Cadenas de Markov para el análisis y detección de anomalías en valores de User-Agent en peticiones web

Mitsiu Alejandro Carreño Sarabia
E23S-18014

Resumen

Se propone un método basado en modelos no supervisados de aprendizaje máquina para evaluar y detectar valores anómalos en el campo "User-Agent" de las peticiones que recibe un servidor web. Mediante este método es posible evaluar el campo "User-Agent" de nuevas peticiones basado en el tráfico histórico del servidor y obtener un índice de similitud respecto a solicitudes pasadas, con ello es posible detectar anomalías o contenido malicioso y tomar acciones correctivas. Este trabajo es fácil de extender e implementar en otros campos también llenados por el usuario o cliente y que por ende también pueden contener información maliciosa.

1. Importancia

Con la expansión del acceso a servicios de internet, así como la creciente disponibilidad de dispositivos de distintas categorías para conectarse a la red, la demanda y tráfico de servicios web se encuentra en constante aumento. Mucho se ha desarrollado en términos de escalabilidad de infraestructura así como adopción de soluciones distribuidas para dar servicio a la creciente demanda.

Pero un aspecto muchas veces ignorado es la importancia de que las organizaciones evalúen cómo es realmente la interacción entre sus clientes y la infraestructura disponible, analizarlo puede ser útil para responder muchas preguntas cómo ¿se está obteniendo el máximo rendimiento de la infraestructura, o es necesario escalar? ¿Acceden desde un dispositivo móvil, una computadora, una pantalla inteligente? E incluso si el contenido que tiene el cliente es sospechoso, anómalo o malicioso. Es por ello que este trabajo propone una metodología para procesar la enorme cantidad de conexiones que recibe un servidor, de manera automática, confiable y partiendo del tráfico habitual del servidor.

De manera específica, el campo "User-Agent" es un campo dentro del estándar HTTP que tiene la intención de informar al servidor el tipo de software responsable de la solicitud HTTP, es decir, explica al servidor si la solicitud se está realizando a través de un navegador web, un web-crawler, así como la arquitectura y sistema operativo. La finalidad de esta información es ajustar la respuesta al contexto de donde se originó la solicitud (ej. contextos de accesibilidad, lectores braille, adaptar el diseño a distintas resoluciones de pantalla o definir el idioma preferido). Sin embargo, al ser un campo que el servidor lee y analiza, lo vuelve un vector de ataque, en el que el cliente puede aprovechar vulnerabilidades y ejecutar acciones que el servidor no estaba destinado a realizar.

2. Proceso

2.1 Recopilación de archivos log

Primero fue necesario conseguir una fuente de datos extensa y real, para ello se solicitó acceso a un servidor que aloja varios sitios web en ambiente de producción, es decir, los servidores a los que acceden usuarios reales y que contienen información real.

En este caso, como la investigación está centrada en el campo "User-Agent" se accedió a los registros log del servidor HTTP, en estos logs se registran todas las conexiones que tuvo un servidor, la hora en que se realizó la conexión, el usuario, su User-Agent y su dirección IP, el recurso solicitado, el estatus de resultado de dicha solicitud, así como la cantidad de bytes intercambiados. El servidor HTTP estaba configurado para comprimir y crear un nuevo archivo cuando se alcanzara cierto espacio en disco, por lo que fue necesario copiar múltiples archivos.

2.2 Convertir en formato estructurado

Todos los campos mencionados se encuentran en formato texto plano con distintos separadores (espacios, guiones, y delimitadores como comillas y corchetes) por lo que fue necesario implementar una expresión regular que permitiera detectar y clasificar cada campo y almacenarlo en un DataFrame de Pandas. Además campos como date_time en formato [dd/mm/yyyy:hh:mm:ss timezone] además de almacenar el original se expandieron para poder realizar búsquedas por fecha y por hora, un tratamiento se realizó a campos request y req_uri los cuales estaban en formato codificado para URL, el cuál reemplaza signos y caracteres por valores porcentuales, esto ayuda al correcto procesamiento de urls en servidores pero en este caso, no son necesarios y sólo dificultan la lectura de los valores reales y dificultan posibles análisis de esos campos.

2.3 Análisis de información

El campo User-Agent sigue las especificaciones de la Semántica HTTP (https://www.rfc-editor.org/rfc/rfc9110.html#name-user-agent), donde lo reconocen como una serie de tokens (palabras claves) los cuales pueden o no tener uno o varios comentarios, por convención los tokens más relevantes se listan primero. Además cada token puede especificar una versión, un ejemplo de User-Agent sería: "Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0" para un navegador Firefox en un entorno de Windows 10 otro valor válido es "Mozilla/5.0 (iPad; U; CPU OS 3_2_1 like Mac OS X; en-us) AppleWebKit/531.21.10 (KHTML, like Gecko) Mobile/7B405" perteneciente a un Safari en Ipad o "Googlebot/2.1 (+http://www.google.com/bot.html)" para un web-crawler de Google.

Como la descripción del campo User-Agent lo explica, los valores son una serie de tokens, los cuales, no son completamente lenguaje humano, pero tampoco tienen la rigidez de un lenguaje únicamente para máquina. Por ello se decidió darles un tratamiento de procesamiento de lenguaje natural, en específico de generación de lenguaje natural y en esta caso se está tratando con datos no etiquetados, es decir, no tenemos una clasificación previa si un determinado User-Agent es normal, anormal o malicioso. Es importante resaltar que dada la naturaleza de los datos, es muy difícil llegar a contar con datos etiquetados en este contexto, ya que, con cada despliegue de nuevas versiones de navegadores, sistemas operativos, y motores de navegadores, los valores de User-Agent están

constantemente cambiando pero siguen siendo válidos y no maliciosos.

Para procesar los distintos valores se empleó un proceso de tokenización basada en trigramas, es decir el valor de User-Agent se divide todos los posibles en espacios de tres caracteres contiguos. De esta manera descomponemos las palabras, pero a la vez, podemos analizar con qué frecuencia ocurren estos intervalos de tres letras a la vez. Por otro lado, podemos emplear las cadenas de Markov, muy relacionadas con la probabilidad condicionada y el teorema de Bayes nos permiten evaluar, dado cierto trigrama, cuál es el siguiente trigrama más probable. Empleando estas dos técnicas podemos construir un generador de lenguaje natural en el que basado en nuestros datos de entrenamiento, evaluemos nuevos valores de User-Agents y determinemos qué tan similar es al resto de los valores conocidos.

Para el desarrollo del sistema, se decidió omitir la biblioteca NLTK debido a que si bien el texto a procesar no es completamente lenguaje máquina, es aún más lejano al lenguaje empleado por humanos, en este contexto, caracteres como diagonales, guiones, puntos y números tienen un significado muy distinto a cuando se usan en el lenguaje humano, por ello, opte por usar el framework “Flare” desarrollado por Austin Taylor (https://github.com/austin-taylor/flare) debido a que este framework desde su concepción fue pensado para analizar textos del contexto de redes, y por ello interpreta de mejor manera los signos y caracteres mencionados.

3. Resultados

Una vez entrenado el modelo, el framework nos permite evaluar nuevos valores de User-Agent y estimar qué tan similares son a valores con los que fue entrenado el modelo, esto sienta un precedente para evaluar qué tan anómalo es un nuevo valor y se puede establecer un límite mínimo de similitud antes de marcar un User-Agent como sospechoso.

4. Conclusiones

El análisis de registros log es si bien una tarea tediosa y con grandes volúmenes de datos, es necesaria para evaluar el correcto funcionamiento y uso de los sistemas.

Este modelo fue entrenado con los datos de poco más de un millón de conexiones distintas, de las cuales se encontraron tres mil valores distintos de User-Agent, lo cuál solo representa la información de dieciséis días. Lo cual pone en perspectiva la velocidad en que crecen los datos y lo imposible de analizarlos de manera manual, es necesario procesarlo con ayuda de aprendizaje de máquinas que puedan lidiar con el volumen de datos. A la vez, tiene un impacto importante desde el punto de vista de negocio, ya que asegura que el desarrollo cumpla las necesidades de los clientes (resolución de pantalla, idioma, accesibilidad), da un mejor entendimiento del rendimiento de la infraestructura, y ayuda a detectar y corregir ataques web, los cuales siempre suponen un riesgo.

Finalmente, este sistema puede ser una base para análisis más complejos y extensos que ayuden a detectar y prevenir ataques.