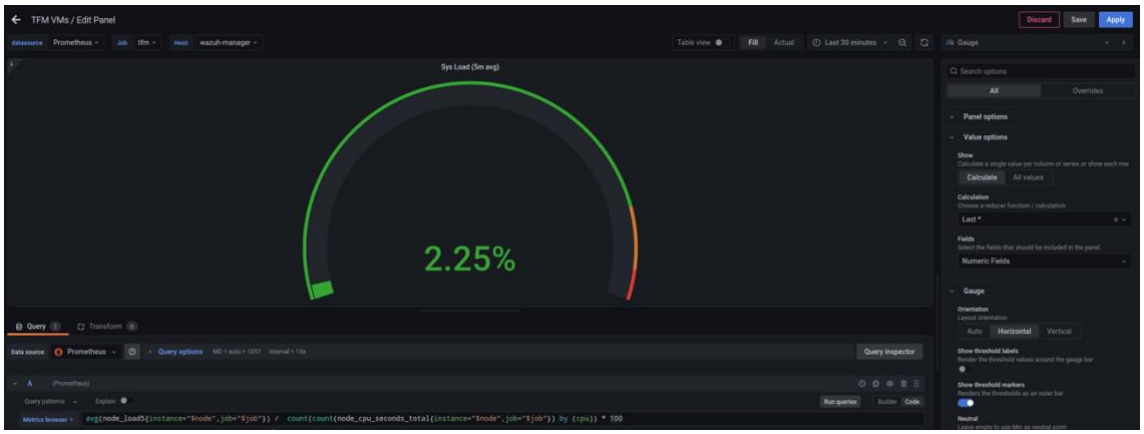

Ilustración 75 - Ejemplo de dashboard en Grafana

Anexo 3

Proxmox se ha utilizado como hipervisor en el mini PC Intel NUC para montar una parte de las máquinas virtuales del laboratorio.

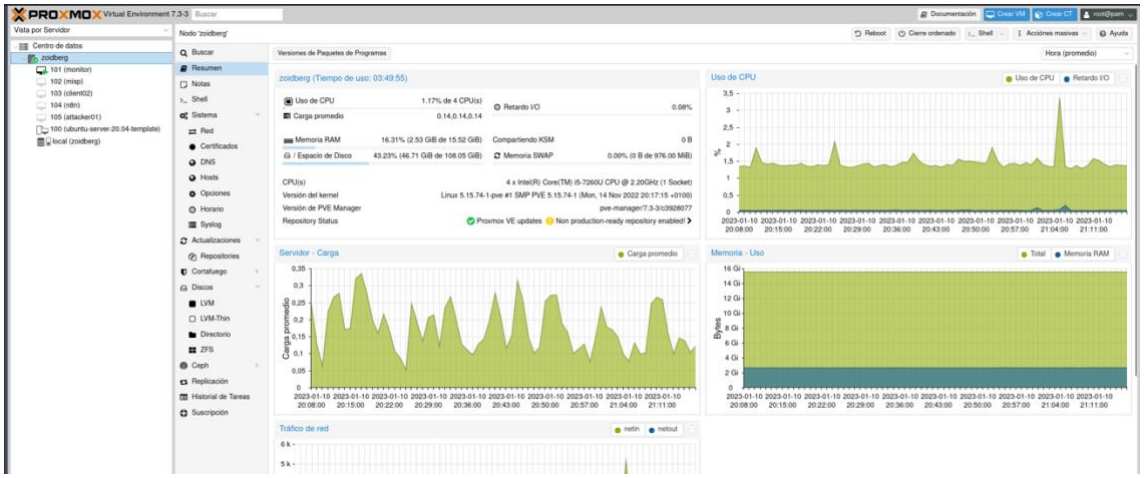

Ilustración 76 - Página principal de Proxmox

Anexo 4

Para configurar Wazuh con sus tres elementos en diferentes hosts, hay que configurar un fichero de configuración que se utilizará para ir uno a uno instalando los distintos componentes. El orden es primero el Indexer, luego el Server y, por último, el Dashboard.

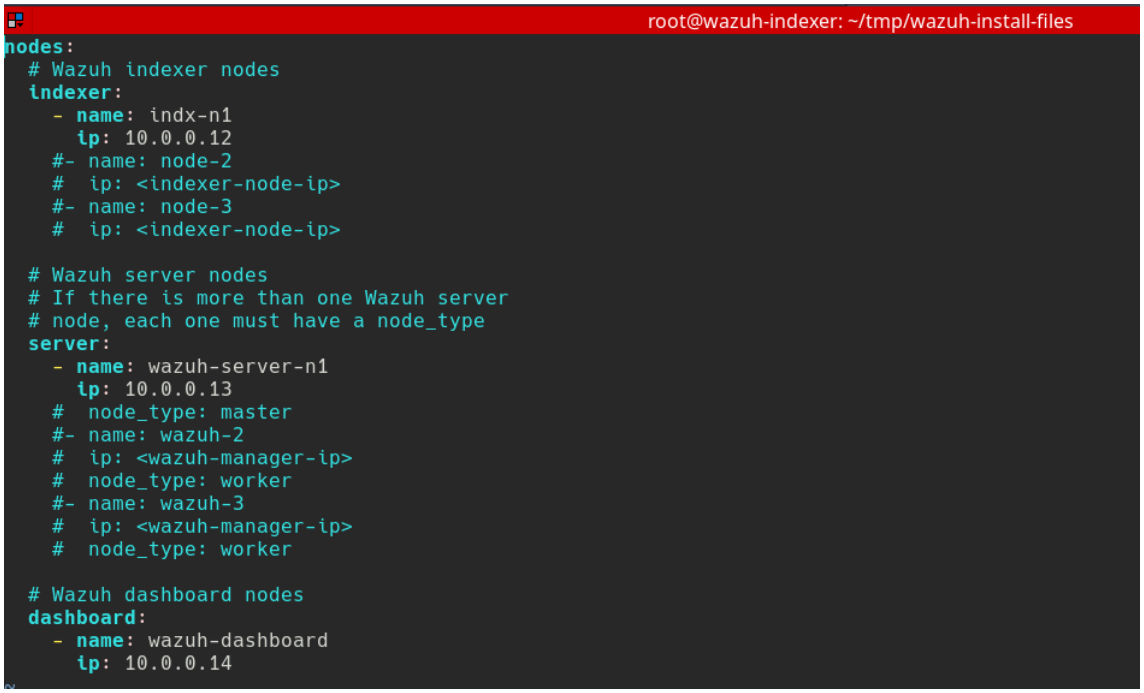

Ilustración 77 - Fichero de configuración de Wazuh para el stack

En este mismo fichero también definimos si vamos a utilizar un clúster de nodos o no. Durante la instalación del Indexer se generará una contraseña y una serie de ficheros que se utilizarán en los demás para llevar a cabo la instalación y la integración de todos ellos. Hay que tener definidas las direcciones IPs de todos los elementos antes de empezar con la instalación.