OSSEC

Open Source HIDS SECurity

Francisco José Fernández Muelas

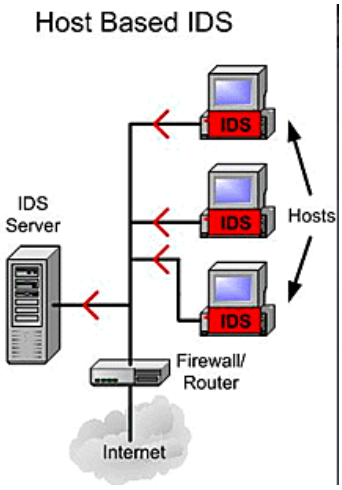
Ricardo Minelli

Servidores Web de Altas Prestaciones

2016

Introducción

OSSEC es un Host-based intrusion detection system (Sistema de detección de intrusos en un Host). Esto básicamente viene a ser un despliegue de IDS's basado en la arquitectura host-server. En esta arquitectura un manager (el server) recopila la información que producen los agents (hosts) y procesa esa información para generar alertas que informen al administrador del estado del sistema.



OSSEC es open source, con una licencia "GNU Public License (version 2)". Fue desarrollado por Daniel Cid, que lo hizo público en 2004. En 2009 la empresa Trend Micro adquirió los derechos para seguir manteniéndolo en el formato de código libre. Actualmente, su código se encuentra disponible en GitHub y cualquiera puede revisarlo o proponer mejoras.

Es un framework ampliamente usado en entornos profesionales. A pesar de llevar tantos años desarrollado, se producen de media unas 5000 descargas al mes. Aunque su instalación y configuración no es realmente muy compleja, al ser una herramienta tan grande, con tantas posibilidades de configuración y tantas características, varias empresas ofrecen sus servicios de soporte profesional para OSSEC, como Alient Vault y OSSEC.

Características de OSSEC

La principal función de OSSEC es monitorizar sistemas para detectar y evitar ataques malintencionados. Para ello, monitoriza cada uno de los logs que se producen en el sistema y

los analiza para entender que está pasando en el sistema. No solo monitoriza logs, si no que también escaneo puertos, procesos e interfaces en busca de anomalías.
Cuando una anomalía es detectada, OSSEC se encarga de alertar al administrador. Para ello, puede almacenar las alertas en archivos o bases de datos para su posterior consultar enviarlas via email mediante un servidor de correo.

Por otro lado, OSSEC es multiplataforma. Mientras que el "manager" tiene que estar instalado en un entorno Linux, los "agents" que monitoriza pueden ser de prácticamente cualquier sistema operativo: Linux, Solaris, AIX, HP-UX, BSD, Windows, Mac y VMware ESX.

Todo esto conlleva a que OSSEC además ayude a cumplir el estándar "Payment Card Industry Data Security Standard". Un sistema con OSSEC debidamente instalado y configurado cumple varias restricciones necesarias para poder trabajar con tarjetas de crédito y sistemas de pago bajo este estándar.

Rootcheck, procesos e integridad

OSSEC comprueba cada archivo en el sistema con una base de datos de malware, de manera similar a como lo haría un antivirus convencional. Además, cuando se instala (en un entorno que se entiende como no comprometido) hace un hash de cada uno de los archivos importantes del sistema y lo almacena. Periódicamente comprueba que el hash de los archivos coincide con el hash que se hizo al principio, y en caso de que no coincida, se entenderá que ha sido modificado de forma malintencionada (puesto que esos archivos sensibles no debería ser modificados) y se generará una alarma.

Además de esto, hace búsquedas para encontrar procesos ocultos. Esto lo realiza enviando señales (por ejemplo, con la orden "kill") a todos los posibles PIDs de un sistema. En caso de que algún de estos PIDs responda, y además no aparezca en un listado normal de procesos activos (como la orden "ps aux"), se entenderá que es un proceso que se ha ocultado intencionadamente y puede ser malicioso. De la misma manera, se hace con puertos e interfaces.

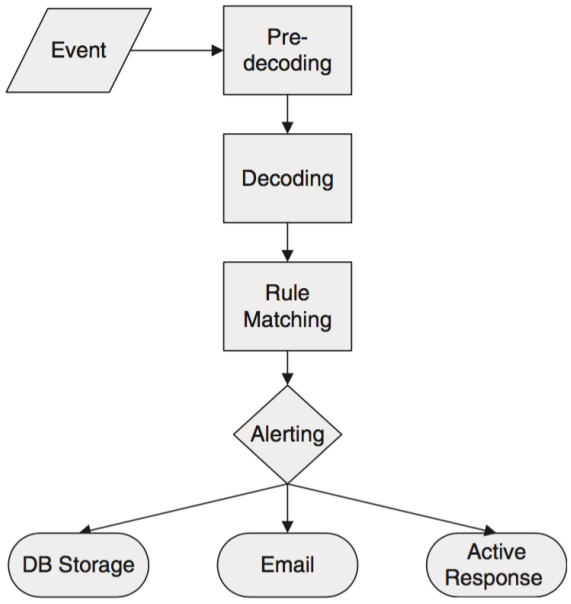
Otra de sus funciones es escanear el sistema en busca de archivos con propiedades sospechosas. Un ejemplo de esto podría ser un archivo de root que tenga habilitados los permisos de escritura para "others". OSSEC detectaría este archivo como potencialmente peligroso y generaría una alerta.

Análisis de logs

Realmente, OSSEC está diseñado como un LIDS: "log-based intrusion detection", es decir, un sistema de detección de intrusos basado en logs. El objetivo de un sistema de tales características es detectar ataques, uso indebido de recursos o errores en el sistema a través de información obtenida mediante el análisis de los logs que producen las aplicaciones, servicios o el propio sistema.

Los logs de cada uno de los "agents" son enviados y analizados en tiempo real por el "manager". El funcionamiento es a base de una serie de "decoders" y "rules" que descodifican y analizan los logs (utilizando expresiones regulares y un sistema de jerarquías) y generan una respuesta.

Cuando una entrada es añadida a un fichero de log se genera un evento que pasa por las siguientes fases:




```
Apr 14 17:32:06 linux_server sshd[1025]: Accepted password for dcid from
192.168.2.180 port 1618 ssh2
```

Dado un log en formato estándar "syslog" como el anterior, se produciría un evento que analizaría el "manager".

En primer lugar, el evento pasaría una fase de “predecoding” en la que se extraería la información común a todos los logs en el estándar: fecha y hora, host y programa que lo produjo. Después de esto, el evento pasa por los decoders que adecuados para ese tipo de log. Por ejemplo, este sería el decoder por el que pasaría el log:

```
<decoder name="sshd-test">
  <program_name>sshd</program_name>
  <regex>^Accepted \S+ for (\S+) from (\S+) port </regex>
  <order>user, srcip</order>
</decoder>
```

Como se puede ver, el formato es XML. En él podemos ver: el nombre del decoder; la etiqueta “program_name” que especifica que este decoder solo se activa si el log viene producido por el programa especificado (en este caso ssh); la etiqueta “regex”, que indica la expresión regular con la que tiene que “encajar” el log para poder seguir adelante la fase de descodificación; y por último en la etiqueta “order” se especifican las variables en las que se almacenaran los elementos especificados en paréntesis en la expresión regular (wrappers).

El siguiente paso es buscar una regla cuyas condiciones se ajusten al contenido del log, al igual que con el decoder. La diferencia reside en que mientras que la función del decoder era la de “sacar” información relevante en varibles, la función de una regla es alertar o realizar alguna acción de respuesta activa. Un ejemplo de una regla sería:

```
<rule id="100124" level="5">
  <decoded_as>sshd</decoded_as>
  <match>^Failed password</match>
  <description>Failed SSHD password attempt</description>
</rule>
```

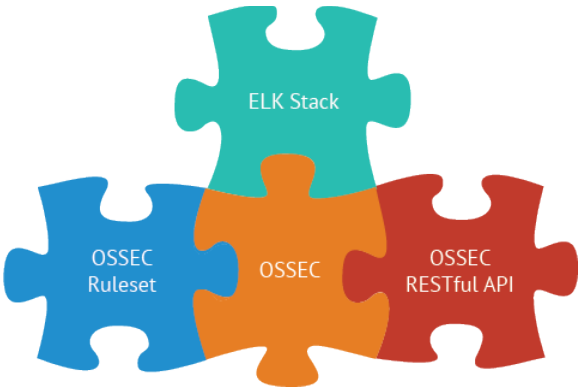
En la que las condiciones vienen definidas por las etiquetas XML: la cabecera indica la ID de la regla y el nivel de alerta, siendo 1 el mínimo y 15 el máximo; “decoded_as” que indica que la regla solo se activaría si el log ha pasado previamente por un decoder cuyo nombre coincide con el que se indica aquí; “match”, que indica la expresión regular que tiene que cumplirse en el log para que la regla siga adelante; y “description”, donde se indica el mensaje que va a ser indicado en la regla que se va a producir.

La alerta que se producirá tras la ejecución de esta regla es:

```
** Alert 1199140089.788: - syslog,sshd,authentication_failure, 2007 Apr 14
17:33:09 linux_server->/var/log/auth.log
Rule: 100124 (level 7) -> 'Failed SSHD password attempt'
Src IP: 192.168.2.180 User: lcid
Apr 14 17:33:09 linux_server sshd[1231]: Failed password for invalid user
lcid from 192.168.2.180 port 1799 ssh2
```

Wazuh

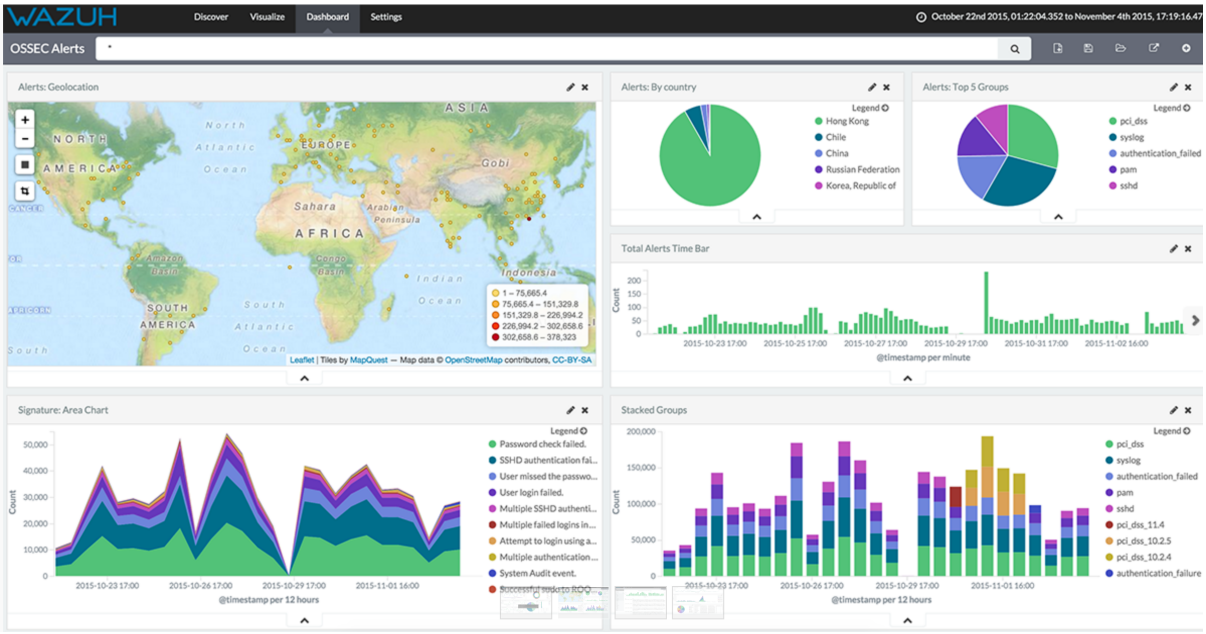
La empresa Wazuh, con sede en California, pero con una oficina en Granada, se dedica a ampliar la funcionalidad de OSSEC y a dar soporte comercial para empresas. Para ello, han hecho un "fork" de la herramienta y trabajan en 3 áreas principales: un "ruleset" ampliado, ELK stack y una API RESTful.



- El nuevo ruleset (conjunto de reglas) incluye cientos de reglas nuevas (además de mejorar las existentes) para logs que producen aplicaciones o servicios que no estaban contemplados en el conjunto original de OSSEC: Netscaler, Serv-U SolarWinds, Auditd, USB, etc.
- ELK stack: es un añadido compuesto por tres herramientas que producen una interfaz gráfica que permite la visualización de una gran cantidad de información sobre las alertas de una manera muy cómoda. Las tres herramientas que conforman el stack son:
    - Elasticsearch: motor de busqueda de texto completo, escalable y en tiempo real.
    - Logstash: herramienta para relcolectar logs, "parsearlos" y almacenarlos para un uso posterior.

- Kibana: un dashboard para la presentación de datos.

El resultado de la unión de estos 3 elementos es:



- RESTful API: Una API de peticiones REST que permite: añadir, eliminar o reiniciar agentes; recibir información del estado de los agentes, comprobar los resultados de syscheck y rootcheck; y exportar e importar "keys" de agentes para añadir elementos al despliegue.

Otras mejoras que ha producido la empresa son la integración con el sistema de Dockers, para poder hacer el despliegue de la herramienta de una manera mucho más sencilla y cómoda que instalarlo manualmente en el sistema; o la integración con Puppets, para tener un entorno configurado rápidamente con todo lo necesario para que OSSEC se ejecute.

Bibliografía:

Documentación oficial OSSEC – http://ossec.github.io/docs/
https://ossec-docs.readthedocs.io

Documentación Wazuh - http://documentation.wazuh.com/en/latest/about.html

Blog Wazuh http://blog.wazuh.com