Universidad Nacional del Litoral

Proyecto Final de Carrera

Diseño de un sistema de detección de anomalías en redes de computadoras.

Pineda Leandro

dirigido por Ing. Miguel Angel Robledo
codirigido por Ing. Gabriel Filippa

Repositorio git del documento (solo versión borrador):
https://github.com/leandropineda/anteproyecto

Santa Fe
28 de agosto de 2016

Justificación

La cantidad de servicios que organismos públicos y privados ofrecen a través de Internet aumenta constantemente, y en consecuencia también lo hace la complejidad de los sistemas necesarios para proveerlos.

La integridad de estos sistemas puede verse comprometida principalmente por dos factores: las fallas de los componentes de la infraestructura de red[1], que afectan a una parte o a la totalidad de las funcionalidades del sistema, y los numerosos tipos de ataques[2][3] a los servicios públicos ofrecidos a los usuarios. Hacer frente a estos últimos es de extrema importancia dado que su fin puede ser no solo afectar la calidad de los servicios ofrecidos, sino que también pueden comprometer información sensible de la organización. Además de ataques externos, los sistemas pueden ser vulnerados por usuarios malintensionados con acceso físico a la red interna de la organización, o por actores inadvertidos[4][5][6][7].

Utilizar únicamente el conjunto de herramientas clásicas para implementar políticas de seguridad como firewall, lista de control de acceso, credenciales de usuario, proxy de red entre otros, resultaba adecuado 10 años atras, pero en la actualidad estas herramientas están siendo cada vez menos efectivas en la tarea de bloquear ataques dirigidos y malware avanzados. Esto hace que cada vez sea más común la utilización de otras herramientas que monitorean continuamente eventos en busca de patrones que puedan indicar comportamientos anómalos en el tráfico de red, provocados tanto por ataques como por fallas en los dispositivos de red. Los sistemas de detección/prevención de intrusiones (IDS o IPS por las siglas en inglés *Intrusion Detection/Prevention System*) pueden realizar esta tarea utilizando alguna de estas técnicas de detección: detección de uso indebido o *signature-based detection* y detección de anomalías o *anomaly-based detection*[8]. En el primer caso, el sistema determina la ocurrencia de ataques comparando la actividad de la red y de los subsistemas que la componen, contra un conjunto de patrones de ataques conocidos. Si los patrones coinciden, el ataque es informado y pueden tomarse medidas al respecto. En el segundo caso, se modela el comportamiento normal de la red y se usa como base para buscar comportamientos anómalos, los cuales pueden ser producto de fallas o de potenciales ataques.

Como alternativa a las soluciones de hardware dedicado o sistemas privativos que ofrecen algunas empresas, la comunidad de software libre desarrolla y mantiene la aplicación Snort[1], que es un IDS basado en detección de uso indebido. Si bien esta herramienta es efectiva detectando ataques conocidos, es dependiente de la definición de un conjunto efectivo de reglas por parte del administrador (las cuales son difíciles de mantener) y de una base de datos actualizada con los patrones de ataques o *malware*. Otra desventaja de esta técnica de detección es la imposibilidad de detectar ataques *zero-day*[2][8]. Otra herramienta que suele utilizarse en conjunto con Snort es Bro[3], que es una plataforma para análisis de tráfico de red. Sin embargo, Bro también se basa en la definición de reglas de monitoreo y utiliza detección de uso indebido.

Con el fin de disponer de otra herramienta para la identificación de comportamientos sospechosos que complemente las funcionalidades de las que se utilizan en la actualidad, el objetivo de este proyecto es el de diseñar un software que modele de forma automática el comportamiento normal de la red utilizando las técnicas del estado del arte, para identificar y alertar comportamiento anómalo en la infraestructura de red de datos de la Secretaría de Tecnologías para la Gestión de la ciudad de Santa Fe. El modelo de comportamiento normal es calculado automaticamente por el software, brindando la posibilidad de detectar efectivamente cualquier evento poco usual en los sistemas. Para esto, se hará uso de la información generada por los dispositivos de

[1] https://www.snort.org/

[2] ataques que explotan vulnerabilidades que aún no han sido conocidas o que no están contempladas en al conjunto definido de reglas

[3] https://www.bro.org/

infraestructura de red como routers y los logs de servidores y de los subsistemas que interactuan dentro de la red.

Objetivos

Generales

Diseñar un sistema de detección de intrusiones basado en técnicas de detección de anomalías.

Específicos

- Identificar un conjunto de tecnologías y herramientas adecuadas para el diseño del sistema.
- Determinar las técnicas de modelado de tráfico de red necesarias.
- Implementar un software que modele el tráfico normal de red de forma automática, utilizando técnicas no supervisadas.
- Implementar un software que genere alertas ante la ocurrencia de eventos anómalos.
- Diseñar un sistema que evite clasificar falsos negativos. Este objetivo es de particular importancia dado que un evento anómalo clasificado como falso negativo es un evento potencialmente dañino considerado como normal.
- Elaborar un documento que describa la arquitectura del sistema.

Alcances

Funcionales

- El sistema utilizará información de capa de red[9] y capa de transporte[10][11], e información proveniente de logs de servidores web y gestores de bases de datos.
- Los datos serán procesados utilizando técnicas de *Streaming*[12]
- El sistema identificará posibles comportamientos anómalos y generará las alarmas correspondientes.
- Con cada alarma, el sistema mostrará que probabilidad existe de que el evento sea un falso positivo.
- El sistema proveerá al usuario una interfaz web para la visualización de los eventos y las alarmas.
- El sistema almacenará información de eventos ocurridos en archivos de texto plano y en formato semi-estructurado.
- El sistema no utilizará técnicas de detección *signature-based*.

No Funcionales

- Se proveerá un manual de instalación y configuración.

Exclusiones

El proyecto no contempla la instalación del sistema en un ámbito de producción.

Supuestos

Los datos necesarios para realizar pruebas pueden ser generados arbitrariamente. Además, se utilizarán datos provistos por el Centro de Cómputos de la Provincia de Santa Fe.

Criterios de Aceptación

Se considera que el proyecto está aceptado cuando cumple en un 90 % los requisitos funcionales, con un nivel mínimo de aceptación del 75 %. Los prototipos deben tener implementadas todas las funcionalidades planificadas.

Metodología

La metodología de desarrollo del proyecto será incremental e iterativa. Incremental porque varios componentes y funcionalidades del sistema se desarrollarán en momentos diferentes y serán integradas cuando sean completadas. Iterativa pues se invertirán esfuerzos en revisar constantemente partes del sistema, tanto para mejorar la calidad externa como interna del software[13].

Dado que los requerimientos de los interesados pueden cambiar en el transcurso de la ejecución del proyecto, se propone utilizar un enfoque de desarrollo ágil. El mismo contempla la posibilidad de cambios en los alcances y en las prioridades de las diferentes funcionalidades, y fundamentalmente promueve la entrega continua de software y la inclusión de los interesados en el proceso de desarrollo.

Plan de Tareas

El proyecto se divide en 5 incrementos. A continuación se da una breve descripción de los mismos:

Incremento 1: Investigación preliminar La primer parte del proyecto consiste en determinar que conjunto de tecnologías serán utilizadas, y elaborará una descripción a alto nivel de las diferentes componentes del sistema. Además se realizará una investigación sobre las técnicas de modelado de comportamiento existentes con el fin de determinar cuales serán implementadas.

Incremento 2: Modelado de comportamiento En esta etapa se implementarán algunas de las técnicas de modelado de comportamiento seleccionadas. Se utilizarán solo los datos provenientes de la capa de transporte y se entregará un software con una interfaz web sencilla donde se muestren los principales parámetros del modelado.

Incremento 3: Detección de anomalías Se implementarán las funcionalidades de detección de anomalías y se agregará al software mecanismos de generación de alarmas. Además se mejorará la interfaz de usuario.

Incremento 4: Modelado de comportamiento de subsistemas Con el fin de mejorar la identificación de anomalías se incorporará al modelo de comportamiento información de los diferentes subsistemas que componen la infraestructura de red.

Incremento 5: Pruebas exhaustivas y documentación Se completará el desarrollo del software y se redactará la documentación necesaria. Dado que la tasa de falsos negativos[4] debe ser mínima, se realizarán pruebas exhaustivas de funcionamiento. Una vez concluidas las pruebas, se elaborará un informe con los resultados obtenidos

Al finalizar la primera iteración de cada incremento se obtiene una herramienta de software con las funcionalidades descriptas y calidad de producto final, con excepción de la etapa de investigación preliminar donde se obtendrá un informe en soporte escrito o digital.

Entregable	Fecha de entrega
Informe de avance 1	04/10/2016
Informe de avance 2	08/11/2016
Informe de avance 3	16/12/2016
Informe de avance 4	14/02/2017

Cuadro 1: Fechas de entrega de informes de avance.

Plan de Tareas

La duración total del proyecto es de 466 horas, con una dedicación de 20 horas semanales. A continuación se detalla el plan de tareas.

1. **Investigación preliminar** (66hs)
   1.1. Estudio comparativo de las tecnologías y métodos de modelado. (24hs)
   1.2. Diseño conceptual del sistema. (30hs)
   1.3. Documentación. (12hs)
2. **Modelado de comportamiento** (84hs)
   2.1. Instalación y configuración de la plataforma de desarrollo. (12hs)
   2.2. Implementación de funcionalidad de captura de tráfico de red. (24hs)
   2.3. Implementación de modelado de tráfico. (24hs)
   2.4. Implementación de intefaz web. (24hs)
3. **Detección de anomalías** (98hs)
   3.1. Implementación de funcionalidad de detección de anomalías. (30hs)
   3.2. Pruebas de implementación. (24hs)
   3.3. Implementación de módulo de alarmas. (24hs)
   3.4. Mejora de interfaz web. (20hs)

[4]La existencia de falsos negativos implica la existencia de ataques verdaderos que no son detectados por el IDS

4. **Modelado de comportamiento de subsistemas** (146hs)
   4.1. Implementación de funcionalidad de captura de logs de subsistemas. (72hs)
      - Servidores web (24hs).
      - Gestores de base de datos (24hs).
      - Firewalls (24hs).
   4.2. Implementación de modelado de comportamiento de subsistemas. (30hs)
   4.3. Pruebas de implementación. (24hs)
   4.4. Implementación de funcionalidad de detección de anomalías y alarmas. (20hs)
5. **Pruebas exhaustivas y documentación** (72hs)
   5.1. Pruebas de detección. (20hs)
   5.2. Elaboración de informe de desempeño. (12hs)
   5.3. Elaboración de informe final. (40hs)

Informes de avance

Se presentarán 4 informes de avance en las fechas de finalización de cada etapa, detalladas en el Cuadro 2. A continuación se detalla que información será incluida en cada informe:

Informe de avance 1 Contendrá los resultados obtenidos en los estudios comparativos de las tecnologías y las técnicas de modelado y detección y justificará la elección de las mismas. Además se incluirá una descripción general de la arquitectura del sistema.

Informe de avance 2 Contendrá información sobre los criterios de selección de características para el modelado de tráfico de red. Se proveerá la guía de instalación y configuración de la plataforma. Además tendrá información sobre cambios realizados en los entregables anteriores y el desempeño del modelo implementado.

Informe de avance 3 Contendrá información sobre los criterios de selección de características para el modelado del comportamiento de los subsistemas, y el desempeño del modelo implementado. Además se detallarán los cambios realizados en los entregables anteriores.

Informe de avance 4 Se describirán las pruebas de integración del sistema. Además tendrá información sobre cambios realizados en los entregables anteriores y el desempeño general del sistema. También se incluirán los resultados de las pruebas de detección de la última iteración.

Etapa	Inicio	Finalización	Duración
Investigación preliminar	01/09/2016	30/09/2016	4 semanas
Modelado de comportamiento	03/10/2016	04/11/2016	4 semanas
Detección de anomalías	07/11/2016	16/12/2016	5 semanas
Modelado de comportamiento de subsistemas	19/12/2016	18/02/2017	8 semanas
Pruebas exhaustivas y documentación	20/02/2017	25/03/2017	5 semanas

Cuadro 2: Fechas estimativas de inicio y fin de actividades.

Diagrama de Gantt

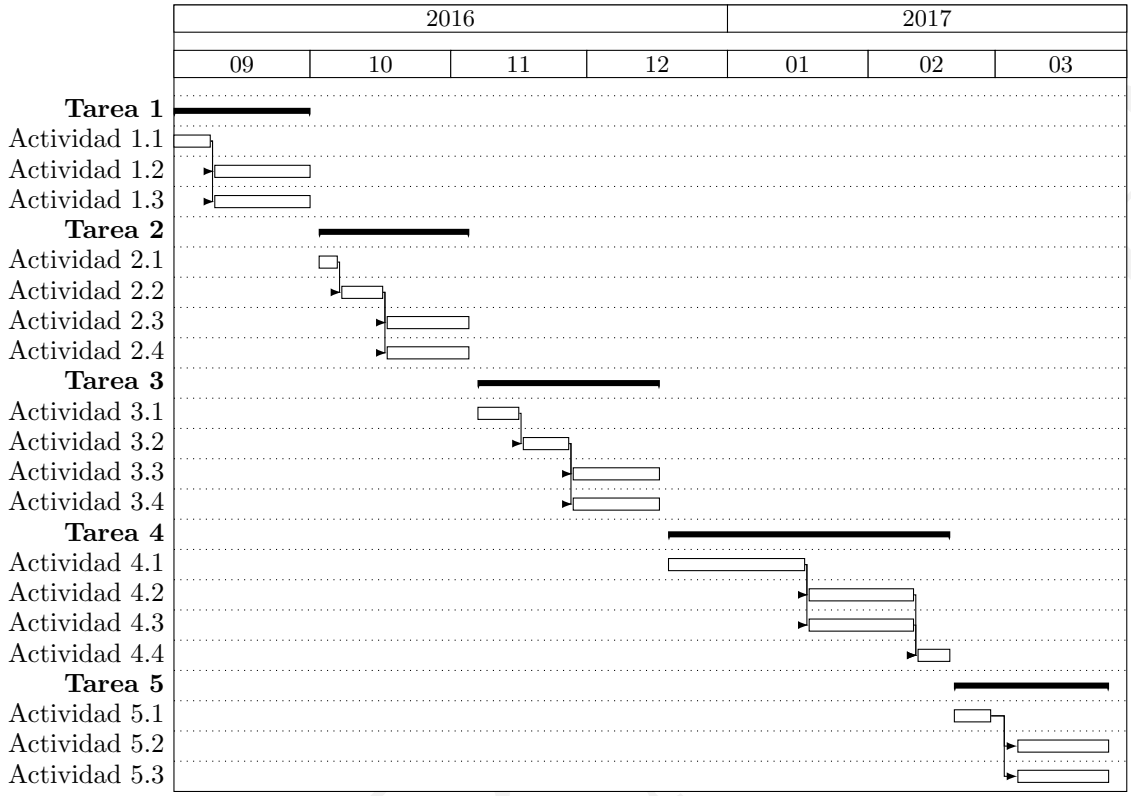

Figura 1: Diagrama de Gantt del proyecto

Bibliografía

[1] P. Gill, N. Jain y N. Nagappan, "Understanding network failures in data centers: Measurement, analysis, and implications", *SIGCOMM Comput. Commun. Rev.*, vol. 41, n.º 4, págs. 350-361, ago. de 2011, ISSN: 0146-4833. DOI: 10.1145/2043164.2018477. dirección: http://doi.acm.org/10.1145/2043164.2018477.

[2] S. Karumanchi y A. C. Squicciarini, "In the wild: A large scale study of web services vulnerabilities", en *Proceedings of the 29th Annual ACM Symposium on Applied Computing*, ép. SAC '14, Gyeongju, Republic of Korea: ACM, 2014, págs. 1239-1246, ISBN: 978-1-4503-2469-4. DOI: 10.1145/2554850.2555010. dirección: http://doi.acm.org/10.1145/2554850.2555010.

[3] P. Mutchler, A. Doupé, J. Mitchell, C. Kruegel y G. Vigna, "A large-scale study of mobile web app security", en *Proceedings of the Mobile Security Technologies Workshop (MoST)*, mayo de 2015.

[4] "Human errors and violations in computer and information security: The viewpoint of network administrators and security specialists", *Applied Ergonomic*,

[5] S. Kraemer, P. Carayon y J. Clem, "Human and organizational factors in computer and information security: Pathways to vulnerabilities", *Computers & Security*, vol. 28, n.º 7, págs. 509-520, 2009, ISSN: 0167-4048. DOI: http://dx.doi.org/10.1016/j.cose.2009.04.006". dirección: http://www.sciencedirect.com/science/article/pii/S0167404809000467.

[6] D. Liginlal, I. Sim y L. Khansa, "How significant is human error as a cause of privacy breaches? an empirical study and a framework for error management", *Computers & Security*, vol. 28, n.º 3–4, págs. 215 -228, 2009, ISSN: 0167-4048. DOI: http://dx.doi.org/10.1016/j.cose.2008.11.003. dirección: http://www.sciencedirect.com/science/article/pii/S0167404808001181.

[7] M. Ahmed, L. Sharif, M. Kabir y M. Al-maimani, *Human errors in information security*, 2012.

[8] A. Milenkoski, M. Vieira, S. Kounev, A. Avritzer y B. D. Payne, "Evaluating computer intrusion detection systems: A survey of common practices", *ACM Comput. Surv.*, vol. 48, n.º 1, 12:1-12:41, sep. de 2015, ISSN: 0360-0300. DOI: 10.1145/2808691. dirección: http://doi.acm.org/10.1145/2808691.

[9] J. Postel, *Internet protocol*, RFC 791 (INTERNET STANDARD), Updated by RFCs 1349, 2474, 6864, Internet Engineering Task Force, sep. de 1981. dirección: http://www.ietf.org/rfc/rfc791.txt.

[10] ——, *Transmission control protocol*, RFC 793 (INTERNET STANDARD), Updated by RFCs 1122, 3168, 6093, 6528, Internet Engineering Task Force, sep. de 1981. dirección: http://www.ietf.org/rfc/rfc793.txt.

[11] T. Socolofsky y C. Kale, *Tcp/ip tutorial*, RFC 1180 (Informational), Internet Engineering Task Force, ene. de 1991. dirección: `http://www.ietf.org/rfc/rfc1180.txt`.

[12] F. Fischer, F. Mansmann y D. A. Keim, "Real-time visual analytics for event data streams", en *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, ép. SAC '12, Trento, Italy: ACM, 2012, págs. 801-806, ISBN: 978-1-4503-0857-1. DOI: `10.1145/2245276.2245432`. dirección: `http://doi.acm.org/10.1145/2245276.2245432`.

[13] ISO/IEC, *ISO/IEC 9126. Software engineering – Product quality.* ISO/IEC, 2001.

[14] M. Xie, S. Han, B. Tian y S. Parvin, "Anomaly detection in wireless sensor networks: A survey", *J. Netw. Comput. Appl.*, vol. 34, n.º 4, págs. 1302-1325, jul. de 2011, ISSN: 1084-8045. DOI: `10.1016/j.jnca.2011.03.004`. dirección: `http://dx.doi.org/10.1016/j.jnca.2011.03.004`.

[15] A. Tanenbaum, *Computer Networks*, 4th. Prentice Hall Professional Technical Reference, 2002, ISBN: 0130661023.

[16] W. Stallings, *Data and Computer Communications (5th Ed.)* Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1997, ISBN: 0-02-415425-3.

[17] Q. Anderson, *Storm Real-Time Processing Cookbook*. Packt Publishing, 2013, ISBN: 1782164421, 9781782164425.

[18] P. Hunt, M. Konar, F. P. Junqueira y B. Reed, "Zookeeper: Wait-free coordination for internet-scale systems", en *Proceedings of the 2010 USENIX Conference on USENIX Annual Technical Conference*, ép. USENIXATC'10, Boston, MA: USENIX Association, 2010, págs. 11-11.

[19] F. Junqueira y B. Reed, *ZooKeeper: Distributed Process Coordination*. O'Reilly Media, 2013, ISBN: 9781449361280.

[20] J. Leibiusky, G. Eisbruch y D. Simonassi, *Getting Started with Storm*. O'Reilly Media, Inc., 2012, ISBN: 1449324010, 9781449324018.