Administración de Bitácoras de Dispositivos de Seguridad utilizando Aplicaciones de Orquestación

Reinaldo N. Mayol-Arnao
Email: mayol@ula.ve
Doctorado en Electrónica, Universidad Industrial de Santander
y
Centro Nacional de Cálculo Científico, Universidad de Los Andes, Venezuela

Luis A. Nuñez
Email: lnunez@uis.edu.co, nunez@ula.ve
Centro Virtual de Altos Estudios en Altas Energías, cevale2
Escuela de Física, Universidad Industrial de Santander
y
Centro Nacional de Cómputo Científico, Universidad de Los Andes, Venezuela

Jorge Luis Chacón Velazco
Email:jchacon@uis.edu.co
Grupo de Investigación en Energía y Medio Ambiente
Centro de Cálculo Científico Avanzado, Universidad Industrial de Santander, Colombia

Resumen

Se describe una prueba de concepto para la orquestación de aplicaciones de manejo de bitácoras y su posible relación con incidentes de seguridad informática en redes. La prueba se ilustra con un modelo capaz de obtener información, usando SNMP o Syslog, desde servidores o routers. El modelo está compuesto por pequeñas aplicaciones orquestadas mediante TAVERNA, que permiten obtener información referida a eventos, usuarios y fechas provenientes de las bitácoras de los dispositivos antes descritos.

Finalmente se comentan los resultados obtenidos, y las principales dificultades y limitaciones encontradas.

Abstract

We describe a proof of concepts for applications orchestration, used for devices log management.

The model is capable to obtain information, by SNMP or Syslog, from servers or routers. It is composed by small applications orchestrated by means of TAVERNA, which allow recollect information referred to events, users and dates from the devices logs.

Finally we comment the obtained results and the principal difficulties and limitations found.

I-Introducción

Un aspecto importante de las estrategias de seguridad de la información es la gestión de los datos (bitácoras[1] en lo sucesivo) que generan los dispositivos de seguridad y los dispositivos de comunicaciones y redes en general.[1]

Es imprescindible obtener información de las bitácoras de estos dispositivos, sin embargo, quizá el volumen de datos almacenados en ellos puede forzarnos a utilizar, entre otras, técnicas de minería de datos para descubrir patrones, tendencias, o incluso actos aislados, que en un determinado contexto pueden significar evidencias de un posible incidente de inseguridad. Esta es la base de los sistemas de Detección y Prevención de Intrusos. Existe una gran diversidad de dispositivos de seguridad (*firewalls*, detectores de intrusos, sistemas antivirus, servidores, etc.) que se encuentran distribuidos en ambientes de redes, los cuales proveen este tipo de información.

[1] logs

A continuación mencionamos algunas condiciones que modelan el tratamiento de las bitácoras en la actualidad:

1. No existe un estándar para la generación de las bitácoras, por lo que cada fabricante (o incluso cada modelo) genera información diferente y en formatos disímiles.

2. Incluso en el caso (poco común) de que una organización normalizara sus dispositivos de seguridad no le servirá de mucho pues, en un mundo interconectado e interdependiente como el actual, requerirá de datos de terceros.

3. Los volúmenes de datos son muy grandes, a modo de ejemplo, el rastreo de la actividad de una sola red local puede generar varios cientos de megabits de datos (texto) al día. [2]

4. El poder computacional necesario para el manejo de las bitácoras es grande y por regla general inexistente en la mayoría de las organizaciones.

5. Los datos son perecederos y su preservación se ve afectada por la duración y capacidad de los equipos que los generan, los interpretan o por la capacidad de la organización de almacenarlos.[3]

La situación descrita arriba es cotidiana. Vivimos la era de la información, la de los datos y los de sensores. Hay datos por doquier, en diferentes formatos, recolectados con objetivos disímiles, manejados por actores diversos, pero datos al fin. [4][5]

Los términos "ciberinfraestructura", "e-ciencia" y más recientemente uno más amplio, "e-investigación", han sido acuñados para describir nuevas formas de producción y diseminación del conocimiento (ver [6][7][8][9]y las referencias allí citadas). Uno de los retos que habremos de enfrentar en esta nueva manera de hacer ciencia es: manejar, administrar, analizar y preservar el "diluvio de datos"[9]. Esta avalancha de registros de todo tipo, viene generada por experimentos de escala mundial (aceleradores de partículas, red de observatorios terrestres y satelitales e infinidad de los más variados tipos sensores), desbordando toda capacidad de manejo que no sea mediante las TIC.

Pero esta avalancha de datos y la necesidad de convertirla en un flujo equivalente de información no se circunscriba solamente al ámbito de las ciencias. Cualquier actividad humana parece estar impregnada ineludiblemente de esta realidad. Por ejemplo, durante el año 2009, los aviones norteamericanos que sobrevolaron Iraq y Afganistán enviaron a tierra el equivalente a 24 años de video, pero durante el 2010 se espera que esta cantidad se multiplique por 10 y para el 2011 la escala será 3 veces superior. La cadena de tiendas Wal-Mart, posee una base de datos estimada en más de 2.5 petabytes: el equivalente a 167 veces los libros en la Biblioteca del Congreso de Estados Unidos.[5][6]

El tratamiento a esta avalancha de datos que obliga a la utilización de nuevos modelos de hacer ciencia[10], nos motiva a proponer una manera diferente de manejar la información de seguridad.

Dentro de las estrategias de manejo de la seguridad de la información, el manejo de las bitácoras tiene un papel preponderante. Pero obviamente este proceso es computacionalmente costoso y no puede seguir siendo realizado en un único punto.

Un enfoque para lograr el manejo de las bitácoras es la utilización de aplicaciones distribuidas para su acopio y procesamiento, incluyendo la caracterización mediante metadata, que es una de nuestras principales propuesta. Una alternativa es la creación de pequeños programas distribuidos a través de internet, y que puedan ser orquestados para lograr un fin común.

Este documento describe algunas pruebas de conceptos realizadas para la manipulación básica de bitácoras utilizando un programa de orquestación. Se describen algunos resultados preliminares, así como las principales limitaciones encontradas.

II- Los sistemas de orquestación de aplicaciones

En este documento utilizaremos los términos de **aplicaciones de orquestación** para referirnos a aquellos programas conocidos en la literatura con el término *"workflow*". La función fundamental de las aplicaciones de orquestación es la

automatización de procedimientos a través de los cuales los datos son recolectados, clasificados, normalizados, computados y transferidos entre elementos según un grupo definido de reglas.[13][14]

El uso de aplicaciones de orquestación trae las siguientes ventajas, obvias si se trata de un entorno como el que describimos para el manejo de bitácoras:

- Aislamiento de las intríngulis del cómputo.
- Posibilidad de construir aplicaciones dinámicas orquestando recursos locales y remotos.
- Facilidad de reutilización de código, que puede llegar incluso a la utilización de bibliotecas de servicios, aplicaciones, partes de estas o procesos, accesibles vía Web.

Entre las características importantes que deben ser tomadas en cuenta al diseñar aplicaciones generales formadas por pequeños programas orquestados se encuentran las siguientes [5]:

1. Forma de orquestación.
2. Modelos para el manejo de la información (tanto datos como de control)
3. Modelos para cronometrar recursos
4. Modelos para la tolerancia a fallas
5. Modelos para el manejo de la Calidad de Servicio.

Yu el at. en [15] hacen una clasificación de los procesos de orquestación y una exhaustiva comparación entre varios de estos sistemas tomando criterios como la organización de operaciones, la forma de obtener información de los recursos, el tipo de asignación, el manejo de fallas, etc.

Durante el uso de My Experiment[16], un repositorio abierto y cooperativo de programas de orquestación científica, hemos encontrado que no existen en su biblioteca proyectos asociados al manejo de seguridad de la información utilizando aplicaciones de orquestación para e-investigación. El uso de este tipo de repositorios globales es de gran importancia para la comunidad computacional en general, pues permite compartir recursos, ampliar los criterios de cooperación e incluso de validación de múltiples proyectos. Es decir, aquí hay un área de trabajo importante por explotar.

III-La experiencia con un sistema de orquestación de aplicaciones

Durante el proceso de prueba de conceptos para el manejo de bitácoras de dispositivos se decidió preparar un protocolo de pruebas dirigido a dos objetivos fundamentales:

1. Acopio de bitácoras desde dispositivos remotos
2. Minería básica de las bitácoras acopiadas.

IV-Acopio de Bitácoras

Para realizar el acopio de bitácoras se probaron 2 técnicas, no excluyentes y fuertemente dependientes de la plataforma. En principio se utilizó un mandato vía SNMP para indicar al dispositivo que enviara los datos a un repositorio central. En el caso de equipos servidores se utilizó la sincronización utilizando syslog.

Analizando los métodos utilizados es posible concluir que el segundo método es más flexible y seguro. En el caso del **SNMP** existen tres razones fundamentales para desechar su uso: En primer lugar implica serios riesgos de seguridad pues hay que utilizar la comunidad de escritura para indicar ese tipo de órdenes. En este orden de ideas es común referirse a dos tipos de problemas de seguridad: los primeros, superados desde la versión 2 del protocolo, se refieren a interceptaciones y manipulaciones de los datos y las órdenes mientras viajan por la red. En la actualidad esta situación no es de consideración en cualquier implementación correcta del sistema de monitoreo. Sin embargo, todavía persisten los riesgos reales, suspicacias y temores a la posibilidad de la manipulación remota de la configuración de los dispositivos. Esta situación tiene una implicación adicional; en la mayoría de las organizaciones los equipos de comunicaciones están configurados para no responder a las solicitudes **SNMP** hechas con la comunidad de escritura. Por último, la utilización de este método implica también limitaciones en cuanto el tipo de equipos que soportan este tipo de transferencias (mayormente dependiente de la implementación de **SNMP**). Sin embargo, como prueba de concepto es una opción válida que, además, permitió evaluar las posibilidades reales

de la aplicación de orquestación para el manejo de aplicaciones de red en tiempo real.

El otro mecanismo de acopio utilizado es ampliamente conocido: ***syslog***, un estándar de facto para el envío de mensajes de registro sobre IP. Un mensaje de registro puede contener cualquier información de un servidor o un equipo de comunicaciones. La mayoría de los sistemas operativos incluyen algún tipo de interfaz con ***syslog***, incluyendo aquellos de dispositivos de comunicaciones. En nuestra prueba de conceptos utilizamos ***syslog*** para concentrar las bitácoras en un repositorio central, desde donde hicimos algunos ejercicios de organización y minería de datos sobre las bitácoras colectadas.

En este caso, adicionalmente, y pensando en una implementación real tiene otra ventaja. En los entornos productivos es común que se concentren, siguiendo un esquema preestablecido, las bitácoras en un repositorio central, aunque sea para su respaldo.

Con este análisis en mente, se decidió utilizar finalmente syslog como mecanismo de recolección de las bitácoras.

V-Implementación de mecanismos de búsqueda orquestados.

Una vez clara la necesidad de implementar sistemas distribuidos para el manejo de las bitácoras, se decidió crear una estructura de módulos que pudiesen ser orquestados.

Para la realización de los módulos de búsqueda se utilizó un lenguaje interpretado (considerando que para esta prueba de conceptos el rendimiento del sistema no es un elemento significativo). Como programa de orquestación de los módulos de rastreo se seleccionó TAVERNA de Manchester University.[7]

Para la selección de la herramienta de orquestación se tomaron en cuenta los siguientes criterios:

- Plataforma donde se puede ejecutar,
- Forma de la especificación del modelo (buscando que fuese lo más abstracta posible de forma que pudiese ser migrada sin grandes problemas de plataforma, pero que a su vez fuese posible acceder al *hardware* cuando fuese necesario),
- Forma en que se organizan los elementos del modelo (buscando que pudiesen ser directamente manipuladas por el usuario),
- Cantidad de Información disponible.

Otros elementos como el manejo de comunicaciones, datos y QoS fueron deliberadamente obviados en este nivel de pruebas y deberán ser tomados en cuenta en diseños posteriores.

Para realizar las pruebas del modelo se utilizó una red formada por 4 equipos servidores, con sistema operativo GNU-LINUX (Debian) interconectados por una red dedicada capa 2. 3 de estos equipos se utilizaron para el montaje de los servicios monitoreados, mientras que la última de las estaciones se utilizó para la colección, mediante syslog, de las bitácoras seleccionadas.[2] Las máquinas tenían velocidades entre 1.6 -2.0 Mb/s y 2Gb de RAM.

Como objetivos del modelo creado se propuso un sistema capaz de obtener datos desde un repositorio central, filtrarlos eliminando elementos pre-establecidos como no significativos, y organizarlos en las siguientes categorías:

- Tipos de registros,
- Frecuencia de ocurrencia de los registros,
- Usuarios involucrados en los registros,
- Fecha de los registros

En la Figura 1 se muestra el esquema general del modelo creado.

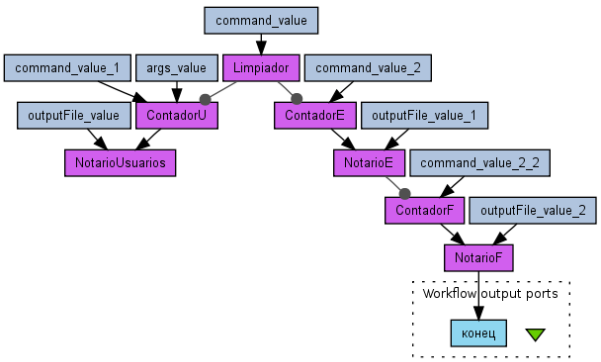

Figura1: Modelo General del Workflow para manejo de bitácoras

[2] Se instalaron los siguientes servicios: Apache Web, SMTP, PTR

En el esquema mostrado en la Figura 1 se distinguen 3 tipos de módulos: Limpiador, Contadores y Notarios.

En primer lugar, el Módulo **Limpiador** se encarga de la depuración de las bitácoras, eliminando de éstas algunos elementos que no interesaban en este nivel de diseño tales como comentarios, detalles secundarios de los eventos, trazas de tráfico o localizaciones de memoria y registros incompletos.

Adicionalmente, antes del comienzo del trabajo se realizó la caracterización de la estructura de los eventos que habían sido seleccionados de forma que pudiesen ser reconocidos por el módulo de limpieza. Todas los registros que no coincidían con los seleccionados fueron eliminados.

Los eventos seleccionados corresponden a las siguientes categorías:

1. Inicio-parada de Sistema Operativo.
2. Inicio-parada de servicios (ya fue descrito que servicios fueron instalados).
3. Envío-recepción de correos electrónicos.
4. Acceso al servidor web.
5. Inicio del proceso de impresión.
6. Errores en el manejo de los servicios.

Como resultado de la ejecución de este módulo se redujo en un 27% el tamaño bruto de las bitácoras seleccionadas[3]. Aunque no era parte de la prueba a este nivel, un trabajo importante en el futuro será evaluar la efectividad de este proceso.

El segundo tipo de módulo son los llamados **Contadores**. Estos son el centro de la operación del modelo, ya que en ellos se realizan las operaciones de búsqueda y ordenamiento de los datos.

Dentro de los módulos contadores se distinguen 3 funcionalmente diferentes (E,U,F). Mientras el módulo **ContadorE** se encarga de buscar y contar eventos (ver Figura 2), el módulo **ContadorU** (ver Figura 3) se encarga de obtener información de usuarios y el módulo **ContadorF** hace conteos por fecha de los eventos (véase Figura 4).

La base del algoritmo es la búsqueda lineal de los patrones seleccionados y que hemos descrito anteriormente. El algoritmo, en seudo-codigo, se muestra a continuación:

[3] Se pasó de 3016 registros reconocidos a aproximadamente 2200 registros filtrados.

```
Contador (usuario,fecha,eventos)
Leer_linea( línea)
SI leer_linea = última entonces
  Salir(0);
De lo contrario
 Buscar _patrón _coincidente;
 Conteo++;
IS
```

Como entrada al algoritmo **Contador** está un parámetro que define el uso (conteo de usuarios, fechas o eventos). Es decir, mismo algoritmo sirve para cualquiera de los nodos contadores (U,F,E). El algoritmo utiliza otras funciones: *leer_linea* (que permite leer de la bitácora una a una las líneas de búsqueda) y *Buscar_Patrón_Coincidente* que permite extraer la parte de la línea que se busca (usuarios, fechas o eventos) . Cada vez que se encuentra un nuevo subpatrón (por ejemplo un nuevo usuario) el programa escribe una nueva entrada y le agrega un contador (separado del subpatron por :) Las Figuras 2, 3 y 4 muestran las salidas de cada uno de los módulos Contadores E, U y F respectivamente.

Refinar los algoritmos de búsqueda, mejorar su eficiencia y portabilidad es parte de las tareas futuras.

```
00 afrodita kernel: [
0.397096] io scheduler
noop registered:210
00 afrodita kernel: [
0.397099] io scheduler
anticipatory
registered:210
00 afrodita kernel: [
0.397101] io scheduler
```

Figura 2: Extracto de la salida del contador de eventos. El número final luego de los dos puntos indica la cantidad de eventos detectados.

```
root,210

reinaldo,3600

admin,34

syslog:345
```

Figura 3: Extracto de la salida del contador de usuarios. El número luego de la coma indica la cantidad de eventos asociados al usuario.

```
Jun 25: 29
Jun 24: 234
Jun 23: 121
Jun 22: 91
Jun 20: 233
```

Figura 4: Extracto de salida del contador de fechas. El número luego de los dos puntos indica la cantidad de eventos ocurridos en cada día.

Finalmente se distinguen los módulos **Notarios**. Estos son los encargados de escribir en disco, en formato XML, los resultados de la ejecución de los módulos **Contadores**.

VI-Conclusiones y trabajo futuro

Logramos un mecanismo, que aunque primario, satisface las expectativas de las pruebas conceptuales de la utilización de un esquema de orquestación de aplicaciones para la manipulación de bitácoras de servidores.

La manipulación, con los esquemas y la plataforma descritos, tuvo un alto consumo computacional. El sistema se ejecutó en aproximadamente 45 minutos. Evidentemente el paso a plataformas de cómputo intensivo y la mejora de los algoritmos es imperativo. Sin embargo, el rendimiento no era un parámetro crucial al momento de planificar las pruebas que lo que buscaban fundamentalmente es mostrar la posibilidad de utilizar sistemas de orquestación con nuevos propósitos.

Adicionalmente el hecho de tener que preseleccionar los patrones de búsqueda, concordante con lo planteado por otros autores, es un limitación importante del modelo.

En este proceso hemos probado las posibilidades que ofrece la creación de pequeñas aplicaciones que realicen funciones básicas pero que al ser correctamente orquestadas pueden generar grandes cantidades de información.

En la prueba de conceptos hemos reconocido algunas de las limitaciones del sistema de orquestación utilizado, entre ellos:

- El lenguaje en que está escrita la aplicación limita su rendimiento.[4]
- No hay acceso a elementos importantes de la orquestación en entornos distribuidos tales como control de fallos, errores, manejo de comunicaciones, elementos de seguridad como autenticación de aplicaciones, entre otros.[5]
- Algunos módulos pre-creados y creados por la comunidad tienen limitaciones en su usabilidad. Por ejemplo, la posibilidad de ejecutar programas en el shell de comandos acciones desde el workflow está limitada a algunos comandos y prácticamente sin ningún argumento.

Los pasos siguientes de nuestra investigación deberán estar asociados a:

- Migrar a un sistema de orquestación que de acceso a todos los parámetros de orquestación que hemos comentado en este documento.
- Finalmente, durante las pruebas una de las dificultades fundamentales a vencer estuvo, como era esperado, en la falta de normalización entre las bitácoras. Por lo que puede definirse como uno de los elementos fundamentales de trabajo futuro la creación de mecanismos de búsqueda, depuración y extracción de información más eficientes y eficaces que los utilizados en este experimento.

VII- Referencias

[1] W.H. Baker and L. Wallace, "Is Information Security Under Control?," , *IEEE Security & Privacy*, 2007.

[2] B.J. Jansen, I. Taksa, and A. Spink, "Research and Methodological Foundations of Transaction Log Analysis," *Behaviorism*, pp. 1-16.

[4] Java JDK 1.4

[5] Taverna fue concebido sólo como un sistema para aislar a científicos de las intríngulis del cálculo científico.

[3] B.J. Jansen and A. Spink, "Foundations of Transaction Log Analysis," *Behaviorism*, pp. 1-16.

[4] "Technology: The data deluge," *The Economist*, 2010.

[5] "Data, data Everywhere," *The Economist*, 2010.

[6] T. Hey and A.E. Trefethen, "Cyberinfrastructure for e-Science.," *Science (New York, N.Y.)*, vol. 308, 2005, pp. 817-21.

[7] I. Foster, "Service-oriented science," *Science*, pp. 814-817.

[8] T.A. Hey Tony, "Data Deluge: An e-Science Perpective," *Annals of Physics*, vol. 54, 2003, p. 258.

[9] T.A. Hey Tony, "E-Science and its implications," *Phil.Trans. r. Soc Lond A.*, 2003, pp. 1809-1825.

[10] Abbott Mark R., "A New Path for Science?," *The 4th Paradigm: Data-Intensive Scientific Discovery*, T. Hey, S. Tansley, and K. Tolle, Microsoft Research, 2009, pp. 111-116.

[11] R. Gabriel, T. Hoppe, A. Pastwa, and S. Sowa, "Analyzing Malware Log Data to Support Security Information and Event Management: Some Research Results," *2009 First International Confernce on Advances in Databases, Knowledge, and Data Applications*, 2009, pp. 108-113.

[12] M. Meng, "Network Security Data Mining Based on Decomposition Wavelet," *Proceedings of the 7th World Congress on Intelligent Control and Automation*, Chongqing, China: , pp. 6646-6649.

[13] Y. Gil, E. Deelman, J. Blythe, C. Kesselman, and H. Tangmunarunkit, "Artificial intelligence and grids: Workflow planning and beyond," *IEEE Intelligent Systems*, vol. 19, 2004, p. 26–33.

[14] J. Cao, M. Li, W. Wei, and S. Zhang, "A Distributed Re-configurable Grid Workflow Engine," *Computational Science–ICCS 2006*, 2006, p. 948–955.

[15] J. Yu and R.B. A, "Taxonomy of Workflow Management Systems for Grid Computing," *Technical Report GRIDS-TR*, pp. -2005-1.

[16] "My Experiment Project.."