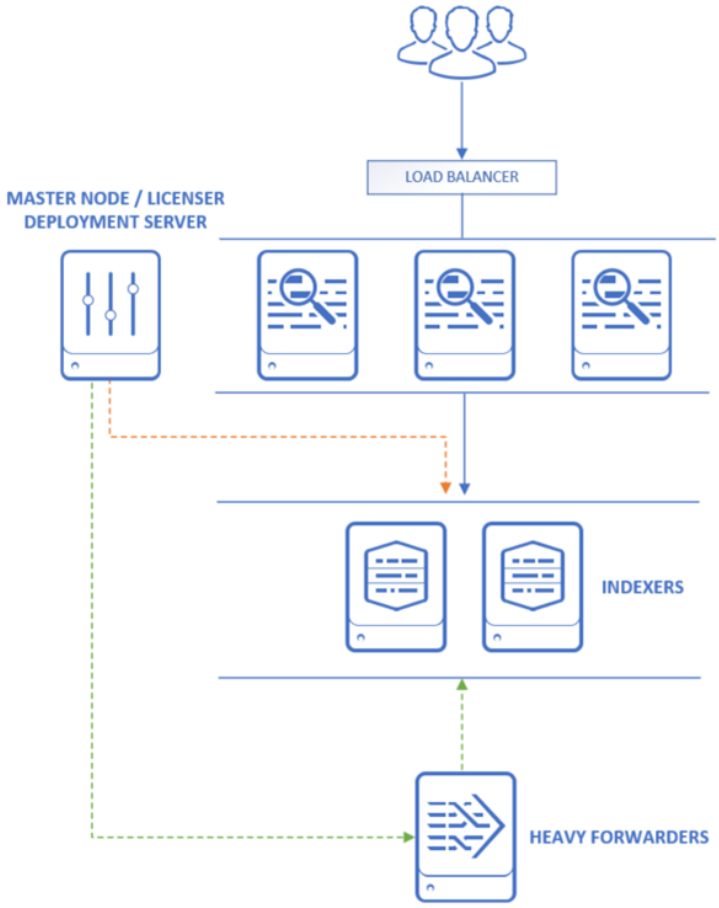
David Fernández Alejo

CASO PRÁCTICO SIEM



Sabiendo que la arquitectura es la indicada en la anterior imagen, se describirían brevemente cada uno de los componentes de la solución *on premise* basada en Splunk.

Una solución on premise se caracteriza por el hecho de que el cliente dispone de una herramienta SIEM desplegada dentro de su infraestructura. Todas las capas de las que se compone una arquitectura SIEM quedan alojadas dentro de la infraestructura del cliente. A su vez, el cliente deberá asumir los costes de la tecnología desplegada, así como de su mantenimiento administración y explotación. En el caso de que sea necesario un despliegue inicial de la herramienta SIEM, los tiempos dependerán de los recursos del cliente y, por lo general, serán más largos que en la modalidad *as a service*. Otra de las principales características del SIEM, es que la herramienta deberá dimensionarse según las necesidades del cliente y, en caso de aumentar los requerimientos, el cliente deberá de hacer frente a nuevas adquisiciones de *hardware* o *software*.

Sabiendo que la arquitectura desplegada cuenta con distintas capas (capa de recolección, capa de almacenamiento, capa de correlación y capa de presentación), se

explicará los componentes de cada capa, sabiendo que cada una de ellas tiene unas funcionalidades y características totalmente diferenciadas.

En primer lugar, en la parte superior de imagen tenemos el load balancer, que se encarga de distribuir los datos entre las distintas instancias receptoras, en este caso de los *Search head cluster.* Cada receptor obtiene una de las partes de los datos totales y, en conjunto, los receptores almacenan todos los datos. El equilibro de carga permite el escalamiento horizontal (permite que la infraestructura crezca de manera eficiente y resistente al añadir más nodos, distribuyendo las cargas de trabajo para garantizar un rendimiento óptimo en situaciones de alta demanda) para un mejor rendimiento.

A su vez, tenemos el master node, licenser y deployment server, a continuación, se describirán cada uno de ellos:

- El master node es el responsable de gestionar un cluster de indexadores, coordina y asegura la replicación de los datos entre los indexadores del clúster y supervisa el estado de los indexadores, si uno de ellos falla, el master node reequilibra la carga y los datos.
- El licenser cumple la función de administrador de licencias, gestiona y monitorea el uso de la licencia de la plataforma. Se utiliza para garantizar que la plataforma se utilice dentro de los límites de la licencia adquirida, monitoreando el volumen de los datos que se indexan diariamente, aplicando políticas de licencias y gestionando posibles violaciones si se exceden los límites.
- El deployment server, actúa como administrador de configuración centralizado que se encarga de gestionar la configuración y distribución de aplicaciones y actualizaciones a otros nodos, conocidos como clientes de despliegue. Facilita la administración eficiente de múltiples forwarders, indexadores y otros nodos de splunk, permitiendo actualizaciones automatizadas y consistentes, reduciendo el esfuerzo manual y el riesgo de errores.

En la capa de correlación (se encarga de relacionar eventos provenientes de diferentes fuentes de datos) se incluyen los Search headers, que son los servidores que gestionan las consultas de los usuarios y se encargan de analizar y presentar los datos de los indexadores. Es el servidor en la arquitectura Splunk donde los usuarios interactúan para realizar búsquedas, crear paneles, informes, alertas y análisis sobre los datos almacenados en los indexadores. El Search head, no almacena los datos, sino que actúa como intermediario, distribuyendo las consultas a los nodos indexadores.

En la capa de almacenamiento (la capa de almacenamiento se encarga de almacenar la máxima cantidad posible de información para ser procesada) se encuentran los *indexers,* servidores responsables de recibir, procesar, y almacenar los datos en índices,

lo que permite realizar búsquedas rápidas y eficientes en grandes volúmenes de datos. Los eventos se asocian a un índice, de forma que la información está segmentada y se pueden aplicar diferentes periodos de retención por índice. Los *indexers* clasifican los logs en diferentes estados y permiten el cambio de uno a otro para mejorar el rendimiento de las búsquedas.

En la capa de recolección (realiza la integración o agregación de los eventos de las diferentes soluciones de seguridad) se encuentra el *Heavy forwarder,* un servidor que se encarga de recibir los eventos que las distintas tecnologías envían. Además, puede realizar un procesamiento avanzado de datos antes de reenviarlo a los indexadores. Esto incluye tareas como filtrado, transformación y extracción de campos.

A continuación, se responderán a las preguntas indicadas en el enunciado.

1. Dado que él directamente no gestiona este proyecto, y de cara a unos requisitos de auditoría interna que le han fijado, te solicita que le especifiques, sobre la actual arquitectura, dónde se dispone de alta disponibilidad y dónde no, en cuyo caso habrás de plantear una solución plausible para tal fin.

La arquitectura de alta disponibilidad o HA se refiere a un conjunto de estrategias y configuraciones diseñadas para garantizar que un sistema o servicio esté disponible y funcione correctamente durante el mayor tiempo posible, minimizando el tiempo de inactividad o las interrupciones. Teniendo en cuenta lo que significa la alta disponibilidad, se indicarán aquellos componentes de la arquitectura mencionada en el enunciado que tengan varios servidores para la realización de la misma tarea.

En el caso de la arquitectura planteada en el enunciado, se puede observar que tenemos alta disponibilidad en la capa de almacenamiento, correlación y en la capa de presentación.

En primer lugar, en la capa de presentación y correlación, tenemos varios *Search Head,* cuya función principal es la gestión de consultas de los usuarios, se encargan de analizar y presentar los datos de los indexadores. Al tener un cluster de *Search Head,* si uno de los nodos de búsqueda falla, los otros nodos continúan gestionando las búsquedas. En Splunk un miembro del cluster tiene el rol de capitán, lo que significa que coordina las actividades de replicación y de trabajo entre todos los miembros. Los miembros del clúster se comunican entre sí para programar trabajos, replicar artefactos, actualizar configuraciones y coordinar otras actividades dentro del clúster. Los usuarios pueden acceder opcionalmente a los encabezados de búsqueda a través del balanceador de carga, el cual distribuye las solicitudes entre los distintos nodos, si uno de los nodos falla, el balanceador puede redirigir el tráfico al resto de nodo disponibles sin interrumpir el tráfico.

En segundo lugar, en la capa de almacenamiento tenemos varios *indexers*, cuya función principal es almacenar los datos. Se ocupan de almacenar eventos enviados por las diferentes fuentes a la plataforma. Cuando se implementa un clúster, se habilita y configura el nodo administrador del clúster y los nodos pares que realizan la indexación. También se habilita un cabezal de búsqueda para buscar datos en el clúster. A su vez, se configuran reenvíos para enviar datos al clúster. Los entornos con varios *indexers*, ofrecen solidas capacidades de recuperación ante desastres porque le permite distribuir copias de sus datos en varias ubicaciones. Si un sito deja de funcionar, la indexación y la búsqueda pueden continuar en los sitios restantes, sin interrupciones ni perdida de datos. Además, permite la afinidad de búsqueda, lo que reduce el tráfico de red al limitar las búsquedas a los datos locales.

Los elementos de la arquitectura planteada en el enunciado que no tienen alta disponibilidad son el: master node, licenser, deployment server y heavy forwarder.

En el caso del master node, la alta disponibilidad puede llegar a ser recomendada, ya que gestiona la replicación de datos, la distribución de buckets y asegura que se mantenga el factor de replicación y el factor de búsqueda en todo el clúster. Se puede recomendar la configuración de un master node de respaldo o una configuración en modo failover, para que en caso de que falle el principal se gestione por el de nodo de respaldo. Sin embargo, si se pierde el master node, no se produce ningún efecto en el funcionamiento del clúster, ni en la búsqueda, ni en la indexación. El efecto si falla, es que si un nodo indexador (adicional) falla antes de que se recupere el master node, las búsquedas se verán afectadas y los datos se indicarán como no buscables. El master node, no tiene estado, por lo que la recuperación no necesita ningún tipo de recuperación de datos.

Por otro lado, en el caso del Heavy Forwarder tiene sentido una configuración de alta disponibilidad, de modo que, si alguno de ellos no está activo, otros pueden aceptar los datos. En splunk se pueden configurar los Universal forwarders para enviar datos a múltiples heavys forwarders, cada Universal forwarder, envía los datos a más de un destino para garantizar la entrega. La solución externa a splunk para conseguir alta disponibilidad sería utilizar un balanceador de carga para que en caso de que uno falle se envíen los datos al nodo activo.

En el caso del licenser o el deployment server se podría implementar una solución de alta disponibilidad, pero no es muy habitual.

2. Adicionalmente, y con motivo de realizar una disminución de costes, te solicita que lleves a cabo un estudio sobre la viabilidad de modificar la arquitectura actual empleando otras alternativas posibles y que se las plantees, indicando sus características.

Con la pretensión de disminuir costes, se indicarán alternativas a la arquitectura proporcionada en el enunciado.

En primer lugar, siendo la arquitectura más barata, se plantea la solución All-in-one. En dicha arquitectura, todas las capas de las que se compone un SIEM, se alojan en un mismo servidor. Al no tener replicación o redundancia, cualquier fallo en algunas de las capas hace que la solución deje de funcionar. En Splunk, el indexer y el Search Head están en el mismo servidor.

En segundo lugar, se plantea la arquitectura distribuida, la cual se caracteriza por dividir las diferentes capas de las que se compone un SIEM en varios servidores, con el objetivo de aumentar las capacidades de recolección/almacenamiento o correlación. Al igual la solución *All-in-one,* no tiene ningún tipo de replicación o tolerancia a fallos, por lo que cualquier fallo en algunas de las capas hace que se pierdan las funcionalidades de la solución. Como punto diferenciador, se indica que, en casa de fallo en alguna de las capas, es posible que no haya perdida de información, a diferencia de la solución all-in-one. Dicha arquitectura tiene baja tolerancia a fallos y tiene una mejora de performance con respecto a la solución all-in-one. Dicha arquitectura en Splunk, consiste en separar los roles Search Head, Indexer y Forwarder en varios servidores.

Por último, se plantea una solución HA solamente en la capa de almacenamiento. Dicha arquitectura se caracteriza por tener alta disponibilidad en la capa de almacenamiento, en caso de pérdida de uno de los nodos de almacenamiento, no hay pérdida de la información reportada al SIEM. Dicha arquitectura tiene media tolerancia a fallos, tiene una mejora de performance con respecto a la arquitectura distribuida y a a su vez el coste de desplegar este clúster es mayor que los mencionados anteriormente. El despliegue de este tipo de arquitectura en Splunk se caracteriza por qué se debe crear un clúster de indexadores. Para la creación y gestión de dicho clúster es necesario desplegar otro rol llamado Master Cluster node.

3. De cara a realizar ese posible cambio de arquitectura, es necesario disponer de ciertos datos para buscar soluciones dentro del mercado que se ajusten a dicho escenario. Es por ello que, sabiendo que un evento son 560 bytes y que se necesita almacenarlos un mínimo de dos años por cumplimiento normativo, con un consumo medio de ingesta de 3000 EPS (events per second), se requiere conocer el tamaño en disco que se precisará para almacenar esa cantidad de datos, asumiendo una ratio de compresión de 10:1.

Evento = 560 bytes

Almacenamiento mínimo = 2 años

Consumo medio= 3.000 EPS

Ratio de compresión=10:1

En primer lugar, se calculan los eventos por día:

- Eventos por día = Eventos por segundo * segundos en un día
    - EPD= 3.000 * 60*60*24= 259.200.000 eventos por día

A continuación, se calcula el tamaño total de un día de eventos normalizados en disco:

- Tamaño de un día= Eventos por día * tamaño evento normalizado
    - TamDia= 259.200.000 * 560 = 145.152.000.000 bytes

El tamaño de un día comprimido:

- Tamaño_Cp = Tamaño_Día * Tasa_Compresión
    - Tamaño_Cp = 145,15 GB * 1/10 = 14.515.200.000 bytes

Finalmente, el tamaño total de almacenamiento necesario para dos años será el siguiente:

- Tamaño_Año = Tamaño_Cp * 730 = 14.515.200.000 * 730 = 10.596.096.000.000 bytes

Por lo tanto, el tamaño de disco necesario para almacenar dos años de este tipo de eventos es: **10,5961 TB**

4. Por otro lado, es necesario saber cómo otros fabricantes afrontan este tipo de implementaciones y conocer sus puntos débiles y fuertes para tener una clara visión de sus funcionalidades (limitar la respuesta a dos fabricantes únicamente).

Se va a aportar información sobre IBM QRadar y MicroFocus ArcSight, ambos se encuentran en las posiciones líderes del mercado según los informes Forrester y Gartner.

<u>IBM QRadar</u>

Los principales componentes de QRadar siguiendo el modelo de capas son los siguientes:

- En la capa de recolección se puede afrontar de dos formas:

    - Utilización del *Event Collector*, componente que puede ser tanto virtual cómo físico desde donde se configura la recolección de las distintas tecnologías.
    - Para los eventos relaccionados con productos de Microsoft, existe un agente llamado *WinCollect*, que es el encargado de la recolección de eventos.

  En la capa de recolección es donde se realiza la normalización y parseo de los distintos eventos.

- La capa de correlación es proporcionada por el componente llamado *Event Processor*, en el caso de los flows se llama *Flow Processor*.
- El almacenamiento por defecto de los eventos se realiza en la capa de correlación y es posible añadir un componente denominado Data Nodo para realizar un almacenamiento de larga duración.
- La capa de presentación se denomina QRadar Console, una interfaz web desde donde se centraliza la administración de los distintos componentes de la solución.

Además, se añaden otros componentes opcionales los cuales complementan el ecosistema SIEM, son los siguientes:

- Vulnerability Manager: se encarga de los escaneos de vulnearabilidad , así como de la gestión de las vulnerabiliades desde el punto de vista del seguimiento y remediación.
- Risk Manager: sirve para supervisar configuraciones de dispositivos como routers o switches para identificar cambios en entornos de red y priorizar riesgos.
- X-Force: Es el módulo de threat intelligence en QRadar, sirve para incorporar feeds de inteligencia y provee una serie de casos de uso para la monitorización del tráfico contra dichos feeds.
- UBA: proporciona capacidad de realizar análisis de patrones y tendencias de usuarios y activos.
- Watson: Es una aplicación que ayuda a la investigación y análisis de las alertas en QRadar, añadiendo información de los distintos parámetros de las alertas. La idea de enriquecer los eventos con esta información es ayudar a los analistas a descartar los falsos positivos de las alertas reales.

Algunos de los puntos fuertes de IBM QRadar son los siguientes:

- Permite integrar grandes volúmenes de datos de diferentes fuentes y los correlaciona para detectar amenazas y anomalías de forma eficiente.
- Esta diseñado para aportar funcionalidades de alta disponibilidad (HA) y recuperación de desastres.

- Tiene la capacidad de personalizar alertas y reglas para ajustarse a las necesidades de seguridad de una organización.
- Ofrece un conjunto completo de funcionalidades para la gestión de eventos de seguridad, que van desde la recolección de datos, correlación, investigación y remediación de incidentes.

Algunos de los puntos débiles de IBM QRadar son los siguientes:

- No todos los eventos son normalizados en la capa de recolección.
- QRadar es capaz de generar ofensas basadas en casos de uso, pero la gestión de las ofensas se reduce a la notificación y acciones asociadas.
- El coste es elevado, similar a Splunk.
- Para manejar grandes volúmenes de datos en tiempo real y aprovechar todas sus capacidades, QRadar necesita una infraestructura de hardware sólida.
- QRadar es una plataforma SIEM tan poderosa como compleja, es por eso que su curva de aprendizaje tiene a ser un poco mayor que la de otras soluciones.

Microfocus ArcSight

Está diseñada como herramienta centralizadora de logs de distintas tecnologías cuyos principales objetivos son correlar toda la información para detectar amenazas de seguridad y garantizar que se cumple la cadena de custodia de los logs para cumplir con las auditorías.

Los principales componentes de Microfocus ArcSight siguiendo el modelo de capas son los siguientes:

- La capa de recolección está basada en agentes llamados connectors que se encargan de recibir/leer los eventos de las distintas tecnologías. Después de recibir los eventos, estos son normalizados, permitiendo realizar las tereas de filtrado, agregación, categorización y normalización. Se utilizan los *SmartConnectors* para recibir/leer eventos y a su vez ofrece un framework de desarrollo de conectores para las tecnologías que no son soportadas de forma nativa por el fabricante, llamado *FlexConnector*.
- En la capa de correlación y presentación, se utiliza el ESM, que se encarga de la correlación de eventos y el acceso a ellos.
- El almacenamiento de los eventos se realiza en el ESM, además, existe otro componente llamado *logger* que permite un almacenamiento de larga duración.

Además, al igual que IBM QRadar, ofrece otros componentes, entre los que destacan el ArcMC (se usa para la gestión centralizada de los conectores) y el Activate Framework (creado para compartir y distribuir reglas de correlación de forma centralizada).

Algunos de los puntos fuertes de Microfocus ArcSight son:

- Separación lógica entre la capa de correlación y la capa de almacenamiento
- Normalización, agregación y filtrado antes en capa de recolección
- *Pattern discovery*: herramienta de detección de amenazas que, con base en búsquedas predefinidas, permite la detección de patrones de comportamiento.
- La capacidad de correlación de eventos de seguridad en tiempo real es una de las principales fortalezas de ArchSight.
- Está especialmente diseñado para grandes empresas con requisitos de seguridad complejos.

Algunos de los puntos débiles de Microfocus ArcSight son:

- Tiene una curva de aprendizaje considerable y puede ser difícil de configurar y mantener sin un equipo especializado.
- Tiene un elevado coste de licenciamiento.
- Pobre capacidad de presentación de informes y cuadros de mandos.
- Las actualizaciones pueden ser complejas y disruptivas, requiriendo un planteamiento y ejecución cuidadosos para evitar problemas en la continuidad de los servicios.

Para poder implementar la arquitectura planteada en el enunciado, tanto IBM QRadar como ArcSight pueden gestionar arquitecturas distribuidas como se ha mencionado anteriormente en cada tecnología.

IBM QRadar tiene la posibilidad de la configuración de alta disponibilidad, en forma de nodo primario/secundario (el tráfico fluye en solo uno de los segmentos de red redundantes y los dispositivos primarios de la red procesan todo el tráfico, si el principal falla, el secundario toma el control) o mediante una agrupación en un cluster (los segmentos de red de alta disponibilidad realizan el equilibrio de carga entre ellos). Soporta configuraciones de balanceo de carga que distribuyen el tráfico y los eventos entre varios nodos o componentes del sistema. Además, QRadar también se integra con bases de datos configuradas para alta disponibilidad.

En Microfocus ArcSight permite configurar sus componentes clave, como los Event Processors (procesadores de eventos) y Event Receivers (receptores de eventos), en configuraciones redundantes. A su vez, puede distribuir la carga de trabajo entre múltiples servidores mediante los load balancers, lo que no solo mejora el rendimiento, sino que también ayuda a garantizar la disponibilidad del sistema. Además, es capaz de realizar un failover automático hacia otro servidor o nodo en el clúster, sin intervención manual.