Utilización de la plataforma Hadoop para la implementación de un programa que permita determinar mensajes spam

Gustavo Crespo P. [1] Susana Véliz M. [2] Vanessa Cedeño M. Msc.[3]
Facultad de Ingeniería en Electricidad y Computación [1] [2] [3]
Escuela Superior Politécnica del Litoral (ESPOL)
Campus Gustavo Galindo, Km 30.5 vía Perimetral
Apartado 09-01-5863. Guayaquil-Ecuador
gcrespo@fiec.espol.edu.ec [1] sveliz@fiec.espol.edu.ec[2] vcedeno@espol.edu.ec [3]

Resumen

Se denomina spam a los mensajes no solicitados, de remitente no conocido que perjudican de alguna o varias maneras al receptor. Habitualmente son de tipo publicitario y enviados en cantidades masivas. Este proyecto propone un modelo de análisis de los archivos que sigan el formato de un correo electrónico, según la norma RFC822 con el fin de determinar potenciales mensajes spam. Para el filtrado de los mismos hemos utilizado la plataforma Apache Hadoop junto con la plataforma para analizar grandes cantidades de datos en un lenguaje de alto nivel Apache Pig. Para este estudio, obtuvimos una cantidad predeterminada de correos ham y spam. Se realizó un análisis utilizando la metodología de filtros bayesianos aplicados a los mensajes electrónicos introducida por el ensayista, programador, diseñador de lenguajes y co-fundador de viaweb Paul Graham, en modo stand alone así como en multinodo para ver las diferencias de tiempos de ejecución con uno y varios computadores. El conocimiento previo de la cantidad exacta de mensajes spam nos permitió determinar el grado de exactitud de nuestro filtro.

Palabras Claves: *Mensajes electrónicos, Correo, Filtro bayesiano, RFC822, Mensajes Spam, Mensajes Ham.*

Abstract

It's called spam to unsolicited messages from unknown senders that harm in one or more ways the receiver. Usually they are of advertising type and sent in bulk. This project proposes a model for analyzing electronic messages or emails that follows the format RFC822 standard to determinate potential spam messages. For the filtering we used the platform Apache Hadoop binding with High level language Apache Pig. For this study, we got a predetermined amount of ham and spam emails. The analysis was conducted using the methodology of Bayesian Filters applied to electronic messages by the essayist, programmer, designer and co-founder of viaweb Paul Graham, in standalone mode and in multinode to see the differences in execution times with one or several computers. Foreknowledge about exact amount of spam messages allowed us to determine the degree of accuracy of our filter.

Keywords*: Electronic Messages, mail, Bayesian Filer, RFC822, Spam Messages, Ham Messages.*

1. Introducción

En el transcurso de los años de la última década las cuentas de correo electrónico han sido afectadas por Spam de correo no solicitado, anónimo y masivo, los cuales han sido combatidos con diferentes filtros de spam. Dichos spam usan direcciones de correo falsas o que pertenecen a otros. Los spammers ganan dinero con el pequeño porcentaje que les responde, por eso la mayoría de las veces envían correos a grandes escales.

Las técnicas de los spammers evolucionan cada vez que un filtro de correo electrónico es mejorado, ya que cada vez que una compañía de seguridad informática desarrolla un filtro con mayor efectividad en la detección de los mismos, los spammers reinvierten sus ganancias para implementar nuevas técnicas que les permitan evadir estos filtros, lo cual lo convierte un ciclo interminable de tácticas entre los spammers y las empresas.

La detección de posibles correos spam es un problema para el cual diseñamos un filtro bayesiano basado en probabilidades e Inteligencia Artificial, en el cual se realiza un entrenamiento previo a el filtro para que pueda aprender y detectar a lo que nosotros denominamos spam. Y luego con estos datos y probabilidades realizar un reconocimiento de cada mensaje electrónico y etiquetar con una probabilidad a cada uno de ellos.

Este proyecto propone un modelo de análisis de los archivos que sigan el formato de un correo electrónico, según la norma RFC822. Tenemos la firme convicción de que los correos spam pueden ser detectados utilizando filtros.

Durante décadas los spammers han eludido los antispam con el fin de entregar su mensaje, si diseñamos un software inteligente que sea capaz de reconocer estos mensajes podremos detenerlos.

2. Fundamentos Teóricos

Se denomina spam a los mensajes no solicitados, de remitente no conocido que perjudican de alguna o varias maneras al receptor. Habitualmente son de tipo publicitario y enviados en cantidades masivas.

El spam se fundamenta en: robo de servicios, fraude y engaño, mediante la transferencia del costo de quien lo envía (el spammer) a quien lo recibe (la víctima). Aunque la mayor parte de los productos y servicios que se ofrecen no fuesen de dudosa legalidad, un negocio que toma algo de sus potenciales Clientes sin su autorización previa, que se aprovecha de los incautos e inocentes, y abusan de los recursos del Internet estuvo, está y estará, condenado al más rotundo de los fracasos [2].

2.1. Apache Hadoop

El clúster incluye un "single master", y múltiples "worker nodes". El nodo maestro corre los demonios de hadoop JobTracker, y Namenode.

Un esclavo o nodo trabajador actúa como un nodo de datos y un rastreador de tareas, en el cual correrán los demonios Datanode y Tasktracker.

El nodo secundario, representa el nodo backup del nodo maestro, debido a que en este se almacena la información del nodo maestro, si es que este llega a fallar. Sobre este nodo corre el demonio Secondary NameNode.

En grandes clúster, el HDFS es administrado a través de un servidor NameNode dedicado para alojar los índices de sistemas de archivos y un secundario NameNode que puede generar "snapshots" de las estructuras de memoria del NameNode, para prevenir corrupción en los archivos de sistema y reducir perdida de datos. Similarmente un servidor "JobTracker" independiente puede administrar la planificación de tareas. [3]

2.2. Apache Pig

Apache Pig provee un motor para la ejecución de flujo de datos en paralelo sobre Hadoop. Esto incluye un lenguaje de alto nivel, denominado Pig Latin, para expresar este flujo de datos.

Pig Latin incluye operadores para muchas de las operaciones tradicionales de datos, como lo son join, sort, filter, etc. Así como la capacidad para que los usuarios desarrollen sus propias funciones para leer, procesar y escribir datos. [14]

3. Objetivos

Diseñar un modelo de análisis de los archivos que sigan el formato de un correo electrónico, según la norma RFC822.

Realizar un filtro utilizando Apache Hadoop como plataforma principal utilizando sus características de computación, almacenamiento paralelo, junto con Apache Pig el cual nos proporciona un lenguaje de alto nivel llamado Pig Latin con el cual es realizado el filtro bayesiano propuesto, utilizando métodos de probabilidades para determinar luego del entrenamiento, la probabilidad de que un mensaje sea un spam.

Analizar la eficacia del filtro, basándonos en la cantidad de mensajes spam y mensajes no spam que detecto contrastándolos con la cantidad real de mensajes spam y no spam.

4. Problemática

Para los destinatarios el spam es de fácil reconocimiento, por lo general al leer su correo desechan aquellos no deseados.

Si utilizamos un poco de Inteligencia Artificial para automatizar este proceso, podremos filtrar los correos a través de sencillos algoritmos a través de combinaciones bayesianos de las probabilidades de palabras spam individuales.

5. Solución

Para la detección de mensajes spam, diseñamos un filtro bayesiano, basado en las características del filtro propuesto por Paul Graham en el ensayo "A Plan for Spam".

El cual básicamente es creado utilizando un corpus de mensajes spam y mensajes no spam, en el cual se analiza la probabilidad de que una palabra contenida en un mensaje, represente que este mensaje sea o no un spam, con una cierta probabilidad.

Filtro bayesiano implementado sobre el lenguaje de alto nivel Apache Pig, que opera sobre Apache Hadoop para el almacenamiento y computación paralela, así como Apache Tika, como parser principal para los datos de entrada, que siguen un formato RFC822.

6. Análisis de la Solución

Reconocimiento de mensajes Spam: Un mensaje será reconocido para nuestro filtro como spam si la probabilidad de spam obtenida por el filtro, según las palabras que contenga devuelva como resultado un índice mayor o igual al 90%.

La solución propuesta para la detección de mensajes spam, se basa en los siguientes componentes:

6.1. Parser TIKA

Es el primer componente por el cual la data atraviesa dentro del programa Antispam, aquí ingresa el mensaje según el estándar RFC822 y el parser de Apache Tika extrae la meta data, encabezados y

contenidos del mensaje para luego ser procesados por el filtro. [15]

6.2. Filtro (word, probabilidad-spam)

Este es el filtro bayesiano, previamente entrenado utilizando un corpus de mensajes de spam y otro corpus de mensajes no spam, con el cual se determina la probabilidad individual de que esta palabra en un mensaje sea un spam.

6.3. Formula de probabilidad

Esta es la fórmula de probabilidad conjunta en la cual se suman las probabilidades individuales de las palabras contenidas en el mensaje, y se determina su porcentaje de probabilidad de que dicho mensaje sea un spam.

7. Herramientas a utilizarse

Requerimientos de Software: Sistema operativo de cualquier distribución de Linux, Apache Hadoop, Apache Pig, Apache Tika, Pig Latin, Java 1.6 o superior, SSH.

Java para que funcione Hadoop, obligatoriamente se debe definir la variable de entorno JAVA_HOME.

Apache Hadoop para almacenar los datos y levantar los demonios.

Apache Pig. Utilizando su lenguaje de alto nivel Pig Latin para crear Jobs de Hadoop, sin preocuparnos en el desarrollo de bajo nivel de clases Mappers o Reducers.

8. Pruebas y Resultados

Las pruebas se realizaron en dos ambientes: stand alone y multinodo. En cada ambiente se procesaron la misma cantidad de correos. Para la implementación se diseñó el script tesis.pig**.** Esta script tiene como objetivo determinar la probabilidad de spam de cada correo a evaluar.

En stand alone se realizó una prueba con los 52790 spam y 32990.

Por los datos obtenidos en las pruebas, donde lo que cambia son la cantidad de maquinas esclavo, se puede verificar que el porcentaje de eficacia del filtro para detectar correos spam es del 88.99%, esto de acuerdo a el análisis de aplicar el filtro únicamente a los mensajes spam, para correlacionar las pruebas totales contra los datos reales contenidos en los corpus, mientras que el porcentaje de falsos negativos asciende a 11.01% lo que representa que toda esta cantidad de mensajes electrónicos no fueron filtrados, por alguna razón y pasaron la verificación del filtro, así como vemos disminuido el porcentaje de falsos positivos a solo 3.51%, el cual representa un valor bastante aceptable, para ser un filtro entrenado por aproximadamente unos 5000 mensajes electrónicos.

En el siguiente gráfico se muestran los tiempos de duración de cada una de las pruebas realizadas con lo cual podemos visualizar que la prueba 3 tuvo un menor tiempo de duración debido a que se utilizo una mayor cantidad de nodos y el procesamiento paralelo tuvo éxito.

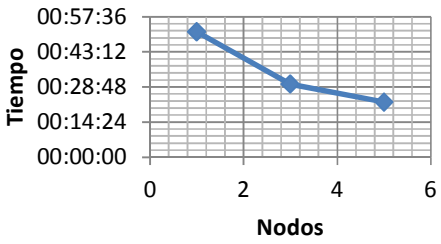

Figura 1. Comparación de la Duración contra la cantidad de nodos

Entonces por regresión lineal para calcular aproximadamente la curva del tiempo en función de la cantidad de nodos, tendríamos lo siguiente:

$$\gamma = \beta 0 + \beta 1 x$$

$$\beta 1 = \frac{n\sum xy - (\sum x)(\sum y)}{n\sum(x^2) - (\sum x)^2}$$

$$\beta 0 = \frac{\sum y - \beta 1(\sum x)}{n}$$

Si n = 3 | Numero de pruebas

Tabla 1. Pruebas con diferentes números nodos

x	y	xy	x^2
0	∞	∞	∞
1	0:51:23 | 51.38	51.38	1
3	0:29:56 | 29.94	89.82	9
5	0:25:42 | 25.70	128.50	25
$\sum x = 9$	$\sum y = 107.2$	$\sum xy = 269.82$	$\sum x^2 = 35$

$$\beta 1 = \frac{3(254.20) - (9)(103.9)}{3(35) - (81)} = -7.1875$$

$$\beta 0 = \frac{103.9 - (-7.1875)(9)}{3} = 56.1958$$

Entonces, la variable dependiente tiempo en función de la cantidad de nodos según la regresión lineal seria:

$$\gamma = 56.1958 - 7.1875x$$

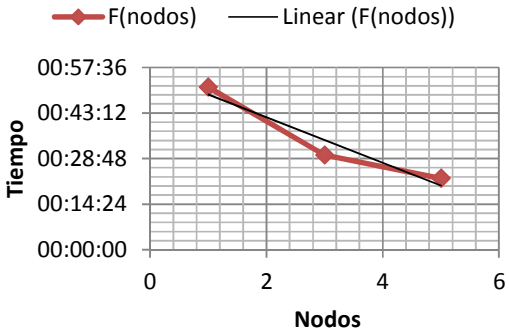

Figura 2. Representación de la ecuación de regresión lineal

9. Conclusiones

1. Se ha determinado que el uso de filtros bayesianos utilizando un corpus de entrada mínimo para entrenar al filtro con aproximadamente 4000 mensajes entre mensajes spam y no spam tiene resultados bastante aceptables que ascienden a un 88.99% en un corpus de aproximadamente 85780.

2. El grado de probabilidad para que un mensaje sea considerado mensaje spam, es que la probabilidad combinada basada en la fórmula del filtro sea mayor o igual al 90% lo que nos permite modificar estos resultados si así desea, dejándole un sesgo mayor o menor si así se desea.

3. Lo mejor a través del tiempo en esta clase de filtros bayesianos aplicados a la detección de spam es que no sería suficiente para los spammers hacer sus mensajes spam únicos o sencillamente dejar de utilizar palabras sucias que fácilmente sean detectadas, sino que aun tener que mantenerse cambiando el contenido del mensaje en cada uno de ellos y aun así si lo que desean es apuntar hacia un link, este si no es detectado a la primera vez, debido a la retroalimentación, seria detectado en sucesivas oportunidades para lo cual quedaría registrado este sitio, y ya sea que cada día tendrían que registrar nuevos sitios que no hayan sido registrados porque sencillamente de otra manera el filtro spam lo detectaría sin problemas.

4. La herramienta de Base que se ha utilizado fue Hadoop de Apache debido a que los requisitos, parámetros de entrada y condiciones así lo requirieron, basándonos en su arquitectura maestro – esclavo , y su sistema de archivos distribuidos, HDFS, los cuales representan una poderosa herramienta tanto para tratar gran cantidad de datos o data masiva como es nuestro caso, como la tolerancia a fallos tanto para Hardware como para Software, debido a que para el caso de uso real, es necesario tener la posibilidad de agregar más espacio para la nueva data, basta con tan solo agregar un nuevo nodo o terminal, para contar con el almacenamiento y capacidades de este nuevo nodo.

5. Además de contar con la replicación de la data lo cual no afectaría el workflow en tiempo real, si un terminal fallase por cualquier razón, el clúster y el JobTracker junto con el NameNode se basarán en la ubicación de los archivos de datos o bloques de datos replicados, para no afectar el rendimiento, ni resultados finales de la data, condición completamente importante para el análisis de resultados.

6. Debido a esto y las condiciones anteriormente expuestas, fue necesario para nosotros una herramienta que nos proporcione tolerancia a fallos, capacidad de almacenamiento masivo de datos, pero que a su vez, esto no disminuya su tiempo de respuesta, replicación de datos y un gran soporte detrás de esto, como lo proporciona Apache. Por esto y más, nuestra opción fue Apache Hadoop.

10. Recomendaciones

1. El filtro necesita un entrenamiento previo para la primera detección de spam de preferencia con una cantidad considerable de correos, luego de este entrenamiento será capaz de determinar por sí solo si un correo es legítimo o spam.

2. Mantener el grado de probabilidad en valores cercanos al 90% sin embargo se puede aumentar o disminuir el sesgo según se requiera.

3. Trabajar con varios nodos para prevención de fallos, mayor capacidad de almacenamiento y un tiempo de respuesta más optimo.

4. El clúster de computadoras o data center sobre el cual va a correr la plataforma Hadoop, debe ser analizada y revisada previamente por un especialista en redes, donde no se encuentre perdida de datos, ni ninguna clase de error en el envío de paquetes de una maquina a otra, debido que cada máquina ya sea configurado como maestro o esclavo, va a comunicarse directa o indirectamente con cada una de los otros host, y esto ocasiona un error fatal, a la hora de el traspaso de data entre hosts.

11. Referencias

[1] Kaspersky Lab, “Evolución del spam”, Disponible en línea en: http://www.viruslist.com/sp/spam/info?chapter=153350530, Fecha del último acceso: Octubre del 2011.

[2] Hermann Juergen, “Problema del Spam”, Disponible en línea en: http://www.cauce.org.ar/ProblemaDelSpam, Fecha del último acceso: Octubre del 2011.

[3] Chuck Lam, “Hadoop in Action”, Manning Publications, 2011, 325 Páginas.

[4] Apache Software Foundation, “Apache Hadoop”, Disponible en: http://hadoop.apache.org/ Fecha del último acceso: Noviembre del 2011.

[5] Apache Software Foundation, “HDFS Architecture”, Disponible en línea en: http://hadoop.apache.org/common/docs/r0.20.203.0/hdfs_design.html, Fecha del último acceso: Noviembre del 2011.

[6] Apache Software Foundation, “Single Node Setup”, Disponible en: http://hadoop.apache.org/common/docs/r0.20.203.0/single_node_setup.html, Fecha del último acceso: Noviembre del 2011.

[7] Apache Software Foundation, “Clúster Setup”, Disponible en: http://hadoop.apache.org/common/docs/r0.20.203.0/cluster_setup.html, Fecha del último acceso: Octubre del 2011.

[8] Apache Software Foundation, “MapReduce Tutorial”, Disponible en: http://hadoop.apache.org/common/docs/r0.20.203.0/mapred_tutorial.html, Fecha del último acceso: Noviembre del 2011.

[9] Ghemawat Sanjay – Gobioff Howard – Leung Shun Tak, “The Google File System”, Disponible en línea en: http://research.google.com/archive/gfs.html, Fecha del último acceso: Noviembre del 2011.

[10] Tom White, “Hadoop: The Definitive Guide 2nd Edition”, O’Reilly Media, 2010, 626 Páginas.

[11] G. Dalkilic & M. H. Ozcanhan, (2009). “A simple yet effective spam blocking method. SIN09 Proceedings of the 2nd International Conference on Security of Information and Networks (pp. 179-185)”. Disponible en: http://dx.doi.org/10.1145/1626195.1626241, Fecha del último acceso: Diciembre del 2011.

[12] P. He, X. Wen, & W. Zheng, (2009). “A simple Method for filtering Image Spam.2009 Eight IEEE ACIS International Conference on Computer and Information Science, 910-913 IEEE”. Disponible en: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5223148 Fecha del último acceso: Diciembre del 2011.

[13] J. Dean, & S. Ghemawat, (2008). “MapReduce: Simplified Data Processing on Large Clusters. Communications of the ACM, 51(1), 1-13, ACM”. Disponible en: http://portal.acm.org/citation.cfm?id=1327492. Fecha del último acceso: Diciembre del 2011.

[14] Apache Software Foundation, “Apache Pig”. Disponible en: http://pig.apache.org/ Fecha del último acceso: Mayo del 2012.

[15] Apache Software Foundation, “Apache Tika”. Disponible en: http://tika.apache.org/ Fecha del último acceso: Mayo del 2012.

[16] Internet FAQ Archives, “RFC 822 – Estándar para el formato de Texto ARPA DE INTERNET”. Disponible en: http://www.faqs.org/rfcs/rfc822.html#b Fecha del último acceso: Mayo del 2012.

[17] Paul Graham, “A Plan for Spam”. Disponible en: http://www.paulgraham.com/spam.html Fecha del último acceso: Mayo del 2012.

[18] Paul Graham, “Better Bayesian Filtering”. Disponible en: http://www.paulgraham.com/better.html Fecha del último acceso: Mayo del 2012.

[19] Paul Graham, “Hackers&Painters First Edition”, O’Reilly Media, 2004, 274 Páginas.