Análisis de la plataforma Ossim

Sistema de gestión de la información Open Source

Autor: Adrián Puchades Olmos.
Dirigido por: Lourdes Peñalver Herrero.

Universidad Politécnica de Valencia
Master en Ingeniería de Computadores
Valencia, Diciembre 2008

Índice.

1. Introducción.

Si lanzamos una mirada al pasado, podemos ver como poco a poco la capa tecnológica ha ido ocupando un espacio importante en las empresas y eso ha hecho que la tecnología evolucione de manera rápida y eficiente. Se puede ver como el campo de la seguridad informática ha evolucionado, desde los primeros cortafuegos hasta los más avanzados IDS (Sistemas de Detección de Intrusos).

Sin ninguna, duda hoy en día podemos encontrar en el mercado un amplio repertorio de aplicaciones de seguridad, cada una con un fin especifico como son los cortafuegos, IDS, detectores de vulnerabilidades, programas de monitorización, detectores de anomalías, etc.

Sin embargo, esto tiene un coste que pocas empresas pueden soportar, porque no sólo requiere un gasto económico de software, sino que además se necesita de un personal técnico especializado, que emplee una gran cantidad de horas afrontando los miles de eventos que cada aplicación genera y que la mayoría de ellos son repetitivos o falsos positivos.

Las empresas que deciden realizar una inversión en proteger sus redes, adquieren numerosos dispositivos y pagan licencias con costes elevados. Y para su asombro, se dan cuenta que aun así se hace inviable gestionar la seguridad, ya que reciben miles de alertas, la mayoría de ellas falsos positivos y no saben cual es el estado real de su red.

Mencionando las palabras de Julio Casal integrante en el diseño del sistema Ossim, haré una clara descripción por el cual se ha creado este sistema: *“Nos sorprende que con el fuerte desarrollo tecnológico producido en lo últimos años que nos ha provisto de herramientas con capacidades como la de los IDS, sea tan complejo desde el punto de vista de seguridad, obtener una foto de una red y obtener una información con un grado de abstracción que permita una revisión práctica y asumible”*. Estas necesidades fueron las causantes del proyecto Ossim.

Ossim no es una nueva aplicación de seguridad desarrollada, si no que han desarrollado una plataforma que integra las mejores aplicaciones de seguridad Open Source en una única interfaz, aprovechando lo mejor de cada una de ellas, e interrelacionándolas para obtener una información final mucho más fiable y detallada.

2. ¿Que es Ossim?.

Ossim es la abreviatura de *Open Source Security Information Management System* (Sistema de gestión de la información de seguridad Open Source) desarrollado para gestionar la información de seguridad de una red. Es una distribución que integra más de 22 productos de seguridad todos ellos “Open Source” capaces de correlacionar entre ellos. Ossim es una plataforma compleja pero a su vez potente, ya que integra las soluciones de código libre de seguridad para la monitorización y detección de patrones de redes más conocidas (Snort, nessus, ntop, nmap, nagios, etc), integrándolas en una arquitectura abierta que se aprovechará de todas sus capacidades para aumentar la seguridad en las redes.

El objetivo de Ossim ha sido crear un framework capaz de recolectar toda la información de los diferentes plugins, para integrar e interrelacionar entre si y obtener una visualización única del estado de la red y con el mismo formato, con el objetivo de aumentar la capacidad de detección de anomalías, priorizar los eventos según el contexto en el que se producen y mejorar la visibilidad de la monitorización del estado de la red actual.

El sistema Ossim se puede dividir en 3 capas:

El nivel mas bajo “*preprocesado”*, se compone por un número de detectores, monitores denominados preprocesadores, dispersados por la red, se encargan de realizar la detección y generación de alertas que posteriormente enviarán la información al sistema central para la colección y correlación de los diferentes eventos:

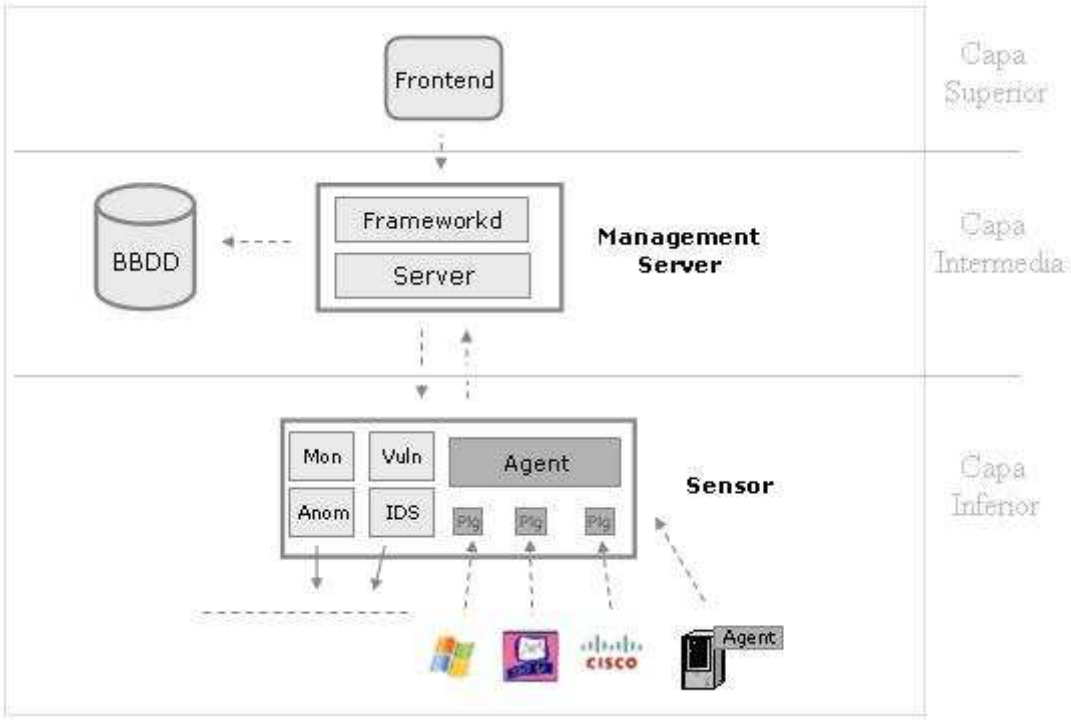
- IDS (detectores de patrones).
- Detectores de anomalías.
- Cortafuegos
- Varios tipos de Monitores

En el nivel intermedio se realiza el postprocesado, donde Ossim desarrolla un proceso de abstracción en el que millones de pequeños eventos incompresibles se convierten en singulares alarmas comprensibles, este proceso se lleva a cabo principalmente en el motor de correlación[1], donde el administrador crea directivas de correlación para unir diferentes eventos de bajo nivel en una única alarma de alto nivel, cuyo objetivo es aumentar la sensibilidad y la fiabilidad de la red.

- Normalización.
- Correlación.
- Priorización.
- Valoración de Riesgos.

[1] *Se define como un algoritmo que realiza una relación a través de unos datos de entrada y ofrece un dato de salida.*

Por último, en el nivel más alto “*Front-end*” se ubica una herramienta de gestión, capaz de configurar y visualizar tanto los módulos externos como los propios del framework, mediante ella podremos crear la topología de la red, inventariar activos, crear las políticas de seguridad, definir las reglas de correlación y enlazar las diferentes herramientas integradas.



1. Representación Ossim 3 capas.

3. El proceso de detección.

El principal objetivo del proyecto Ossim, ha sido el de aumentar la capacidad de detección ofrecida por los productos hasta hoy en día desarrollados. En este punto introduciremos cual es el secreto, de este proceso que el sistema Ossim lleva a cabo para aumentar dicha capacidad de detección.

El proceso de detección se le llama al proceso global desarrollado por el SIM, incluyendo tanto los distintos detectores y monitores de la red como los realizados por el sistema para procesar la información.

Detectores.

Llamamos detector a cualquier aplicación capaz de escuchar en la red en tiempo real en busca de patrones y producir eventos de seguridad ante la localización de situaciones previamente definidas.

La capacidad e Incapacidad de la detección.

La capacidad de detección de un detector la podemos definir mediante dos sustantivos:

- *Sensibilidad*, definida como la capacidad de análisis en profundidad y complejidad, que posee el detector a la hora de localizar un posible ataque.
- *Fiabilidad*, definida como el grado de certeza que nos ofrece el detector ante el aviso de un posible ataque.

Por lo contrario, a pesar del gran desarrollo de estos detectores nos encontramos que están muy lejos de que su capacidad por si mismos sea aceptable. En la actualidad con la utilización de los detectores por separado para afrontar estas dos propiedades, nos encontramos con los dos principales problemas de la detección:

- *Falsos Positivos*, La falta de fiabilidad en los detectores es el causante de los falsos positivos, posibles ataques detectados que realmente no corresponden con ataques reales.

- *Falsos Negativos*, La incapacidad de detección implicaría que un ataque es pasado por alto “falta de sensibilidad”.

PostProceso.

Una vez realizado el preproceso por los diferentes detectores y haber enviado la información al sistema central para realizar la colección, El postproceso es un conjunto de mecanismos capaz de mejorar la sensibilidad y fiabilidad de la detección, disminuyendo los falsos positivos por descarte de estos, o descubrir patrones más complejos que los detectores han sobrepasado para disminuir los faltos negativos.

El postproceso se puede dividir en tres métodos:

- *Priorización*, Todas las alertas recibidas se priorizan mediante un proceso de contextualización desarrollado a través de la definición de una Política topológica de la red. De esta manera se consigue descartar falsos positivos.

- *Valoración de Riesgo*, Cada evento será valorado respecto del riesgo que implica, dependiendo del valor del activo al que el evento se aplica, la amenaza que representa el evento y la probabilidad de que este evento ocurra.

- *Correlación*, Donde se analizaran un conjunto de eventos relacionados para obtener una información de mayor valor. De esta manera aumentaremos la sensibilidad de la red.

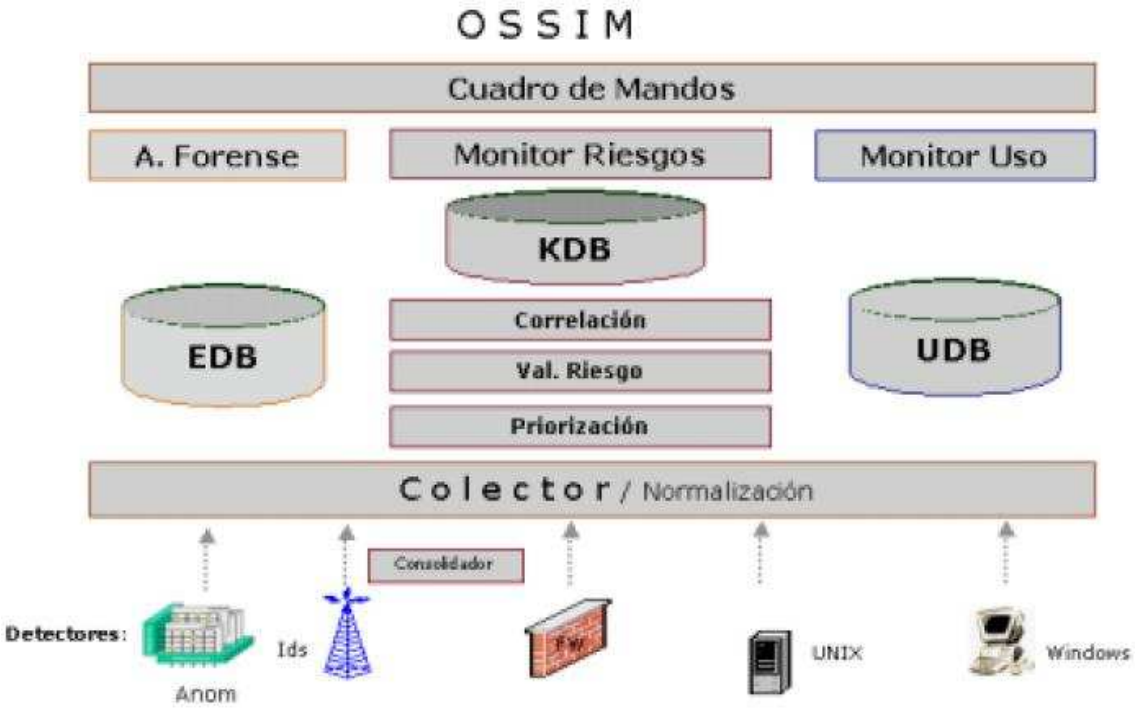
Tras el PostProceso se obtendrá como resultado las “Alarmas”, donde una alarma normalmente será el conjunto de varias alertas producidas. En este punto se habrá obtenido normalmente un mayor grado de sensibilidad, permitiendo localizar patrones más complejos y ofrecer un mayor grado de fiabilidad.

4. Arquitectura.

La arquitectura de Ossim se puede diferenciar en dos partes, una parte se realiza a través de una arquitectura distribuida y la otra sobre una arquitectura centralizada, en ellas se desarrolla los dos momentos diferentes del proceso:

- Preproceso: Que se realiza en los propios monitores y detectores distribuidos.
- Postproceso: Que se realiza en el servidor centralizado.

En el siguiente dibujo se representa de una forma detallada la funcionalidad de cada uno de los dos procesos.



2. Arquitectura Ossim.

Ossim utiliza tres bases de datos heterogéneas para los distintos tipos de datos almacenados:

- EDB base de datos de eventos, la más voluminosa pues almacena todos los eventos recibidos desde los detectores y monitores.
- KDB base de datos del Framework, en la cual se almacena toda la información referente a la red y la definición de la política de seguridad.
- UDB base de datos de perfiles, almacena todos los datos aprendidos por el monitor de perfiles.

5. Funcionalidad.

En el siguiente punto se define cada una de las tres capas que compone Ossim. Cada una de ellas se descompone en varios niveles, formando nueve niveles que serán descritos uno a uno.

A continuación se muestra una representación gráfica de las tres capas con sus nueve niveles intermedios:

3. Representación niveles Ossim.

Se empezará describiendo los niveles más bajos y se irá subiendo uno a uno hasta describir los nueve niveles que componen el sistema Ossim.

5.1 Detectores de Patrones.

Se les denomina a las aplicaciones capaces de escuchar el tráfico de la red, en busca de patrones malignos definidos a través de firmas o reglas, y producir eventos de seguridad.

Las aplicaciones más comunes son los sistemas de detección de intrusos *“IDS”*. Se basan en el análisis detallado de tráfico de la red, comparando el tráfico con las firmas de ataques conocidos o reglas de comportamientos sospechosos, como puede ser el escaneo de puertos. Los IDS analizan tanto el tipo de tráfico como el contenido y el comportamiento de los paquetes de la red.

Cualquier otro dispositivo de la red, como puede ser un router, firewall, o el mismo sistema operativo de los hosts, tienen la capacidad de detectar patrones en la red como puede ser un escaneo de puertos, intentos de spoofing, o posibles ataques por fragmentación, cada uno de ellos tiene su propio log de seguridad capaz de alertar de

posibles problemas en la red, que podremos recolectar para su posterior tratado en los motores de correlación.

Detectores de patronos incluidos en Ossim.

Ossim integra varios detectores de patrones de código abierto. El detector más comun en Ossim es el Snort (NIDS, Network Intrusión Detection System), incluye varios preprocesadores de detección de ataques y anomalías.

Otros detectores incluidos son Snare y Osiris (HIDS Host Intrusión Detection System), instalados en los sistemas monitorizados de la red.

5.2 Detectores de anomalías.

Los detectores de anomalías gozan de una capacidad de detección mucho más compleja e innovadora que la de los detectores de patrones. En este caso al sistema de detección no tenemos que especificarle mediante reglas que es un comportamiento bueno o malo, sino que es capaz de “aprender” por sí solo y alertar cuando un comportamiento difiere del comportamiento normal.

Está técnica provee una solución para controlar el acceso de usuarios privilegiados y ataques internos, como puede ser un empleado desleal, o simplemente hacen un mal uso de los recursos y servicios de la empresa.

Casos en los que los detectores de anomalías son útiles:

- Nuevos ataques para el que aún no existen firmas, puede definir anomalías obvias para un detector de anomalías.
- Un gusano que puede haber sido introducido desde la red interna, malware, ataque de spam, pueden generar un número de conexiones anómalas que son fáciles de detectar.
- Uso de servicios con origen y destino anormales.
- Uso en horarios anormales.
- Exceso de tráfico o de conexiones (programas P2P).
- Cambios de sistemas operativos, ips, macs.

Estas aplicaciones pueden generar un número de nuevas alertas elevado, que podrían empeorar la visibilidad del estado de la red por si solas, pero si tomamos estas alertas como información que acompaña a resto de alertas, los niveles superiores realizarán una correlación más fiable y les permitirá detectar nuevas anomalías.

Detectores de anomalías incluidos en Ossim.

Ossim integra una amplia gama de detectores de anomalías:

- *Spade* detecta conexiones no usuales por puertos y destinos utilizados. Usado para mejorar el reconocimiento sobre ataques sin firma.
- *Aberrant Behaviour* plugin para *Ntop* aprende el uso de parámetros y alerta cuando dichos parámetros se salen de los valores esperados.
- *ArpWatch* utilizado para detectar cambios de mac.
- *Pof* utilizado para detección de cambios de sistema operativo.
- *Pads y Nmap* Utilizado para detectar anomalías en los servicios de red.

5.3 Sistema de Colección y Normalización.

El proceso de colección y normalización se encarga de unificar todos los eventos de seguridad provenientes de cualquier sistema de la red en una única consola y formato.

La recolección de datos se puede hacer de dos formas distintas en el sensor. Se puede enviar los datos desde el equipo analizado usando protocolos nativos del equipo al gestor central, o instalando agentes en el equipo analizado que recopilan la información en el host y la envían seguidamente. Ossim normalmente no utiliza agentes y utiliza las formas de comunicación naturales de los sistemas.

La normalización implica la existencia de un “parser” o traductor que conozca los tipos de formatos de alertas de los diferentes detectores, capaz de homogeneizar el tratamiento y la visualización de todos estos eventos en una única base de datos “EDB”.

EDB, es la base da datos que Ossim utiliza para almacenar todos los eventos que colecciona, es la base de datos más voluminosa.

De esta forma se podrá visualizar en la misma pantalla y con el mismo formato los eventos de seguridad de un determinado momento, ya sean del Router, firewall, IDS o de cualquier host.

Al tener centralizado en la misma base de datos todos los eventos de la red se podrá desarrollar procesos a niveles superiores que permitan detectar patrones más complejos y distribuidos.

5.4 Políticas de priorización.

La prioridad definida para una alerta será dependiente de la topología de la red, inventario de cada máquina y del rol que estas desempeñan en la organización. Si una alerta que se refiere a un ataque al servicio IIS de Microsoft, llega a una máquina con sistema operativos Unix y servidor Apache, la alerta debe de ser despriorizada. En cambio, si existe una conexión sospechosa de un usuario sobre un servidor, el sistema debe priorizar la alerta dependiendo de la ubicación del usuario y del uso de la conexión.

El proceso de priorización de alertas se realiza mediante contextualización, es decir la valoración de la importancia de una alerta depende del escenario de la red. Este escenario está descrito en una base de conocimientos sobre la red formada por:[2]

- Inventario de Máquinas y Redes (ip, mac, sistema operativo, servicios, etc).
- Políticas de Acceso (desde donde a donde está permitido o prohibido).

Todos estos parámetros son alojados en la base de datos “KDB”, que es la base de datos que Ossim utiliza para parametrizar el framework. De esta forma el sistema conocerá la topología de la red, características de las maquinas y las políticas de seguridad definidas.

A través de la valoración de alertas se realizará una de las partes más importantes del filtrado de alertas recibidas por los detectores. Desde el framework del sistema podremos configurar las siguientes características:

- Política de Seguridad
- Inventario de las maquinas de la red.
- Valoración de activos.
- Valoración de amenazas.
- Valoración de fiabilidad de cada alerta.
- Definición de alarmas.

Para que el proceso de priorización sea efectivo se debe realizar una continua y detallada especificación de la situación de la organización.

5.5 Valoración de Riesgos.

La arquitectura de Ossim ha sido diseñada para que todas las decisiones que se tomen a la hora de actuar sobre una alerta, se apoyen en función de la valoración de riesgos calculada, por eso es necesario comprender el proceso de cálculo de valor de riesgo que ossim realiza sobre cada evento.

[2] *Se define como la prioridad que se le asignará a cada alerta dependiendo del escenario de la red, inventario de la maquina, la política descrita y el rol que desempeña.*

La importancia que se debe dar a cada evento será dependiente de los tres factores siguientes:

- El valor del activo “equipo” implicado sobre el evento.
- La amenaza que representa el evento o cuanto daño puede hacer al activo implicado.
- La probabilidad de que este evento ocurra.

Riesgo intrínseco (visión tradicional).

La valoración del riesgo intrínseco o riesgos latentes, es referido al riesgo que debe de soportar una organización por el hecho de poder desarrollar su negocio y las amenazas circunstanciales que existen sobre estos activos.

Con ello se mide el valor del posible impacto de una amenaza sobre un activo con la probabilidad de que esta ocurra.

Riesgo Instantáneo.

Dada la capacidad que Ossim ofrece para el trabajo en tiempo real, se podrá medir el riesgo asociado al esquema actual en tiempo real. En este caso el valor del riesgo se medirá como el daño que produciría el evento y la probabilidad de que esté ocurriendo en este momento la amenaza.

Está probabilidad, derivada de la imperfección de los detectores (falsos positivos), y representará el grado de fiabilidad de estos en la detección de una posible intrusión.

Por ello, el valor de riesgo instantáneo producido por la recepción de una alerta, dependerá del daño que produciría el ataque, la probabilidad de que este ocurra y la fiabilidad que el detector proporciona.

Ossim calculará el riesgo instantáneo de cada evento recibido, que será la medida objetiva que se utilizará para valorar la importancia que un evento puede implicar, así poder descartar falsos positivos que las organizaciones reciben miles al día, y a través de estas medida se valorará la necesidad de actuar.

Ossim incluye un Monitor de Riesgos (descrito posteriormente), que valorará el riesgo acumulado en un rango de tiempo, sobre redes y grupos de trabajo relacionados en un evento.

5.6 El motor de Correlación.

La función de correlación se puede definir como un algoritmo que realiza una operación a través de unos datos de entrada y ofrece un dato de salida.

Los sistemas de correlación suplen la falta de necesidad que hoy en día existe en la mayoría de las redes, aumentando la sensibilidad, fiabilidad, escalabilidad y la visibilidad limitada de cada detector.

Se podría pensar que instalar un sistema único centralizado capaz de localizar toda la información de la red resultaría más fácil. Pero para ello se necesitaría una visibilidad completa desde un punto único de la red y una capacidad de almacenamiento y de memoria ilimitada.

El motor de correlación desarrollado en Ossim, se encarga de comprobar cada uno de los eventos recibidos y busca evidencias o síntomas que prueben la veracidad de un ataque o si se trata de un falso positivo.

En Ossim se ha desarrollado un modelo de correlación tan ambicioso que tiene la capacidad de:

1. Desarrollar patrones específicos para detectar lo conocido y detectable.

2. Desarrollar patrones ambiguos para detectar lo desconocido y no detectable.

3. Poseer una máquina de inferencia configurable a través de reglas relacionadas entre sí capaz de describir patrones más complejos.

4. Permitir enlazar Detectores y Monitores de forma recursiva para crear cada vez objetos más abstractos y capaces.

5. Desarrollar algoritmos que ofrezcan una visión general de la situación de seguridad de la red.

El motor de correlación de Ossim se alimenta mayoritariamente de dos elementos claves en la red de datos, como son:

- *Los monitores*, Proporcionan normalmente indicadores del estado.

- *Los detectores*, Proporcionan normalmente alertas.

Como salida el motor de correlación podrá devolver tanto una alerta como un indicador, con un grado de fiabilidad mayor.

Métodos utilizados en el proceso de correlación.

El proceso de correlación se rige mediante tres métodos heterogéneos pero con un mismo objetivo.

- **Correlación mediante secuencia de eventos (Correlación lógica).** Se centra en buscar ataques conocidos y detectables, relaciona a través de reglas que implementarán una máquina de estados, los patrones y comportamientos conocidos que definen un ataque.

- **Correlación mediante algoritmos Heurísticos.** Este método detectará situaciones sin conocer patrones y comportamientos que definen un ataque.

Implementa funciones que mediante funciones heurísticas intentará descubrir situaciones de riesgo que se alejan del comportamiento cotidiano, intentará suplir la incapacidad del método anterior, además se podrá obtener una visión general del estado de seguridad de la red.

- **Correlación mediante inventariado.** Los ataques recibidos tienen siempre como objetivo un determinado “sistema operativo, servicio especifico, etc..”. Con el inventario de la red podremos descartar falsos positivos a maquinas que no cumplen dichas características y priorizar las maquinas de mayor riesgo como los servidores.

Correlación mediante secuencia de eventos (Correlación lógica).

La correlación lógica se implementa a través del panel de secuencias, en el cual se definen reglas que representan árboles de nodos de condiciones lógicas (secuencia de eventos). Este tipo de estructuras se conoce como árbol de decisión (and/or tree) utilizados en sistemas de inteligencia artificial.

La variable de fiabilidad crece según el motor de correlación avanza a través de los nodos (eventos) cumpliéndose las condiciones de cada uno de ellos. Cuando se cumple la condición de un nodo, el motor de correlación salta al primer hijo, si la condición del hijo no se cumple, se buscará el hermano (nodo en el mismo nivel de dependencia del nodo anterior). De esta manera se implementa la operación “AND” en el eje “Y” y la operación “OR” en el eje “X”. Cuantas más evidencias tengamos, más posibilidades hay de que sea real el ataque.

El motor de correlación desarrollado en Ossim goza las siguientes características:

- Fuente híbrida, acepta tanto patrones procedentes de detectores como indicadores procedentes de monitores.

- Posibilidad de definir orígenes y destinos variables.

- Define una arquitectura recursiva, permite que las alertas de salida se tomen como nuevos eventos que se pueden volver a correlar por otras reglas. Cada regla genera un nueva alerta con una prioridad específica, esta alerta de salida se puede tomar como un evento más de entrada (probablemente con una mayor fiabilidad), creando la recursión y posibilitando la implementación de n niveles de correlación.

- Se puede definir el nivel de prioridad y fiabilidad de las nuevas alertas.

- Utiliza variables “elásticas” o capaces de medir el grado de prioridad y fiabilidad (ej: Denegación de servicios: total -> prioridad grav, 50% -> prioridad media, 15% -> prioridad baja)

- Define una arquitectura distribuida jerarquizada, que permite definir n niveles de correlación en una topología distribuida.

Correlación mediante algoritmos Heurísticos.

Ossim implementa un algoritmo heurístico de correlación por acumulación de eventos en un determinado tiempo, con el objetivo de obtener una imagen del estado general de seguridad de la red.

Desde el cuadro de mando obtendremos una visión a alto nivel de las situaciones de riesgo sin conocer en ningún momento detalle de las características del problema, pero con una rápida y clara visibilidad. Se mostrará el nivel acumulado de riesgo, que será sensible a la cantidad de riesgo acumulado en una ventana de tiempo. Irá subiendo proporcionalmente según la cantidad y la prioridad que tengan los eventos recibidos, e irá bajando con el paso del tiempo en caso de no recibir nuevos eventos. Se dará máxima prioridad a los eventos definidos como “riesgo instantáneo”.

Este método de correlación quiere suplir con un punto de vista opuesto a la correlación mediante secuencias de eventos, donde intenta caracterizar al máximo nivel de detalle los posibles ataques

El objetivo de este método es:

- Ofrecer una visión general rápida de la situación.

- Detectar posibles patrones que al resto de sistemas de correlación puedan pasar por alto, ya sea por ser ataques desconocidos o por falta de capacidad.

Compromiso and Attack Level Monitor (CALM).

Es un algoritmo de valoración por acumulación de eventos con recuperación en el tiempo. Recibe de entrada un alto volumen de eventos y como salida muestra un único indicador del estado general.

La valoración del riesgo se puede realizar tanto a una única maquina de la red, como a un grupo de máquinas, e incluso a un segmento de la red que nos interese monitorizar. La valoración se realiza dependiendo de dos variables:

Acumulación de eventos.

La acumulación se realiza a través de la suma del riesgo instantáneo de cada evento en dos variables de estado:

- **El nivel de compromiso “C”.** Mide la posibilidad de que una máquina se encuentre comprometida, ofrece la evidencia de que ha habido un ataque y ha tenido éxito.

- **El nivel de ataque “A”.** Mide el posible riesgo debido a los ataques recibidos, ataque que podrá o no tener éxito.

La importancia de cada una de las dos variables será dependiente de la situación de la máquina. Por ejemplo, una maquina alojada en una zona desmilitarizada DMZ, expuesta a multitud de ataques obtendrá un alto valor de nivel de ataque “A”, el cual será una situación normal, pero el mínimo nivel que indique un cierto compromiso “C” que pueda hacer pensar que existe un ataque consolidado debe de ser alertado y revisado.

Al contrario, hay casos que una máquina que por su función general, provoca anomalías en la red como un sniffer de seguridad, un servicio con puertos aleatorios, tendrá un nivel de compromiso “C” alto y un nivel de ataque “A” bajo.

Acumulación en el tiempo.

El algoritmo CALM está pensado para la monitorización en tiempo real, ya que el interés de Ossim es saber la valoración de riesgo en un espacio de tiempo cercano, el algoritmo tiene una memoria a corto plazo primando los eventos más recientes y caducando los más antiguos.

El nivel de riesgo se irá rebajando en función de la recuperación en el tiempo. El sistema irá rebajando el valor de los niveles C y A de forma periódica de cada máquina a medida que el tiempo vaya pasando y no se reciban nuevos eventos.

Correlación mediante inventariado.

Todo ataque tiene como objetivo un determinado sistema operativo o servicio especificado. La correlación de inventario comprueba si el sistema atacado usa ese sistema operativo o servicio objetivo del ataque. Si lo usa, podremos determinar que existe riesgo, por lo contrario, se puede confirmar que el evento para dicha maquina es un falso positivo.

Este tipo de correlación depende de la fiabilidad del inventario, Ossim incorpora además del inventario manual, un método de inventario automático.

Inventario automático.

Se realiza en los sensores con detectores pasivos que permiten de forma pasiva ver el tráfico de la red. También se realiza de forma centralizada desde el servidor, mediante analizadores de red que de forma activa encuentran hosts y servicios.

Ambos métodos automáticamente rellenan la base de datos de inventarios con la siguiente información:

- Tipo de sistema operativo y versión.
- Tipo de servicio y versión.

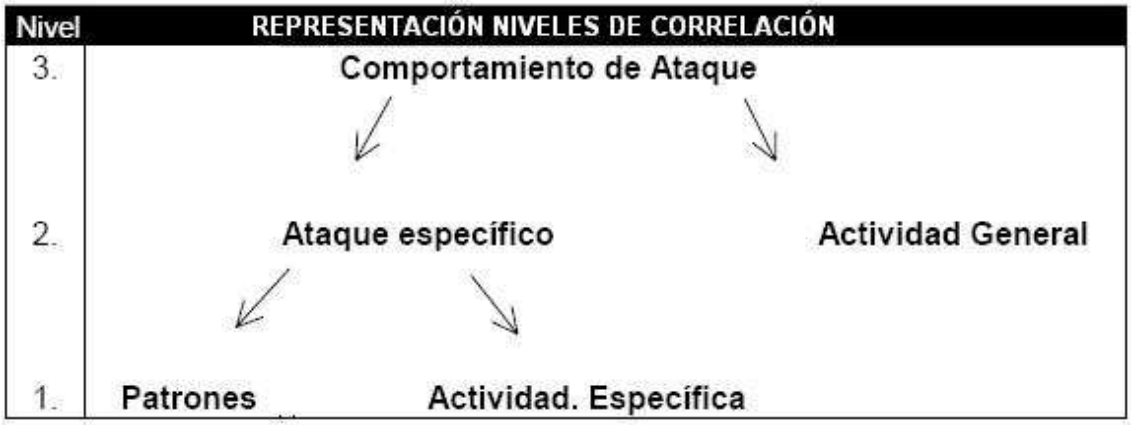
- Dirección IP y MAC.

Ossim realiza el inventariado mediante el uso de aplicaciones de código abierto como:

- Nmap, analizador de red sin necesidad de un agente.
- POf, detector pasivo de sistema operativo sin necesidad de agente.
- Pads detector pasivo de servicios sin necesidad de agente.
- ArpWatch, detector pasivo de paquetes ARP “cambios de mac y de ip” sin necesidad de agente.
- OCS, agente desplegado en los sistemas monitorizados.

Niveles de correlación.

Debido a la recursividad que el modelo de Ossim ofrece se puede crear una jerarquía de niveles casi infinita. A continuación se mostrará un ejemplo con un mínimo de 3 niveles de correlación que se define a continuación:



4. Representación niveles de correlación.

Nivel 2. Ataque específico.

Desde el nivel más bajo el motor de correlación será alimentado por los detectores de patrones y será capaz de procesar estos eventos para detectar nuevas alarmas. Al mismo nivel, también se podrá detectar nuevos eventos por una actividad específica en la red, proporcionando alarmas con una determinada prioridad y fiabilidad. Ahora bien, si desde un nivel más alto se correlacionan de forma cruzada estas alarmas, se podrá obtener una mayor fiabilidad de nuestras alarmas. Se buscará para una firma detectada evidencias que demuestren que se está produciendo un ataque o que es un intento fallido.

Esta correlación es la que marcará la diferencia a la hora de limitar falsos positivos y priorizar ataques reales en el motor de correlación.

Ejemplo con la detección de un Caballo de Troya, las operaciones que un IDS es capaz de detectar son: *“Connect, active, get Info, access, server2client_flow, client2server_flow, response, traffic_detect”*.

La detección de una operación *connect* no es una información de gran valor, en entornos perimetrales se reciben decenas al día, pero si a continuación detectamos cualquier otra operación de las anteriores mencionadas y sobre todo alguna de respuesta, deberemos enviar una alerta con prioridad alta. Pero si además, tras haber obtenido una alerta, se comprueba la actividad de los puertos sobre los que el caballo de Troya opera y resultan tener actividad, se puede confirmar que el intento de conexión ha tenido éxito, obteniendo una alerta con una gran fiabilidad y una prioridad alta.

Nivel 2. Actividad General.

La localización de estos ataques no conocidos será gracias a la generación de actividad anómala por parte del atacante, para ello se monitoriza el uso general de cada usuario, almacenando parámetros como puertos, servicios, tráfico e incluso el horario.

De esta forma se detectará comportamientos sospechosos, con nivel menor de precisión que en la detección de ataques específicos, que muchas veces no será un ataque si no un problema de la red o un mal uso por parte de los empleados.

Nivel 3. Comportamiento de Ataque.

El tercer nivel de correlación se alimenta de las alertas generadas por la correlación de varios ataques específicos o actividades anómalas localizadas, proporcionando un grado de fiabilidad aún mayor.

Pero la correlación de los nuevos niveles no solo reciben como entrada eventos procedentes del nivel inferior, al contrario, se podrá mezclar según convenga. La caracterización de cada nivel debe de localizar patrones de comportamiento que caractericen cual es el objetivo, el camino trazado, el comportamiento del atacante.

5.7 Monitores.

Ossim realiza una monitorización de la red esencial para un sistema de seguridad, ya que sin ella un administrador de seguridad estará ciego cuando ocurran eventos, sin poder distinguir la actividad anómala de la normal.

Ossim realiza diferentes tipos de monitorización:

- **Monitor de Riesgos (RiskMeter).** Representa los valore producidos por el algoritmo CALM, valores que miden el nivel de riesgo de compromiso “C” y el de ataque “A” procedentes de la recepción de alertas que indican que una determinada maquina ha sido comprometido o está siendo atacada.

- **Monitor de Uso.** Ofrece datos generales de la máquina, como el número de bytes que transmite al día.

- **Monitor de Perfiles.** Ofrece datos específicos del uso realizado por el usuario y permite establecer un perfil, (ej: uso de correo, pop y http, perfil de usuario normal), estos datos se obtienen de la base de datos de perfiles “UDB”.

- **Monitor de Sesión.** Permite ver en tiempo real las sesiones que está realizando el usuario. Ofrece una foto instantánea de la actividad de una maquina en la red.

- **Monitor de Caminos.** Ofrece una representación en tiempo real de los caminos trazados en la red entre las diferentes máquinas que interactúan entre ellas en un intervalo de tiempo. El monitor obtiene sus datos de dos de los monitores descritos anteriormente, el de sesiones le proporciona cada uno de los enlaces del momento, y el monitor de riesgo le proporciona el nivel de riesgo de cada máquina para representar cada camino con un color diferente y calcular el riesgo agregado. La monitorización se puede realizar únicamente dibujando las sesiones tcp o dibujando tanto udp como tcp y icmp lo que puede implicar un mapa de red enredado.

- **Monitor de Disponibilidad.** La información de disponibilidad es importante para detectar ataques de denegación de servicios. Ossim incluye el plugin “Nagios” capaz de chequear y mostrar la disponibilidad o no de servicios y equipos en la red.

- **Monitorización Personalizada.** Existe un plugin parametrizable que permite crear monitores personalizados, que extraen cualquier parámetro que se quiera recopilar, filtrar y enviar al motor de correlación para ser procesado.

5.8 Consola forense.

La consola forense es un frontal Web que permite la consulta a toda la información almacenada en el colector.

Esta consola es un buscador que ataca a la base de datos de eventos “EDB”, y permite al administrador analizar a posteriori y de una forma centralizada los eventos de seguridad de todos los elementos críticos de la red.

Al contrario que el monitor de riesgos, esta consola permite profundizar al máximo detalle sobre cada uno de los eventos ocurridos en el sistema.

5.9 Cuadro de mandos.

La última de las funcionalidades ofrecidas por Ossim es el Cuadro de Mandos, donde se podrá configurar una visión a alto nivel del estado de seguridad de la red.

El cuadro de mandos monitorizará una serie de indicadores definidos que medirán el estado de seguridad de la organización, definiendo umbrales que debe cumplir la organización.

Es la principal herramienta para saber en todo momento que ocurre en la red, mostrando la información más concisa y simple posible. A través de él se enlazara con cada una de las herramientas de monitorización para profundizar sobre cualquier problema localizado.

Como ejemplo se podrían visualizar los siguientes datos:

- Monitorización permanente de los niveles de riesgo de las principales redes de la organización.
- Monitorización de las máquinas o subredes que superen el umbral de seguridad.
- Monitorización de perfiles que superen los umbrales por:
    - Uso de tráfico
    - Uso de servicios críticos.
    - Cambios en configuración.
    - Uso de servicios anómalos.
- Monitorización de aquellos parámetros de la red o niveles de servicio que superen el umbral establecido:
    - Número de correos, virus, accesos externos.
    - Latencia de servicios, uso de tráfico por servicios.

6. Flujo de los datos.

Para resumir la integración de cada uno de los niveles descritos anteriormente, se hará un recorrido del flujo desde la generación de un evento.

1. Los eventos son generados por los detectores o monitores, ya sea por la detección de un patrón o una anomalía.

2. Los eventos son procesados en caso necesario por los *consolidadores* antes de ser enviados (encargados de enviar la información agrupada para ocupar el mínimo ancho de banda).

3. Los eventos son recibidos por el colector a través de diferentes protocolos abiertos de comunicación.

4. El *parser* se encarga de normalizarlas y guardarlas si procede en la base de datos de eventos “EDB”.

5. El *parser* se encarga de cualificar los eventos determinando su prioridad según la política de seguridad definida y los datos sobre el sistema atacado localizados en el inventario de sistemas.

6. El *parser* valora el riesgo instantáneo que implica la alerta y en caso de ser necesario envía una alarma al Cuadro de Mandos.

7. Los eventos son procesados por el motor de correlación para generar alarmas, que a su vez lanzará nuevos eventos con una información más completa y fiable al *parser*.

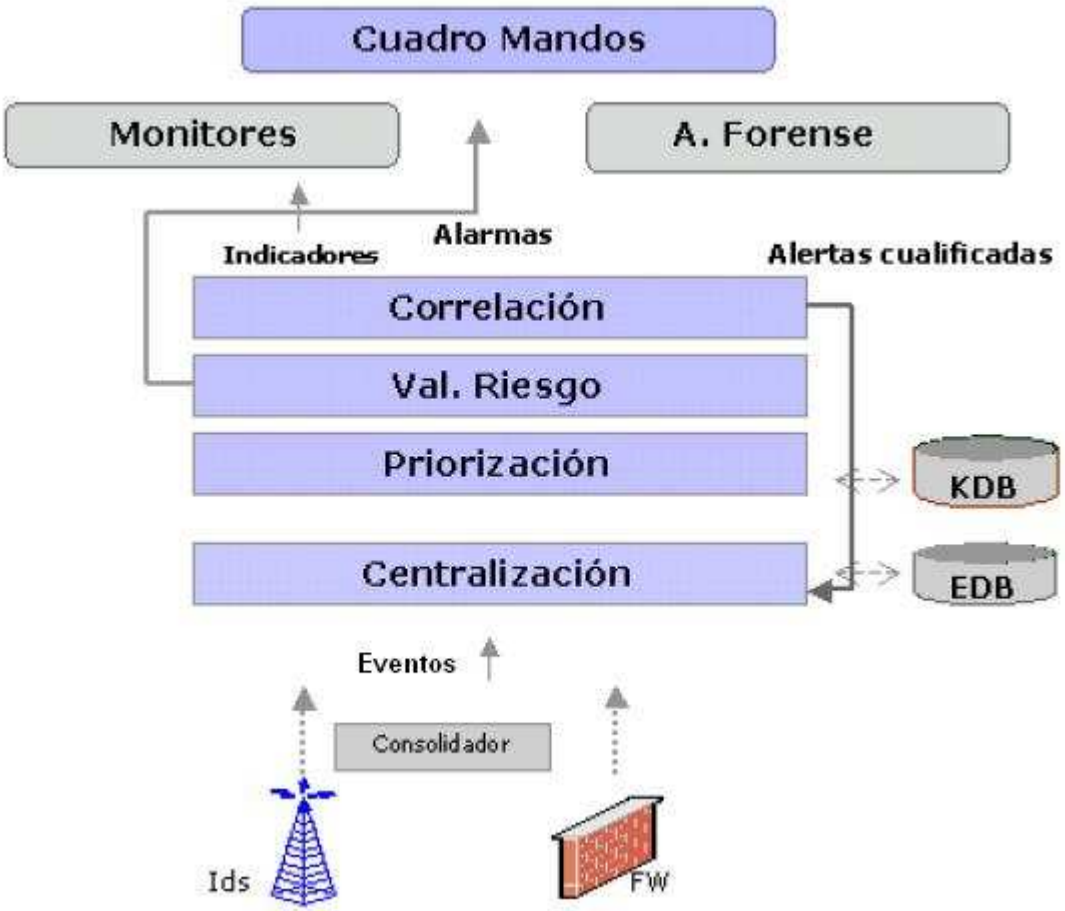
8. El monitor de riesgos visualizará la situación de cada uno de los índices de riesgo según han sido calculados por el algoritmo CALM.

9. El cuadro de mandos mostrará las alarmas más recientes.

10. El administrador podrá desde el cuadro de mandos enlazar y visualizar a través de la consola forense todos los eventos ocurridos en el momento de la alarma.

11. Podrá además comprobar el estado instantáneo de la máquina a través de los monitores de uso, perfiles y sesiones.

El siguiente gráfico representa el flujo de los datos:



5. Representación Flujo de datos.

7. Configuración de Ossim.

Ossim está desarrollado para funcionar sobre el sistema Linux, nos ofrece la posibilidad de ser instalado sobre las distribuciones (Debian 4.0 y Fedora Core 3). No obstante, también ofrece una imagen “VMWare”, donde viene instalado la última distribución de Ossim sobre Debian 4.0. Para nuestra investigación se ha optado por la utilización de la imagen ofrecida, ya que ellos mismos recomiendan el uso de ella.

Una vez instalado Ossim debemos de configurarlo con el mayor nivel de detalle posible, ya que cuanta más información le proporcionemos, aumentará tanto la fiabilidad como la sensibilidad de las alertas producidas. Para ello Ossim ofrece una interfaz web desde donde podremos gestionar y visualizar el estado de la red.

6. Identificación inicio interfaz web Ossim.

7.1 Mapa de la red local analizada.

Para el análisis de la plataforma Ossim se ha utilizado mi red de área local propia, lo que ha sido un inconveniente ya que el número de máquinas y el volumen de carga de trabajo sobre la red es muy bajo. La red consta de los siguientes equipos:

Nombre	Dirección IP	Descripción
RouterOno	192.168.1.1	Pasarela a Internet, servidor DHCP, DNS, etc...
SobreMesaAlzira	192.168.1.10	Maquina de trabajo, servidor VPN
SonyMobile	192.168.1.11	Maquina de trabajo, portátil
SobreMesaSilla	192.168.1.12	Maquina de trabajo exterior conectada por vpn
MediaCenter	192.168.1.13	Centro multimedia, servidor NAS, P2P
vmOssim	192.168.1.69	Servidor virtual Ossim

7. Tabla inventario equipos de la red de área local.

Como se ha descrito a modo resumen en la tabla anterior, la red consta de cinco equipos donde cada uno de ellos emplea un rol diferente dentro del escenario de la lan.

7.2 Inventario de la corporación.

El primer paso a realizar para la configuración del servidor Ossim es el inventario de la estructura de la corporación, deberemos detallar tanto las subredes configuradas como los equipos alojados en cada subred.

Para que esta configuración se realice de forma más amena, Ossim utiliza la aplicación nmap que nos ayudará a encontrar de forma automática los equipos pertenecientes a una red, y obtendrá con un buen nivel de detalle los servicios y el sistema operativo utilizado por cada uno de ellos.

Para nuestro caso se ha configurado una sola red que abarcará todos los equipos mencionados anteriormente.

8. Inventario de redes.

Cabe destacar de la configuración los umbrales C y A y la columna Activo. Estas tres opciones son comunes a todos los niveles de inventario, y son útiles para calcular el valor de riesgo que existe sobre el mismo.

El Activo, se utiliza para argumentar la importancia del equipo en la red, (dependiendo del rol que desempeñe el mismo). Es un valor entre cero y cinco, siendo cinco el valor más alto (lo que indicará que la máquina desempeña un gran papel en la red, como puede ser un servidor). La asignación de un alto valor de los activos a todos los equipos no es un uso eficaz del valor, ya que como más adelante justificaremos las alarmas[3] son generadas por una formula donde el Activo juega un gran papel, y si asignáramos a todas las máquinas un valor alto, obtendríamos un considerable número de alarmas falsas.

Umbral C, indica el nivel máximo de carga permitido para el Compromiso aceptado, Si el nivel es superado debido a la acumulación de varias alertas, se visualizará en el panel de control, cada alerta tiene detallado un nivel de compromiso, que será el valor que afecta en particular a cada equipo.

Umbral A, indica el nivel máximo de carga permitido para el Ataque, del mismo modo que el nivel de compromiso cada alerta tiene definido un nivel de ataque, que será el valor que afecta en particular a los equipos afectados.

[3] Ver formula en el apartado 7.9 Evaluación del riesgo

Estos niveles deben de ser reconfigurados dependiendo del volumen de la red hasta encontrar el nivel óptimo, esto es cuestión de práctica y experiencia el umbral varía para cada medio ambiente.

Una vez definidas las redes deberemos buscar los equipos alojados en cada una de ellas, para ello desde la sección de Herramientas, utilizaremos el *FrontEnd* desarrollado de la aplicación *Nmap* e integrado en Ossim, que buscará para una red definida los equipos pertenecientes a ella y los servicios ofrecidos.

9. Análisis de red con Nmap.

Desde la sección de *Política* podremos visualizar los equipos hallados por *Nmap* y personalizar la configuración de cada uno de ellos con más detalle (como puede ser importante detallar el umbral de compromiso y ataque permitido para cada equipo).

10. Inventario de Equipos.

Desde la misma sección de política podremos crear tanto grupos de equipos, como grupos de redes para poder monitorizar los equipos/redes agrupados entre ellos según nos convenga.

Para nuestro caso se han definido dos grupos de equipos, en uno alojaremos los equipos de uso diario y en el otro se ubicarán tanto el router como el servidor de Ossim, ya que estos dos deben de ser más sensibles a cualquier posible ataque o compromiso.

11. Inventario de Grupos de Equipos

El umbral de compromiso y ataque de cada grupo debe de ser proporcional a la suma de cada uno de los equipos ubicado en el, ya que cuando monitoricemos cada grupo, este nos indicará como nivel de compromiso y ataque, la suma de los niveles de cada una de las máquinas pertenecientes a un grupo determinado.

La configuración de grupos de red en nuestro caso era irrelevante, ya que solo tenemos una red definida, pero para una corporación más grande, existe la posibilidad de crear grupos de redes, donde podremos definir el umbral de importancia que tienen en común.

7.3 Sensores.

Ossim utiliza los sensores para recopilar datos de la red. Estos están distribuidos por la red y envían la información al servidor para realizar la colección de datos.

Desde la sección de sensores dentro del grupo *Política* podemos gestionar los sensores existentes en la red y agregar nuevos sensores. El primer cuadro muestra en una simple visión a modo resumen el conjunto de sensores configurados activos y el total. Desde el segundo cuadro, ya con un nivel más completo de detalle podemos gestionar los sensores añadiendo, modificando o eliminando cada uno de ellos.

12. Inventario de Sensores.

Al insertar un sensor debemos de configurar los siguientes parámetros:

- **IP**. Dirección ip de la máquina con un agente Ossim. La dirección ip debe e ser accesible desde el servidor Ossim.

- **Nombre de la máquina**. Es el nombre del equipo, no tiene porque ser el nombre real del equipo, únicamente es el nombre con el que Ossim identificará al sensor.

- **Prioridad**. Determina la prioridad del sensor dentro de equipo. Una máquina puede tener múltiples sensores y el que más alta prioridad tenga es el primer sensor vinculado con Ossim.

- **Puerto**. Puerto de destino del agente para comunicarse con el servidor Ossim.

- **Activo**. Indica si el sensor está activo o no. Los sensores activos son aquellos que tiene una comunicación con el servidor Ossim.

- **Descripción**. Texto para describir el sensor. Utilizado para hacer una descripción más detallada cuando el nombre no es lo suficientemente descriptivo.

7.4 Servidores.

Como hemos mencionado anteriormente en la funcionalidad, Ossim es un sistema distribuido y podemos tener varios servidores corriendo en la misma red, desde esta sección podremos realizar el inventario de los servidores Ossim (agregando nuevos servidores hijos al servidor maestro) o ver el estado del inventario.

13. Inventario de Servidores.

La sección servidores muestra tanto el estado del servidor maestro si esta o no activo, como el estado y el total de servidores hijos enlazados.

7.5 Puertos.

Cuando realizamos el inventario de las máquinas, Ossim detecta los puertos activos en cada una de ellas y utiliza la nomenclatura RFC para describir cada uno de ellos, sin embargo muchos de estos puertos no están definidos, por lo que realizar un inventario de los puertos utilizados en nuestra red, nos ayudará a tener una visión más rápida y eficaz a la hora de leer las alarmas y eventos.

Desde esta sección podremos visualizar todo los puertos o grupos de puertos inventariados y gestionar el inventario añadiendo, modificando o eliminando los mismos.

14. Inventario de Puertos.

7.6 Definición de la Política.

Una gran parte del comportamiento de Ossim se configura a través de esta sección, desde aquí se controla parte de la colección, correlación y priorización de eventos básicos.

Similar a una política de cortafuegos podremos definir que se puede y que no se debe de realizar en la red. La política es esencialmente un grupo de ajustes que manejan el comportamiento y seguridad de la red, desde aquí podemos definir que hacer para ciertos eventos o alarmas con origen y destino conocido.

Esta sección permite ver la política definida y crear nuevas[4] políticas cuando sea necesario.

15. Definición de la Política.

Por defecto el servidor Ossim no tiene ninguna política definida, ya que este tema es totalmente dependiente de cada entorno. Al insertar una nueva política deberemos establecer los siguientes campos de información:

- **Fuente**. Indica la dirección origen de los acontecimientos que queremos registrar. Los eventos que no tienen un objetivo solo tienen dirección de origen, como pueden ser (cambios de sistemas operativos, cambios de mac, nuevo servicio, identificación de una vulnerabilidad, etc.). En este campo podemos seleccionar cualquiera de los equipos definidos anteriormente e incluso seleccionar como origen una de las redes definidas.

- **Destino**. En este campo indicamos el objetivo del evento. En caso de que no tenga un objetivo como hemos visto anteriormente, marcaremos este campo como cualquiera (Ossim insertará como destino la dirección 0.0.0.0). Al igual que en el campo fuente podremos agregar tanto los equipos definidos como las redes.

[4] Cada vez que se modifica la política, es recomendable reiniciar el servidor, ya que se modifican archivos internos críticos.

- **Puerto de destino**. Identifica el puerto de destino del evento, podemos seleccionar cualquiera de los puertos definidos en la sección de puertos, o une vez más, si el evento no tiene un puerto en relación definido podemos definirlo con la etiqueta cualquiera.

- **Prioridad**. Cada uno de los eventos llega con su propia prioridad y fiabilidad, que ha sido generada en el nivel de priorización. Además estos eventos pueden generar nuevas alarmas con sus propios parámetros de prioridad y fiabilidad. Ambos tipos pueden ser perfeccionados desde la creación de políticas, ajustando la prioridad en caso de concordancia entre el evento/alarma y la política definida. El campo prioridad puede tomar valores entre 0 y 5 o el valor “-1” que obtendrá la prioridad del evento o alarma. El valor 0 hace que el evento sea invisible dentro de Ossim ya que no le da ninguna importancia, por lo contrario el valor 5 le da una gran importancia al evento o alarma.

- **Plugin Grupos**. Los plugins son los tipos de eventos que cada detector o monitor envía a Ossim. Estos plugins vienen ya definidos en la instalación de Ossim y cada vez que desarrollan una nueva versión son actualizados. Los plugins se identifican con dos números.

    - **Plugin Id**. Es el identificador padre y se suelen asignar por agentes (snort 1001, nessus 3001, snare 1518, etc.).
    - **Plugin Sid**. Es el subgrupo identificador dentro de cada “Plugin Id” e identifica los diferentes tipos de eventos para cada agente.

  Desde la sección grupo de plugins podemos agrupar los diferentes identificadores por alguna razón que tengan en común (en Ossim vienen configurados cinco grupos donde se distinguen anomalías, Corta fuegos, eventos Windows, eventos Linux y disponibilidad). Al insertar una nueva política podemos definir que grupos son apropiados para la misma.

- **Sensor**. Identifica el sensor que debe generar los eventos asociados a la política. Como en los anteriores casos podemos seleccionar los sensores definidos en la sección de sensores.

- **Rango de tiempo**. Permite definir en que rango de tiempo esta política va a ser valida.

- **Meta**. Para quien va a ser “instalada” esta política. En realidad no se instala en el objetivo seleccionado, pero especifica que el objetivo ha de tener esta política en cuenta. Podemos elegir tanto los sensores agregados como los servidores.

- **Acciones de la política**. Cuando los objetivos de la política son generados por algún evento, se pueden definir algunas acciones a realizar.

    - **Correlacionar eventos***. Indica si debería la equiparación del evento con la política ser usada para correlacionar con nuevos eventos.

- **Correlación cruzada***. Indica si debería usarse para correlacionar con plugins de correlación cruzada “cross-plugin”, detectores de sistemas “ids-os” o con detectores de servicios “ids-service”.
- **Almacén de evento***. Indica si deberían de ser almacenados en la base de datos.
- **Calificar los eventos**. Indica si estos eventos afectan a la valoración de riesgos, si debe de modificar los niveles de C & A.
- **Reenviar alarmas**. Indica si las alarmas generadas deben de ser reenviadas al resto de servidores.
- **Reenviar eventos**. Indica si los eventos generados deben de ser reenviados al resto de servidores.

- **Descripción**. Es un simple campo de descripción usado para describir de forma breve la política.

7.7 Inventario OCS.

OCS es un sistema GNU *“Cliente/Servidor”* de inventario de Hardware y Software de los equipos, recolecta la información desde la red instalando un agente en cada una de las maquinas a realizar el inventario y la envía al servidor. Centralizando el inventario de todas las máquinas, permitiendo visualizar los resultados de los inventarios y recibir actualizaciones periódicas de cada maquina. La comunicación entre los agentes y el servidor de administración se realiza mediante el protocolo de transferencia de hipertexto http o https (modo seguro), y el formato de los datos está en XML comprimido en Zlib para reducir el tráfico de la red.

El servidor de administración utiliza Apache, MySQL y Perl. OCS es multiplataforma, gracias a su diseño simple y el uso de *“mod_perl”* del lado del servidor, el rendimiento es muy bueno para máquinas que no precisan de una gran potencia.

Los agentes se pueden instalar en multitud de sistemas operativos como son, Windows, Linux, Mac OS X, Solaris, IBM AIX.

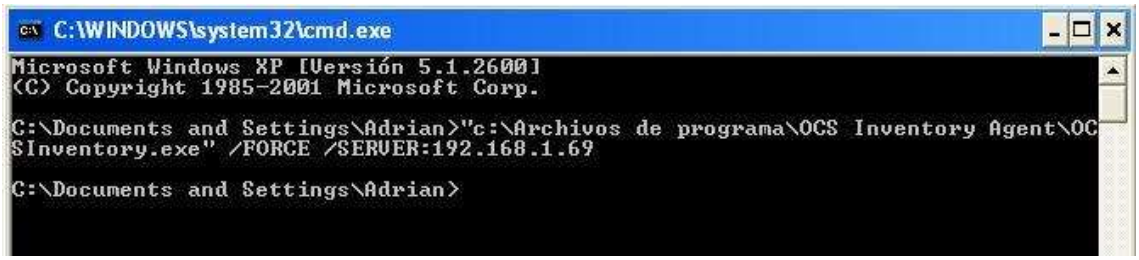
Ossim ha integrado el servidor de OCS para poder realizar un gran inventario de las máquinas contenidas en la red, esto demuestra la poderosa capacidad de inventario que la plataforma Ossim recoge gracias a la integración de OCS.

Instalación del agente OCS.

En cada una de las maquinas que se quiera realizar el inventario deberemos instalar el agente de OCS, para que recoja la información y se la envíe al servidor. La configuración de este agente es muy simple ya que solo bastará con especificar la dirección Ip del servidor para que pueda enviar correctamente la información.

* Estas acciones no se aplicaran sobre objetivos que no tengan una base de datos asociada.

Una vez instalado el agente, desde Ossim no veremos ningún inventario nuevo en la sección de *“Informes/OCS Inventory”*, ya que el agente enviará periódicamente la información recogida, para forzar un primer envío y así comprobar que los datos se reflejan correctamente en el servidor, deberemos lanzar la siguiente instrucción desde la consola de comandos de Windows.



16. Comando para forzar envío de inventario al servidor.

Desde Ossim deberemos ahora sí visualizar los datos del inventario recogidos desde cada agente. OCS tiene un gran potencial ya que recoge multitud de datos sobre el hardware y el software como:

- **Bios**. Número de serie del sistema, modelo, fabricante, versión, fecha.
- **Procesador**. Tipo, velocidad, número de procesadores.
- **Memoria física**. Capacidad en MB y velocidad.
- **Dispositivos de entrada**. Fabricante, tipo, interfaz usada.
- **Puertos de sistema**. Tipo, nombre, descripción.
- **Periféricos de almacenamiento**. Fabricante, modelo, tipo, capacidad.
- **Particiones lógicas**. Sistema de archivos, capacidad, espacio libre.
- **Adaptadores de red**. Nombre, modelo, descripción.
- **Sistema Operativo**. Nombre, versión, etc.
- **Software**. Extraído del registro muestra: nombre versión.
- **Valores del registro**. Se pueden realizar consultas sobre algún valor del registro.

Desde la pantalla principal de OCS visualizaremos a modo de resumen los últimos inventarios recibidos desde cada maquina.

Tag	Last inventory	Computer	User	Operating system	RAM(MB)	CPU(MHz)
0	12/02/2008 23:20:06	vmOssim	root	Linux	504	1729
NA	12/02/2008 23:09:05	SOBREMESA	Adrian	Microsoft Windows XP Professional	1024	2813
NA	12/02/2008 20:28:14	SONYMOBILE	Adrian	Microsoft Windows XP Professional	1024	1729
NA	11/30/2008 13:24:12	SOBREMESAADRIAN	Adrian	Microsoft Windows XP Professional	640	1693
NA	11/30/2008 13:08:14	SOBREMESAADRIAN	Adrian	Microsoft Windows XP Professional	640	1693

17. Inventario OCS integrado en Ossim.

7.8 Correlación.

Como ya mencionamos en el apartado de funcionalidad de Ossim la correlación es una de las funciones más importante del núcleo ya que reduce considerablemente los falsos positivos y aumenta la fiabilidad de los falsos negativos. El número de eventos se reduce a tan sólo algunas alarmas que son mucho más fiables y fáciles de analizar.

El motor de correlación de Ossim se alimenta de las directivas[5] creadas para generar las alarmas. En este punto vamos a explicar con un gran nivel de detalle como se implementa una directiva.

Directivas.

Ossim define las directivas usando el pseudo-lenguaje XML 1.0, a diferencia del resto de la gestión que se realiza vía desde la interfaz web, las directivas deben de crearse desde los archivos de configuración internos en la ruta “/etc/Ossim/Server”, para la nueva versión que están desarrollando prometen integrar la definición de las directivas en el interfaz web.

Las directivas son un tipo especial de “plugin”. Cuando una directiva genera una alarma lo que hace es crear un tipo especial de evento, que al igual que cualquier otro caso debe de ser generado por algún plugin.

Una gran corporación tras una exhaustiva configuración podría crear miles de directivas en su red, que sin una buena organización podría volverse en su contra y ser un monstruo incapaz de manejar. Para que esto no ocurra se ha definido una serie de categorías donde alojaremos las directivas usando el rango definido para cada categoría.

Categoría	Números
Ossim Genérico	1-2999
Correlación de ataques	3000-5999
Virus y Gusanos	6000-8999
Correlación de ataques Web	9000-11999
DoS	12000-14999
Portscan/scan	15000-17999
Comportamiento anómalo	18000-20999
Abuso y error de red	21000-23999
Troyanos	24000-26999
Varios	27000-34999
Contribución del usuario	500000+

18. Categorías.

[5] Una directiva es un conjunto de reglas que se deben de cumplir para que Ossim genere una alarma con un determinado nivel de riesgo.

Elementos de una directiva.

A continuación vamos a describir cada una de las etiquetas que se utilizan en el pseudo-lenguaje XML para definir las directivas.

Una directiva se inicia con tres etiquetas:

- **Id**. Identificador único de la directiva, comprendido dentro del rango de su categoría.
- **Name**. Nombre descriptivo de la directiva.
- **Priority**. Nivel de prioridad que es global a al directiva.

Cada directiva empieza con una regla que debe de coincidir con un evento y aquí empieza la correlación. Las reglas pueden contener las siguientes etiquetas:

- **Type**. Indica el tipo de regla, que puede ser:
  - **Detector**. La regla define eventos que son producidos por los agentes detectores como (snort, spade, apache, etc.).
  - **Monitor**. La regla define eventos que son producidos por el servidor de datos ntop y sesiones ntop.
- **Name**. Nombre descriptivo de la regla.
- **Priority**. Es el nivel de importancia que tiene el evento. No tiene relación con un equipo en concreto ni con el entorno, solo mide la importancia relativa del evento. Por ejemplo, si un servidor Unix con Samba es atacado por un gusano Sasser, el ataque de por si es peligroso, pero ¿le afecta a dicha máquina? Seguramente no, pero la regla de por si no realiza esta distinción por lo que como el ataque es un importante agujero de seguridad tendrá una alta prioridad.
- **Reliability**. Cuando hablamos del riesgo de un ataque se podría definir como la “probabilidad” de que esto ocurra. Volviendo al ejemplo anterior, si una máquina se conecta a 5 máquinas distintas de su propia subred al puerto 445, puede presentar un comportamiento normal, pero si este número va aumentando y ya son 15 las máquinas, empieza a ser un tanto sospechoso, y si se comunica con más de 30 máquinas en menos de una hora, se puede decir que el comportamiento extraño y el ataque comienza a ser evidente. Cada regla tiene su propia fiabilidad, determinando la fiabilidad de esa regla en particular dentro de la cadena de reglas completas de la directiva. El valor de la fiabilidad puede ser entre 0 y 10, se puede especificar un valor absoluto (ej. 6) o relativo (ej. +3 significa tres más que en nivel anterior).
- **Ocurrence**. Indica cuántas veces debemos encontrar una coincidencia para avanzar un nivel de correlación.
- **Time_out**. Indica el tiempo de vida de una regla, esperando a que el evento ocurra, hasta que la regla expira.

- **From**. Ip origen. Hay varios valores posibles para este campo:

  - ANY, como la palabra índica cualquier dirección Ip servirá.
  - Una dirección IPv4 numérica separada con puntos (x.x.x.x). Se puede utilizar más de una dirección Ip separadas por comas
  - Utilizando un nombre de equipo definido.
  - Relativo, se utiliza para referenciar direcciones Ip de niveles anteriores, (ej. “1:SRC_IP” significa que utilizará la dirección Ip origen referenciada dentro de la regla del nivel 1).
  - Negado. También se pueden usar elementos negados, (ej. “!192.168.1.11, HomeNetwork” esto indica que si la red HomeNetwork = 192.168.1.0/24, utilizará todas las Ip contenidas en la red experto la 192.168.1.11).
  - Podemos usar nombre de redes definidos en la base de datos.

- **To**. Ip destino, es el campo usado al referirse a los datos del monitor que no tienen ningún origen. Hay varios valores posibles para este campo:

  - ANY, como la palabra índica cualquier dirección Ip servirá.
  - Una dirección IPv4 numérica separada con puntos (x.x.x.x). Se puede utilizar más de una dirección Ip separadas por comas
  - Utilizando un nombre de equipo definido.
  - Relativo, se utiliza para referenciar direcciones Ip de niveles anteriores, (ej. “2:DST_IP” significa que utilizará la dirección Ip destino referenciada dentro de la regla del nivel 2).
  - Negado. También se pueden usar elementos negados, (ej. “!192.168.1.11, HomeNetwork” esto indica que si la red HomeNetwork = 192.168.1.0/24, utilizará todas las Ip contenidas en la red experto la 192.168.1.11).
  - Podemos usar nombre de redes definidos en la base de datos.

- **Sensor**. Indica el sensor que debe obtener el evento. Ha varios valores posibles para este campo:

  - ANY, como la palabra índica cualquier dirección Ip servirá.
  - Una dirección IPv4 numérica separada con puntos (x.x.x.x). Se puede utilizar más de una dirección Ip separadas por comas
  - Usando un nombre de sensor definido en la interfaz web.
  - Relativo, se utiliza para referenciar direcciones Ip de un sensor en los niveles anteriores, (ej. “1:SENSOR” significa que utilizará la dirección Ip del sensor referenciado dentro de la regla del nivel 1).
  - Negado. También se pueden usar elementos negados, (ej. “!192.168.1.11” utilizará todos los sensores de la red experto el 192.168.1.11).

- **Port_from / Port_to**. Indica los posibles puertos origen y puertos destino del evento. Hay varios valores posibles para este campo:

  - ANY, como la palabra índica cualquier puerto es valido.

- Un número de puerto. Se puede utilizar más de un puerto separados por comas (ej. “24,1024, 445”)
- Usando un rango de puertos (ej.”1000-2000”).
- Relativo, se utiliza para referenciar un puerto en los niveles anteriores, (ej. “1:SRC_PORT” o “2:DST_PORT” significa que utilizará el puerto origen referenciado dentro de la regla del nivel 1).
- Negado. También se pueden usar elementos negados, (ej. “!11,25,!34,!21” o negar un rango “1-1000,!1-100”).

- **Protocolo**. Especifica el protocolo utilizado por el evento, puede ser uno de los siguientes textos:

  - TCP, UDP, ICMP, Host_ARP_Event, Host_OS_Event, Host_Service_Event, Host_IDS_Event, Information_Event. Aunque Host_ARP, Host_OS, etc, no son realmente un protocolo, se pueden usar si queremos definir una directiva con ARP, OS, IDS, o eventos de servicio.

  - Además, también se puede insertar el número del protocolo.

  - Se pueden usar referencias relativas como en los casos anteriores (ej. “1:TCP” o “2:UDP”).

  - Por último, también es posible realizar negaciones (Ej., “!ICMP”).

- **Plugin_id**. Identificador numérico del plugin_id referenciado.

- **Plugin_sid**. Identificador numérico del plugin-sid referenciado, se pueden usar los siguientes valores:

  - ANY, identifica cualquier plugin_sid del padre.

  - Cualquier número de plugin_sid o usarlo como negación (ej. “1,!2,3,!4”).

- **Condition**. Este parámetro y los tres siguientes son solo validos para los tipos de regla “monitor” y algunos tipos de reglas “detector”. Indica la condición lógica que debe coincidir con la regla. Los valores que acepta son:

  - Eq – Igual.
  - Ne – No igual.
  - Lt – Menor que.
  - Gt – Mayor que.
  - Le – Menor o igual que.
  - Ge – Mayor o igual que.

- **Value**. El valor que debe de cumplir el evento usando el comparador de la etiqueta anterior.

- **Interval**. Intervalo de tiempo en el que debe de producirse el evento.

- **Absolute**. Determina si el valor proporcionado es absoluto o relativo. Por ejemplo, si tenemos un valor de “100”, una condición “gt” y un intervalo de 60 (segundos), para un evento de stop “HttpSentBytes” significa:

  - Si el valor es **true**, la condición se cumplirá si el equipo envía más de 100 bytes en 60 segundos por el protocolo http.
  - Si el valor es **false**, la condición se cumplirá si el equipo realiza un incremento en 100 bytes durante el periodo de 60 segundos por el protocolo http.

- **Sticky.** Si traducimos la palabra al castellano como “adherir” seria una buena descripción de la funcionalidad que quiere dar esta etiqueta. Lo que se pretende es abordar un grupo de eventos con un mismo patrón desde una única directiva y no genera múltiples directivas. Fijando este valor a true/false se comportará de la manera deseada.

- **Sticky_different**. Solo valido para reglas con más de una salida. Lo que se pretende es asegurarse de que especificando el parámetro X como la posibilidad de tener varios valores a la vez, se adhieran todas las ocurrencias diferentes para dicho valor. Tomemos como ejemplo un escaneo de puertos, si fijamos la etiqueta sticky_different=”1:DST_PORT”, esto indica que todas las ocurrencias para el mismo equipo con distinto puerto serán adheridas en la misma regla.

- **Groups**. Como la etiqueta “Sticky” pero que afecta a más de una directiva. Si un evento coincide con una directiva definida con un grupo que esta localizado como “sticky”, este no podrá coincidir con otra directiva.

Ejemplo de una directiva que detecta un posible gusano.

En primer lugar, tenemos la etiqueta inicial, donde definiremos el identificador, el nombre y la prioridad.

```
<directive id=”4” name=”Posible gusano” priority=”4”>
```

La primera regla busca alguna conexión anómala con el plugin spade contra los puertos indicados, puertos típicos de un gusano.

```
<rule type=”detector” name=”Extraña conexión en el destino” reliability=”1” ocurrente=”1” from=”ANY” to=”ANY” port_from=”ANY” port_to=”25,80,135,137,139,445,1433,1434” plugin_id=”1104” plugin_sid=”ANY”>
```

La siguiente es una etiqueta de apertura, que indica que si la anterior norma ha ocurrido pase al siguiente nivel de normas.

```
<rules>
```

En el segundo nivel la primera regla indica que la dirección origen ha realizado más de 15 conexiones extrañas sobre el mismo puerto destino para diferentes equipos destino en 3 minutos, como hemos fijado la etiqueta sticky_different= “DST_IP” nos aseguramos que las 15 ocurrencias han sido para diferentes destinos.

```
<rule type = "detector" name = "Demasiadas conexiones raras (15) contra el mismo puerto" reliability="3" time_out="180" occurrence="15" from="1:SRC_IP" to="ANY" port_from="ANY" port_to="1:DST_PORT" plugin_id="1104" plugin_sid="ANY" sticky="true" sticky_different="DST_IP">
```

En los siguientes niveles se va ha seguir la misma lógica pero aumentando el número de ocurrencias y el intervalo de tiempo hasta las 20000 ocurrencias en 12 horas, (43200 segundos) que es la regla definida en el último nivel, (nivel 5) este será el único nivel donde no usamos la etiqueta sticky_different porque en el nivel anterior ya tenemos 2000 diferentes equipos atacados y el servidor no quiere perder de vista ninguno de los equipos.

Conforme va bajando la correlación por los distintos niveles tenemos la confianza de que el ataque es cierto y la fiabilidad va aumentando.

A continuación mostramos el ejemplo de la directiva completo:

```
<directive id=”4” name=”Posible gusano” priority=”4”>
<rule type=”detector” name=”Extraña conexión en el destino” reliability=”1” ocurrente=”1” from=”ANY” to=”ANY” port_from=”ANY” port_to=”25,80,135,137,139,445,1433,1434” plugin_id=”1104” plugin_sid=”ANY”>
    <rules>
    <rule type = "detector" name = "Demasiadas conexiones raras (15) contra el mismo puerto" reliability="3" time_out="180" occurrence="15" from="1:SRC_IP" to="ANY" port_from="ANY" port_to="1:DST_PORT" plugin_id="1104" plugin_sid="ANY" sticky="true" sticky_different="DST_IP">
        <rules>
        <rule type="detector" name=" Demasiadas conexiones raras (300) contra el mismo puerto " reliability="5" time_out="1200" occurrence="300" from="1:SRC_IP" to="ANY" port_from="ANY" port_to="1:DST_PORT" plugin_id="1104" plugin_sid="ANY" sticky="true" sticky_different="DST_IP">
            <rules>
            <rule type="detector" name=" Demasiadas conexiones raras (2000) contra el mismo puerto " reliability="10" time_out="1800" occurrence="2000" from="1:SRC_IP" to="ANY" port_from="ANY" port_to="1:DST_PORT" plugin_id="1104" plugin_sid="ANY" sticky="true" sticky_different="DST_IP">
                <rules>
```

```
<rule type="detector" name=" Demasiadas conexiones raras (20000) contra el mismo puerto" reliability="10" time_out="43200" occurrence="20000" from="1:SRC_IP" to="ANY" port_from="ANY" port_to="1:DST_PORT" plugin_id="1104" plugin_sid="ANY" sticky="true"/>
</rules>
</rule>
</rules>
</rule>
</rules>
</rule>
</rules>
</rule>
</directive>
```

7.9 Evaluación del Riesgo.

El riesgo es una interesante manera de normalizar y determinar un valor que puede ser utilizado para tomar decisiones sobre una serie de ataques realizados en un plazo específico de tiempo.

Como ya hemos visto en el apartado de valoración de riesgo, Ossim utiliza dos valores de riesgo diferentes, uno para el origen y otro para el destino de un evento. Por lo tanto cada caso tiene dos riesgos. Esto significa que como tenemos dos máquinas diferentes para calcular el riesgo, cada una de ellas será evaluada con su propio valor del activo. Estas son las dos reglas que vamos a seguir:

1. Si estamos calculando el riesgo de C, utilizaremos el valor del activo de la dirección origen.
2. Si estamos calculando el riesgo de A, utilizaremos el valor del activo de la máquina destino.

Así que, gracias a esto, el valor de riesgo de cada directiva se utiliza para modificar el valor de C & A de cada máquina. Para normalizar el cálculo del valor de riesgo se utiliza la siguiente formula:

Riesgo = (Activo * Prioridad * Fiabilidad) / 25

Esta es la manera de normalizar el valor del riesgo para todos los casos posibles, siendo el riesgo mínimo un 0 y el máximo riesgo el valor de 10. Las variables internas de la formula pueden tomar los siguientes rangos:

- Prioridad entre 0 y 5.
- Fiabilidad entre 0 y 10.
- Activo entre 0 y 5.

De esta forma justificamos el valor máximo y mínimo que puede alcanzar el nivel de riesgo de una directiva.

8. Gestión de Ossim.

Una vez Ossim ha sido configurado, desde la misma interfaz web tenemos multitud de secciones desde donde podremos visualizar y gestionar el estado de la red. En este capitulo vamos a detallar cada una de las secciones que nos permitirán realizar una gestión diaria de nuestra red.

8.1 Panel de Control.

El panel de control es el punto de partida de la aplicación Ossim. Una vez se ha configurado el servidor, desde aquí podremos controlar el estado de la red de una forma gráfica con una rápida visibilidad.

El propósito de este panel no es más que el de ver con una simple mirada, como esta la organización sin entrar en ningún nivel de detalle. Esto es muy importante ya que nos ahorrará tiempo siempre que la red este en perfecto funcionamiento, y si en cualquier momento queremos profundizar en más detalle, consultaremos el resto de secciones descritas más adelante.

19. Panel de Control.

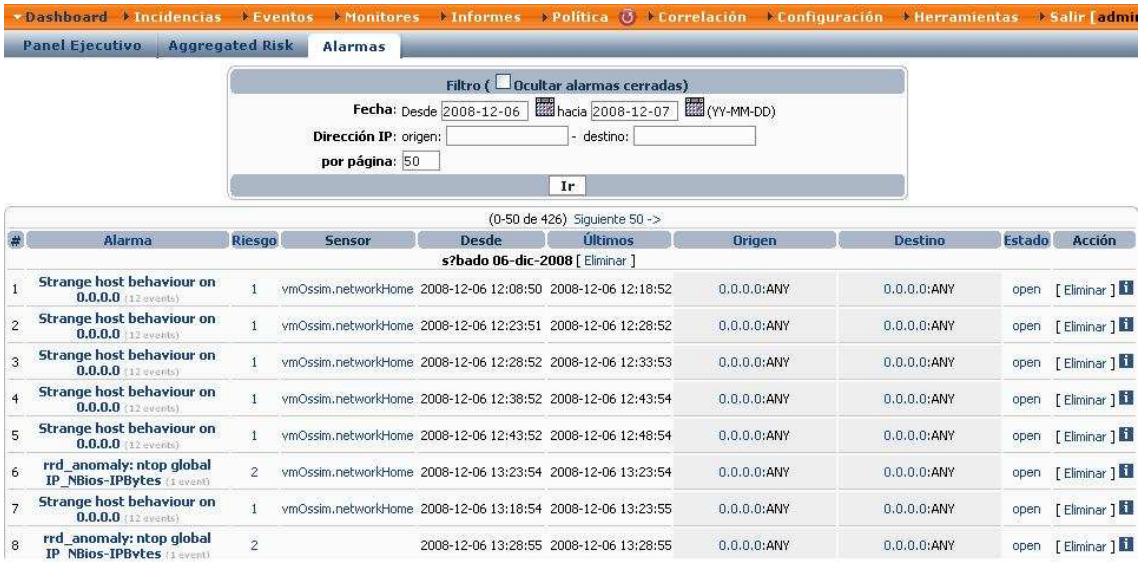
En la imagen anterior mostramos una pequeña parte del panel de control, ya que desde aquí podemos visualizar muchos más gráficos. Desde el botón editar podremos configurar esta pantalla como queramos, añadiendo o quitando gráficos según nos convenga.

La lista de gráficos que se podrá visualizar es:

- **Eventos por Sensor y plugins:** Muestra la cantidad de eventos agrupados por plugins de un determinado sensor.
- **Alarmas y Eventos:** Muestra la cantidad de alarmas y eventos que se han producido por día y agrupados por tipos.
- **Nivel de servicio:** Muestra la disponibilidad del servicio ofrecido.
- **Throughput:** Muestra el nivel de congestión de la red actual contrastado con la predicción que el genera automáticamente basándose en la información recopilada.
- **Top 10 equipos en riesgo:** Muestra los 10 equipos con mayor riesgo.
- **Estado de las incidencias:** Muestra el porcentaje de incidencias cerradas y abiertas.
- **Incidencias resueltas en el tiempo:** Muestra la cantidad de incidencias cerradas por día (en los últimos 6 días).
- **Tipo incidencias:** Muestra el porcentaje de incidencias agrupadas por tipo.
- **Incidencias cerradas por mes:** Muestra la cantidad de incidencias cerradas por mes.
- **Trafico por protocolo:** Muestra el trafico (bytes/sec) de los últimos 10 minutos por protocolo.
- **Inventario de software:** Muestra el porcentaje de tipos de sistemas operativos y software instalado en las máquinas.

8.2 Alarmas.

Desde esta sección podemos ver todas las alarmas generadas por Ossim, con un nivel de detalle más ampliado. Las alarmas son generadas por eventos correlacionados o no, que exceden de un cierto riesgo, por defecto 1. Muestra información acerca de cualquier intrusión o intento de intrusión en la red.



20. Gestión Alarmas.

Como se muestra en la parte superior de la imagen podremos filtrar las alarmas por fecha, dirección ip y estado. Desde la tabla de alarmas podemos visualizar con detalle que es lo que ha ocurrido y gestionarlas.

A continuación explicamos cada uno de los campos:

- **Alarma:** Es el nombre que se le asigno al plugin o a la directiva que ha generado esta alarma, muestra también el número de eventos que han ocurrido sobre la misma directiva.
- **Riesgo:** Es el nivel de riesgo que ha causado esta alarma.
- **Sensor:** Es el nombre del sensor que ha obtenido dicha alarma.
- **Desde-Hasta:** Intervalo de tiempo en el que han transcurrido los eventos.
- **Origen:** Dirección Ip y puerto del equipo origen.
- **Destino:** Dirección Ip y puerto del equipo destino.
- **Estado:** Estado abierto/cerrado.
- **Acción:** Desde esta columna podremos gestionar las alarmas, eliminándolas si no nos interesa tener registro de ella, o creando una incidencia, (*botón cuadrado con una” i” dentro*) desde este botón iremos a la sección de incidencias que será explicada más adelante, donde podremos crear nuevas incidencias y guardar el procedimiento seguido para cerrar las alarmas.

8.3 Incidencias.

La pantalla incidencias muestra el estado de las incidencias creadas automáticamente por Ossim o las incidencias creadas manualmente a partir de una alarma. Desde aquí podremos gestionar las incidencias cerrándolas, enviar un correo o redactar una descripción del procedimiento seguido para cerrar la incidencia.

21. Gestión Incidencias.

Como en la pantalla de alarmas podremos realizar un filtro para poder buscar las incidencias deseadas. Desde esta pantalla podremos gestionar las incidencias o crear nuevas incidencias.

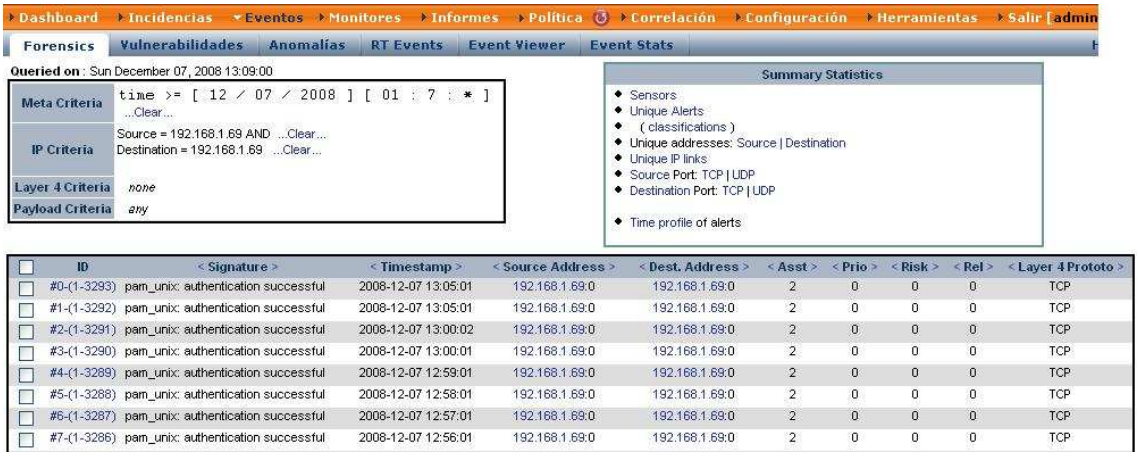
Desde la pestaña “Tipos”, podremos crear nuevos tipos de incidencias para luego clasificar las incidencias, por defecto Ossim ya viene configurado con varios tipos de incidencias.

Desde la pestaña “Etiquetas”, podremos crear nuevas etiquetas, para etiquetar las incidencias.

Desde la última pestaña “Informes”, se visualizan una serie de gráficos estadísticos, sobre las incidencias (Porcentaje de Incidencias por tipo, Porcentaje de incidencias por estado, Porcentaje de Incidencias por usuario y su estado, Tiempo de resolución de incidencias, etc.).

8.4 Eventos.

Desde aquí podemos ver todos los eventos generados por los agentes y que Ossim ha recopilado, desde esta sección se realizará el análisis forense, ya que muestra todas las anomalías y eventos recolectados. Podremos realizar un filtro para solo obtener los eventos relacionados con una alarma, maquina, tipo, etc.

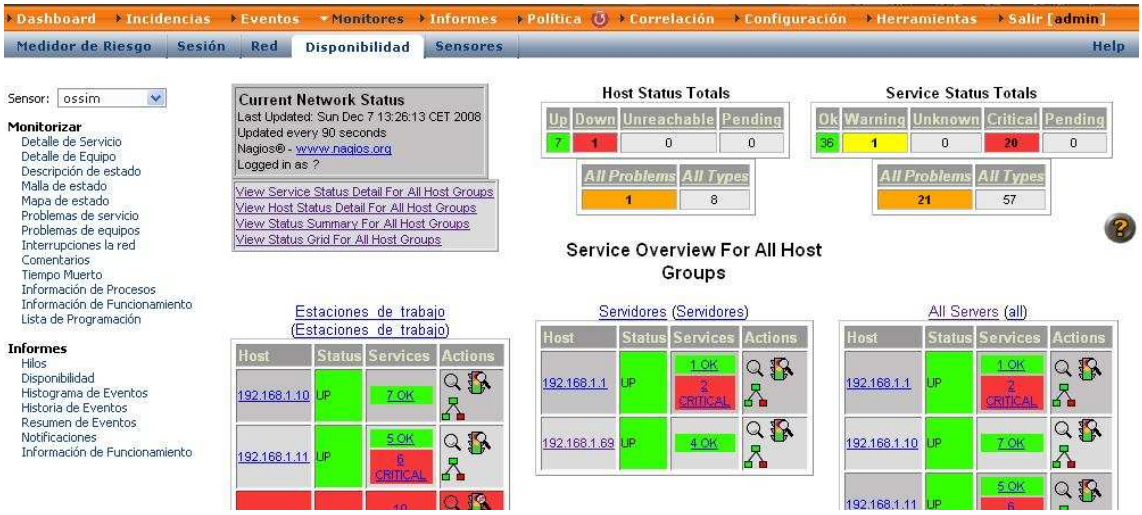


22. Análisis Forense.

Desde esta sección también se podrán ver los eventos en tiempo real desde la pestaña de “RT Eventos” o desde la pestaña de “Event Viewer” los eventos clasificados por tipo y agrupados por día.

8.5 Monitores.

Desde esta sección podremos visualizar con mayor detalle el estado actual de la red. Ossim ha integrado en esta sección las aplicaciones Ntop y Nagios para ver las diferentes posibilidades que nos da cada monitor sobre el estado de la red.



23. Disponibilidad de Servicios.

Desde la sección de monitores podremos ver:

- **Medidor de riesgo:** El riesgo acumulado para las redes y equipos.
- **Sesión:** las sesiones activas por equipo y el tráfico generado por estos.
- **Red:** Estadísticas del tráfico global a la red.
- **Disponibilidad.** La disponibilidad de los servicios ofrecidos por cada máquina, con posibilidad de agrupar por (puerto, equipo, estado, etc.).
- **Sensores.** Estado de los agentes ofrecidos por cada sensor (ntop, snare, nagios, nessus, etc.).

8.6 Informes.

Desde la sección de informes podemos visualizar y generas reportes de tipo “pdf” con las graficas que representan el estado de seguridad y alarmas detectadas por Ossim. Desde aquí también podremos visualizar el inventario realizado por el agente “OCS Inventory”.

Cada una de las pestañas dentro de la sección Informes, muestran datos diferentes que se podrán reportar a pdf desde la pestaña “reporte PDF”.

24. Informes.

Desde las distintas pestañas visualizadas dentro de la sección informes podremos ver:

- **Informe de Equipos**: Resumen de las máquinas inventariadas.
- **Reporte de Alarmas**: Varias representaciones graficas de las alarmas generadas como (los 10 equipos más atacados, los 10 equipos mas atacantes, los 10 puertos más usados, las 10 alarmas más generadas y las 10 alarmas con mayor riesgo).
- **Informe de seguridad**: Se representan las mismas graficas que en apartado anterior pero para los eventos generados.
- **OCS Inventory**: Como ya hemos explicado con más detalle en este documento, aquí se visualiza un exhaustivo inventario de las máquinas.
- **Reporte PDF**: Desde esta pestaña podremos realizar reportes a pdf, de los informes generados en las pestañas anteriores.

8.7 Configuración.

La sección de configuración, nos permite configurar diversas opciones del framework de Ossim como la apariencia, usuarios, plugins, plantilla correo, actualizaciones, etc.

8.8 Herramientas.

La sección de herramientas incorpora utilidades como el análisis de red que nmap proporciona, visor de las reglas creadas, realización de las copias de seguridad, Log de usuario, y una pestaña con enlaces web para descarga de agentes.

9. Conclusión.

Las necesidades que los diseñadores de Ossim han querido suplir desarrollando esta plataforma, son más que acertadas. Ya que hasta ahora gozábamos de aplicaciones teóricamente muy buenas, pero que en la practica eran muy difícil de manejar por la cantidad de eventos recibidos y la complejidad de ellas.

No obstante, Ossim para nada es un sistema fácil de configurar, ya que su nivel de complejidad a la hora de ser configurado es prácticamente infinito, dependiendo del nivel de detalle que se quiera controlar. Pero es que por si mismo, la seguridad de una red es compleja y es un punto del que siempre vamos a partir.

Ossim nos ofrece la posibilidad de realizar desde una configuración de seguridad mínima con un nivel de complejidad mínimo, hasta un nivel de detalle prácticamente infinito que siempre podremos estar mejorando. Amoldándonos a las necesidades de cada empresa.

Además, el principal objetivo por el que Ossim ha sido creado, para facilitar la gestión diaria y la visión del estado de la red en una simple mirada, es donde Ossim esta un nivel por encima de las demás aplicaciones, proporcionando una visión simple y rápida, y una gestión con un volumen de alarmas menor y mucho más fiables.

Por otra parte, las empresas buscan cada vez más el ahorro de costes, recortando todo tipo de gastos que no sea imprescindibles. El uso del software libre cada día es más aceptado por las empresas ya que reproduce un ahorro considerable.

Por eso, que las empresas de sistemas informáticos buscan cada vez más el uso de software libre, ya que con el mismo margen de beneficio, consiguen abaratar costes y poder tener un precio competitivo en el mercado.

Ossim cumple con estas necesidades que hoy en día son el principal objetivo de toda empresa. Si ha este reducción económica le añadimos todas las ventajas presentadas en este documento, se puede decir que Ossim en la actualidad es un de los sistemas mejor posicionado en el mercado para la implantación en las empresas como sistema de gestión de la seguridad.

Como se ha comprobado en esta investigación, Ossim se aprovecha de las mejores aplicaciones de código libre, para desarrollar una plataforma que mejore la gestión de la información de seguridad de las empresas haciendo la vida más fácil a los administradores de red.

No obstante y para concluir, Ossim es un proyecto que sigue en continuo desarrollo y tanto sus creadores como varios colaboradores, siguen actualizando y mejorando para el beneficio de todos.

10. Bibliografía.

http://www.ossim.net/

http://www.ossim.net/docs.php

http://www.alienvault.com/home.php?id=download

http://www.ossim.net/dokuwiki/doku.php?id=user_manual:introduction

http://www.ossim.net/dokuwiki/doku.php?id=faq

http://www.belt.es/noticias/2005/abril/05/osimm.htm

http://www.debian.org/international/spanish/

http://www.seguridaddigital.info/index.php?option=content&task=view&id=108

http://sourceforge.net/projects/os-sim/

http://es.wikipedia.org/wiki/Ossim

Apéndice A – Estructura interna archivos.

Archivos del servidor

- **Libos-sim.h**. Lista de archivos incluidos.
- **Main.c**. Archivo principal de configuración, que inicia el servidor Ossim y arranca los múltiples hilos con sus tareas.
- **Os-sim.h**. Principal estructura donde todos los datos serán almacenados.
- **Sim-action [CH]**. Obsoleto, nunca se utilizo. Sigue en el código fuente por razones históricas.
- **Sim-category [CH]**. Obsoleto, categorías relacionadas con snort. No está en uso en este momento.
- **Sim-classification [CH]**. Obsoleto, clasificación relacionada con snort. No está en uso en este momento.
- **Sim-command [CH]**. Todos los datos que entran en el servidor se analizan aquí y se almacenan en una estructura especial. Estos datos pueden provenir de otro servidor, sensor, o el mismo framework.
- **Sim-config [CH]**. Contiene la configuración del servidor Ossim, la base de datos a la que debe conectarse, puerto servidor.
- **Sim-connect [CH]**. Envía un evento específico al framework. Necesario para hacer coincidir el evento con la acción / respuesta que tratará de hacer.
- **Sim-container [CH]**. Contiene toda la estructura de datos que debe ser almacenada en la memoria, como los equipos, la lista de directivas. Proporciona funciones de acceso a las estructuras de datos, para extraer información especifica de ella.
- **Sim-database [CH]**. Contiene funciones para acceder a las bases de datos.
- **Sim-directive [CH]**. Se utiliza para almacenar información de una directiva.
- **Sim-directive-groups [CH]**. Contiene la asociación de una directiva a un grupo.
- **Sim-enums.h**. Contiene las variables utilizadas en las directivas xml, así como otro tipos de enumerados de ossim, tipo de protocolo, comandos.
- **Sim-event [CH]**. Funciones para extrae o insertar la información de cada evento
- **Sim-host [CH]**. Cada equipo que se define en la sección de política se puede extraer de aquí.
- **Sim-host-level [CH]**. Cada equipo tiene un nivel de C y A asociados que se pueden extraer de aquí.
- **Sim-inet [CH]**. Funciones para transformar y comprobar las relaciones ente los objetos SimInet.
- **Sim-log [CH]**. Registro de manejadores y registro de manipulación.
- **Sim-net [CH]**. Cada red definida en el framework es almacenada aquí.
- **Sim-net-nivel [CH]**. Cada red tiene un C y A asociado que se puede extraer de aquí.
- **Sim-organizer [CH]**. Este es el circuito donde todos los cálculos con respecto a un acontecimiento se realizan.
- **Sim-plugin [CH]**. Almacena en memoria cada uno de los plugins de la base de datos.
- **Sim-plugin-sid [CH]**. Almacena en memoria cada uno de los sub-plugins de la base de datos.

- **Sim-plugin-state [CH]**. Contiene el estado de cada plugin. Si esta activo o no.
- **Sim-policy [CH]**. Contiene todas las políticas creadas.
- **Sim-rule [CH]**. Contiene cada uno de los nodos de las directivas.
- **Sim-scheluder [CH]**. Ejecuta algunas funciones con regularidad.
- **Sim-sensor [CH]**. Contiene información acerca de un sensor. También contiene la cantidad de eventos que han llegado al sensor en los últimos minutos.
- **Sim-server [CH]**. Contiene datos del servidor.
- **Sim-session [CH]**. Contiene información de cada sesión generada por sim-server.
- **Sim-smtp [CH]**. Obsoleto no esta en uso.
- **Sim-util [CH]**. Asistente de funciones.
- **Sim-xml-config [CH]**. Carga la configuración del archivo xml del servidor donde indica la estructura del fichero SimConfig.
- **Sim-xml-directiva [CH]**. Carga el archivo de directivas xml y almacena todos los datos en la memoria, para ser accedida más tarde por e proceso de correlación.

Archivos de los agentes.

- **Doc/ossim-agent.8.gz**
- **Doc/ossim-agent.xml**
- **Etc/aliases.cfg**
- **Etc/config.cfg**
- **Etc/plugins/apache.cfg**
- **Etc/plugins/arpwatch.cfg**
- **Etc/plugins/mwcollect.cfg**
- **Etc/plugins/ntsyslog.cfg**
- **Etc/plugins/pOf.cfg**
- **Etc/plugins/pads.cfg**
- **Etc/plugins/snort.cfg**
- **Etc/plugins/syslog.cfg**
- **Ossim-agent**
- **Ossim_agent/agent.py**
- **Ossim_agent/onfig.py**
- **Ossim_agent/.conn.py**
- **Ossim_agent/.detector.py**
- **Ossim_agent/event.py**
- **Ossim_agent/exceptions.py**
- **Ossim_agent/logger.py**
- **Ossim_agent/parserLog.py**
- **Ossim_agent/parserUtil.py**
- **Ossim_agent/watchdog.py**
- **Ossim_agent/.init.py**

Archivos del FrameworkD

Os-sim/frameworkd:

- **Ossim-framework**
- **Ossimframework/acidCache.py**
- **Ossimframework/action.py**
- **Ossimframework/actionExec.py**
- **Ossimframework/actionMail.py**
- **Ossimframework/actionsuslog.py**
- **Ossimframework/const.py**
- **Ossimframework/controlPanel.py**
- **Ossimframework/doNessus.py**
- **Ossimframework/framework.py**
- **Ossimframework/listener.py**
- **Ossimframework/ossimConf.py**
- **Ossimframework/ossimDB.py**
- **Ossimframework/rddUpdate.py**
- **Ossimframework/soc.py**
- **Ossimframework/util.py**
- **Ossimframework/init.py**