Machine Learning Classifiers to Detect Malicious Websites

Christian Urcuqui
Grupo de Investigación i2t
Universidad Icesi
Cali, Colombia
ccurcuqui@icesi.edu.co

Andres Navarro
Grupo de Investigación i2t
Universidad Icesi
Cali, Colombia
anavarro@icesi.edu.co

Jose Osorio
Universidad Icesi
Cali, Colombia
jose.osorio1@correo.icesi.edu.co

Melisa García
Universidad Icesi
Cali, Colombia
melisa.garcia@correo.icesi.edu.co

Abstract

A risk that exists in Internet is the access of websites with malicious content, because they might be open doors for cybercrimes or be the mechanism to download files in order to affect organizations, persons and the environment. What is more, the attack registers through websites have been part of cyberattacks reports during the last years; this information includes attacks made by the currently risks found in new technologies, such as the IoT. Due the computer security complexity, studies have been working in to use machine learning algorithms to identify web malicious content. This article explores the application of a data analysis process through a framework that includes dynamic, static analysis, updated websites and a low interaction client honeypot in order to classify a website. Furthermore, it evaluates the capacity of the classification of four machine learning through the information analyzed.

1. Introducción

Internet es una tecnología que conecta alrededor de 3 mil millones de usuarios en todo el mundo, además, es un medio que permite tanto a las personas como a las compañías realizar distintas tareas, como por ejemplo, la difusión de información sensible y el acceso a las páginas web. Por otra parte, mientras el uso de las tecnologías se ha incrementado significativamente, la cantidad de vulnerabilidades y de ataques cibernéticos también ha aumentado.

Actualmente se pueden encontrar distintas técnicas, metodologías y sistemas para el análisis de páginas maliciosas [Urcuqui16]. Un ejemplo de ello es la aplicación del análisis estático, dinámico y la inteligencia artificial para la evaluación de elementos que permitan clasificar entre una página benigna y otra de contenido perjudicial para los usuarios. Adicionalmente, también se han propuesto marcos de trabajo para el análisis de sitios web [Bartoli10] [Roesch99]. Por otra parte, como sistemas de seguridad se pueden encontrar: Tripware [Kim94], Nagios [Aman14], entre otros.

2. Estado del arte

Los ataques web pueden ser evaluados a partir de dos enfoques: una detección basada en firmas y otra por anomalías.También, se pueden encontrar trabajos que utilizan algoritmos de machine learning para la detección de páginas web maliciosas que tienen contenido que está relacionado a un tipo de ciberataque [Atienza15].

Roesh M., explica en su estudio [Roesch99] la utilidad de Snort, un sistema de detección de intrusos que utiliza un conjunto de reglas almacenadas en su base de datos para la detección de contenido malicioso en la red.

Un estudio propone un marco de trabajo (Goldrake)

Proceedings of the Spring School of Networks, Pucón, Chile, October 2017, published at http://ceur-ws.org

[Bartoli10] para el análisis de cambios no autorizados sobre páginas web con alto contenido dinámico. Goldrake utiliza una detección por anomalías a través de un servicio de monitoreo que no requiere alguna instalación sobre la infraestructura del sitio web a analizar. Los resultados de la evaluación muestran que el prototipo tiene un buen desempeño tanto en la tasa de falsos positivos y falsos negativos.

Dos estudios han explorado la aplicación de algoritmos de machine learning para la identificación de páginas web maliciosas a partir de la extracción de información del análisis estático y dinámico. En [Mohaisen15] se entrenaron y testearon clasificadores a través de la información obtenida de los objetos transferidos (TF) en el tráfico HTTP tanto de páginas malignas y benignas, con un resultado de detección de 93 % en páginas web maliciosas. Por otra parte, en [Xu13] se realizo un análisis de información obtenida en la capa de aplicación y de red, para ello se utilizo un honeypot de alta interacción (Capture-HPC versión 3.0) y el sniffer TCPDUMP; obteniendo que el clasificador J48 aplicado con todos los datos es mucho más rápido y eficiente que un enfoque tanto dinámico y estático.

3. Metodología

1. *Dataset de URL.* El conjunto de enlaces utilizado se encuentra conformado por sitios maliciosos obtenidos de las siguientes fuentes: machinelearning.inginf.units.it/data-and-tools/hidden-fraudulent-urls-dataset, malwaredomainlist.com y zeuztacker.abuse.ch; de las anteriores se consiguieron un total de 185.181 enlaces. Por otra parte, los enlaces benignos se extrajeron del repositorio https://github.com/faizann24/Using-machine-learning-to-detect-malicious-URLs.git, del cual se adquirieron 345.000 URL.

2. *Verificar el estado de cada página web del dataset de URL.* Para ello se desarrolló una herramienta en Python y con la librería urllib2 se verifico si cada URL se encontraba activa o inactiva. Como resultado se obtuvo un total de 35.279 enlaces maliciosos activos (el 19 %) y se seleccionó una muestra aleatoria de URL benignas a las cuales se les aplico el proceso, y como resultado se obtuvo una lista de tamaño de 27.912.

3. *Selección de características y su generador.* El trabajo parte de las características estudiadas por [Xu13], con la diferencia de que se analizarán los datos generados a partir de un conjunto de URL activas y a las capacidades de un honeypot tipo cliente de baja interacción (Thug).

4. *Recolección del tráfico web.* Cada URL fue ejecutada sobre Thug durante 4 segundos con su configuración por defecto y en paralelo al proceso nuestro script capturó el tráfico web por medio PyShark. Finalmente, del proceso se obtuvieron un total de 756 registros de tráfico de red malicioso y 1.743 del benigno.

5. *Procesamiento de la información.* Se desarrolló otra herramienta en Python para el procesamiento del contenido HTTP y las propiedades de Whois. Con lo anterior se obtuvieron las siguientes características:
Capa de aplicación

   - (A1): Content_length representa el tamaño total de caracteres en la URL
   - (A2): Number_special_characters es el total de caracteres especiales que aparecen sobre la URL (por ejemplo, ?, %, #, &, _,)
   - (A3): HTTPHeader_content_length representa al tamaño del contenido de la cabecera HTTP
   - (A4): HTTPHeader_server provee información acerca del servidor de la página web, entre la que se encuentra su nombre, tipo y versión
   - (A5): HTTPHeader_charset indica la codificación de cada página web (por ejemplo, ANSI, ISO-8859-1, UTF8)
   - (A6): Whois_regDate indica la fecha en que el servidor del sitio web fue registrado
   - (A7): Whois_updated_date es la última fecha en que el servidor fue actualizado
   - (A8): Whois_country indica el país donde se encuentra el servidor del sitio web
   - (A9): Whois_statePro representa a la localización donde fue registrado el sitio web
   - (A10): Whois_Domain indica el dominio del sitio web

   Capa de red

   - (R1): TCP_conversation_exhange cuenta la cantidad de paquetes que hay entre el honeypot y el sitio web por el protocolo TCP
   - (R2): Dist_remote_tcp_port es el número total de puertos distintos a los expuestos en TCP
   - (R3): Remote_ips representa al número direcciones IP conectadas al honeypot
   - (R4): Pkt_without_dns es un arreglo de todos los paquetes que no son DNS

- (R5): TCP_urg_packets representa al número de paquetes TCP con la bandera URG
- (R6): Source_app_packets es el número de paquetes enviados por el honeypot hacia el servidor remoto
- (R7): Remote_app_packets es la variable del volumen en bytes de la comunicación entre el servidor web al honeypot
- (R8): Duration es el tiempo de duración de la página web
- (R9): Avg_local_pkt_rate es el promedio de paquetes locales IP por segundo (paquetes enviados sobre la duración) enviados desde el crawler hacia el servidor web
- (R10): Avg_remote_pkt_rate es el promedio de paquetes remotos IP por segundo envidados desde el servidor remoto hacia el crawler
- (R11): App_packets es el número total de paquetes IP generados en la consulta de la URL, en la cual se incluyen las de DNS
- (R12): DNS_query_times lista de capas DNS queries

Una vez obtenidas las características mencionadas, se aplicó una regla de normalización en los datos numéricos con el fin de representarlos en un rango entre 0 y 1. Adicionalmente, los datos categóricos fueron simbolizados como binarios.

6. *Algoritmos de machine learning y su evaluación.* La tecnología utilizada para el análisis fue R, de esta se seleccionaron los clasificadores de machine learning: J48, Regresión logística (RL), Naive Bayes (NB) y Support Vector Machines (SVM). Por otra parte, cada algoritmo fue utilizado con su configuración por defecto y fue evaluado con los resultados en exactitud, el tiempo de entrenamiento y el valor Cohen's kappa.

4. Experimento y resultados

Análisis de los datos

De los datos obtenidos de la capa de aplicación podemos deducir lo siguiente: el tamaño promedio de las URL (A1) es más mayor en las maliciosas (benignas 53,31 y malignas 85,45), que el número de caracteres especiales (A2) es mayor en las páginas maliciosas (benignas 10,81 y maliciosas 17,20), se presentan mayores indices de servidores maliciosos en Apache y NGINX (A4). Por otra parte, gran parte de los datos de localización con Whois para páginas maliciosas se encuentran en US y CN. Finalmente, hay una mayor proporción de registros en A3 en las páginas maliciosas, pero existen herramientas que permiten reducir el número de caracteres, por lo tanto, el resultado podría ser muy variable y alterado.

Del tráfico web malicioso y benigno de las páginas web procesadas a partir de un honeypot de baja interacción se pueden inferir los siguientes puntos: Para R1 y R2 la cantidad de paquetes TCP es más frecuente en la comunicación entre las páginas benignas y el honeypot (32,79 benignos y 22,47 maliciosos). Por otra parte, los datos indican que hubieron mayor cantidad de conexiones IP al honeypot cliente de baja interacción (R3) cuando se ejecutaron páginas benignas (10,63 benignos y 2,47 maliciosos), pero la cantidad de consultas DNS a servidores remotos (R12) es mayor en las benignas y es probable que el resultado sea influenciado por las capacidades del honeypot (37,99 benignas y 27,66 malignas). Al parecer las páginas maliciosas tienden a tener un menor tiempo de duración en la comunicación (R8) (3,6 segundos en benignos y 3 segundos en maliciosos), además, durante este intervalo de tiempo la cantidad de paquetes transmitidos por segundo es mucho más elevada desde el cliente al servidor (R9) (0,8 benignos y 1,9 maliciosos) en contraste con la cantidad de paquetes recibidos desde el atacante (R10) (44,6 benignos y 14,5 maliciosos), pero, tanto en el total paquetes enviados (R5 y R6) (37,9 benignos y 27,6 maliciosos) y en su tamaño en bytes (R7) las páginas maliciosas tienen menores resultados comparados a los benignos

Algoritmos de machine learning

El proceso de entrenamiento y evaluación se dividió en tres partes: primero se evaluaron las capas de red y de aplicación por separado, en segundo lugar se estudiaron los clasificadores con todas las características, y finalmente se realizó un testeo sobre las características más representativas encontradas en previos estudios (A1, A4 y R8) [Xu13]. Adicionalmente, el dataset utilizado contó con un tamaño de 967 registros y 400 variables (861 observaciones benignas y 106 malignas), debido a que el conjunto de datos no se encontraba balanceado, se aplicó una validación cruzada con un k igual a 10. Ahora bien, las características evaluadas por parte de la capa de aplicación son A1, A2, A3, A4, A5, A8 y A9. Mientras que las evaluadas por parte de la capa de red son desde R1 a R12.

Tabla 1: Algoritmos, capa de aplicación y red

Alg	**Aplicación**		**Red**	
	Exactitud	**Seg**	**Exactitud**	**Seg**
SVM	89,09 %	2	55,16 %	1,9
RL	88,43 %	3,5	54,11 %	0,8
NB	84,7 %	3,3	55,16 %	0,8
J48	90,10 %	4	57,01 %	4

Tabla 2: Algoritmos y toda la matriz de datos

Algoritmo	Exactitud	Tiempo (s)
SVM	97,41 %	3,38
Regresión logística	90,58 %	5,31
Naive Bayes	10,96 %	2,28
J48	98,76 %	53,37

Realizando una predicción de la clasificación por cada capa (Tabla 1), se puede concluir que las características de aplicación tienen mayor influencia en los resultados a diferencia de la capa de red, dos ejemplos son el algoritmo J48 que presenta una exactitud del 90,1 % y una respuesta de 4,05 segundos a comparación del algoritmo de regresión logística que tiene un resultado del 88,43 % y un factor de respuesta de 0,87 segundos. Por otra parte, al realizar la evaluación con todas las características (Tabla 2) gran parte del desempeño de los clasificadores incrementó y así mismo su tiempo de respuesta, entre los resultados podemos resaltar que el J48 aún conserva la mejor clasificación con un 98,76 %, pero con un tiempo de 53,43 segundos, por otra parte el algoritmo de SVM presenta un 97,71 % y con un tiempo mucho menor (3,38 segundos).

Tabla 3: Algoritmos y características A1, A4 y R8

Algoritmo	Exactitud	Tiempo (s)
SVM	85,46 %	1,90
Regresión logística	84,51 %	0,06
Naive Bayes	85,46 %	0,02
J48	96,05 %	0,01

De la Tabla 3 podemos deducir que el algoritmo J48 conserva la mejor exactitud en la clasificación (96,05 %), con un tiempo de 0,01 segundos y con un indicador de kappa del 0,91, este modelo cuenta con una buena clasificación para un contexto donde el tiempo fuera significativo.

5. Conclusiones y trabajos a futuro

Hemos encontrado que gran parte de los honeypots hoy en día presentan una deficiencia en la documentación y también en sus actualizaciones, por lo tanto, se propone a trabajo a futuro realizar un estudio que proponga la evaluación de estas herramientas aplicadas con la metodología utilizada en este artículo. Por otra parte, concluimos que a través de un honeypot de baja interacción y un conjunto de datos reciente, es posible identificar una página web maliciosa con un resultado de exactitud del algoritmo J48 del 98,76 % con todas las características y un 96,05 % para solo tres variables, adicionalmente presentamos los resultados de distintas combinaciones de características debido a que un ataque cibernético puede tener muchas variables y presentarse en distintos contextos.

Referencias

[Urcuqui16] Urcuqui López, C. C., García Peña, M., Osorio Quintero, J. L., and Navarro Cadavid, A. *Antidefacement-State of art.* Sistemas & Telemática, 14(39): 9-27, 2016.

[Bartoli10] Bartoli, A., Davanzo, G., and Medvet, E. *A framework for large-scale detection of web site defacements.* ACM Transactions on Internet Technology (TOIT), 10(3), 10. 2010.

[Roesch99] Roesch, M. *Snort: Lightweight Intrusion Detection for Networks.* In LISA, (Vol. 99, No. 1, pp. 229-238). November 1999.

[Kim94] Kim, G. H., & Spafford, E. H. *The design and implementation of tripwire: A file system integrity checker.* In Proceedings of the 2nd ACM Conference on Computer and Communications Security, (pp. 18-29). ACM. November 1994.

[Aman14] Aman, H., Yamashita, A., Sasaki, T., and Kawahara, M. *Multistage Growth Model for Code Change Events in Open Source Software Development: An Example Using Development of Nagios.* In Software Engineering and Advanced Applications (SEAA), 2014 40th EUROMICRO Conference on (pp. 207-212). IEEE. August 2014.

[Mohaisen15] Mohaisen, A. *Towards automatic and lightweight detection and classification of malicious web contents.* In Hot Topics in Web Systems and Technologies (HotWeb), Third IEEE Workshop on (pp. 67-72). IEEE. November 2015.

[Xu13] Xu, L., Zhan, Z., Xu, S., and Ye, K. *Cross-layer detection of malicious websites.* In Proceedings of the third ACM conference on Data and application security and privacy , (pp. 141-152). ACM. February 2013.

[Atienza15] Atienza, D., Herrero, Á., and Corchado, E. *Neural analysis of http traffic for web attack detection.* In International Joint Conference, (pp. 201-212). Springer, Cham. 2015.