Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 83 (2016) 1213 – 1218

Procedia Computer Science

Second International Workshop on Mobile Cloud Computing Systems, Management, and Security (MCSMS-2016)

A Binary-Based MapReduce Analysis for Cloud Logs

Mouad Lemoudden*, Meryem Amar, Bouabid El Ouahidi

Mohammed-V University, Faculty of Sciences, L.R.I.
B.O. 1014, Rabat, Morocco

Abstract

Efficiently managing and analyzing cloud logs is a difficult and expensive task due the growth in size and variety of formats. In this paper, we propose a binary-based approach for frequency mining correlated attacks in log data. This approach is conceived to work using the MapReduce programming model. Initial experimental results are presented and they serve as the subject of a data mining algorithm to help us predict the likelihood of correlated attacks taking place.

Peer-review under responsibility of the Conference Program Chairs

Keywords: cloud; big data; logs; log management; binary approach; predict security attacks

1. Motivation

As cloud computing technology[1,2,3] adoption continues to progress, massive growth in the scale of log data generated has been observed, giving rise to the ability to perform security and auditing tasks through log management[4,5]. The volume and variety of cloud logs is experiencing a massive rise in size that it becomes increasingly difficult for log management solutions to store log files, parse them, and trace potential issues. Such growth has driven us to use the Hadoop framework which allows for storing, processing and analysis[6].

As such, actual log management tools cannot competently manage cloud log data[7]. Therefore, a novel approach is needed. Logs are usually flat files, where each line is a record containing at least a timestamp, event identifiers and the actual free text log message containing the information of the executed event[8]. The rise in size of log data marks

* Corresponding author. Tel.: +212 6 68 29 13 53; fax: + 212 5 37 77 42 61.
E-mail address: mouad.lemoudden@gmail.com

Peer-review under responsibility of the Conference Program Chairs
doi:10.1016/j.procs.2016.04.253

one of the three important attributes that gave rise to the big data movement[9,10]. Velocity expresses the speed with which data is generated. Lastly, Variety indicates the diverse sources of data.

In this present work, we performed data extraction from log files to JSON format because of JSON's lightness and easiness to parse. We also propose a binary-based approach for uncovering correlated security attacks that are detected in log files. This approach is implemented on the Big Data framework Hadoop[11], using the MapReduce programming model[12]. Experimental results are presented. We finally interpret our results using a data mining algorithm to predict incoming attacks.

2. Web Application Attacks in Log Files

Cloud computing services in nature are web applications which render desirable computing services on demand[13], especially with the natural cloud computing adoption of Web 2.0[14]. On the other hand, there are domain-specific cloud platforms like Google AppEngine and SalesForce business software development platform that are being targeted exclusively as traditional web applications[15,16]. Thus, we consider web applications to be an exemplary use case of cloud computing that offers rich grounds on which we can implement our proposed approach. This allows us also to benefit from the strong existing research and resources dedicated to the study of Web application log files and their impact on security.

OWASP[18] (Open Web Application Security Project) lists the 10 most important web application security weaknesses, gathering their datasets from numerous Web application security actors including SaaS (Software as a Service) vendors. Here are some of the most prominent attacks that are recognizable in a web server log file using specific regular expressions: Cross Site Scripting (XSS), Injection Flaws, and Insecure Direct Object Reference.

It is essential to consider log data as very important and to exploit it using Big Data techniques. Making it possible for organizations to gracefully gain access to a wealth of insight residing in unstructured log files, which is presently being underutilized because of its vast variety and volume.

3. Extracting Data from Log Files to JSON Format

We begin our work by retrieving structured data out of unstructured log files, in the Common Log Format (CLF) specification[22], for further data processing and storage. Extracting data from log files has grown into a considerable technical task because it has to deal with a variety of formats. To achieve a proper data extraction, we elected to work with Python programming language due to its flexibility, efficiency and relative ease when handling parsing tasks. In our Python program, we used Pyparsing, a useful class library to construct grammar parsers directly in Python code[19].

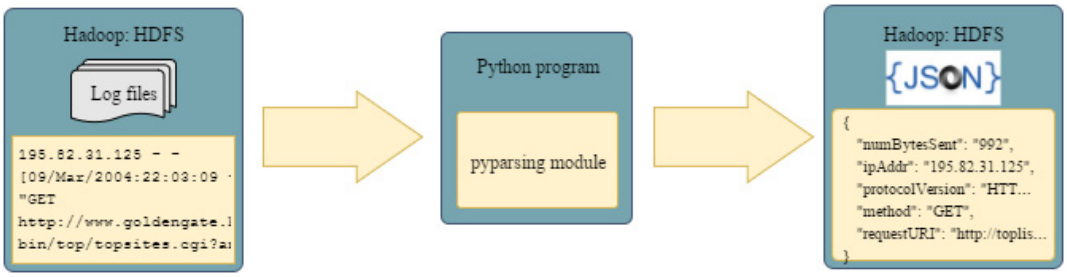

Fig (1). Data extraction from raw log files to JSON.

The result of this stage in our work is a JSON (JavaScript Object Notation) file containing variables that correspond to log file records, as shown in the above figure. JSON is designed to be a light data exchange language which is easy for computers to parse and use. Compared to other means of structured data exchange, like XML, JSON provides significant performance gains and is able to parse up to one hundred times faster[20].

4. Description of the Binary-Based Method

The binary-based method is a data mining technique for discovering associations between related attacks that are detected in the log file. The proposed technique is composed of two binary data structures as well as algorithms for frequency analysis[21]. The task of this technique can be decomposed as:

- Find all frequent set of attacks in the log file.
- Use a frequency code to generate the combination and association between the other related attacks.

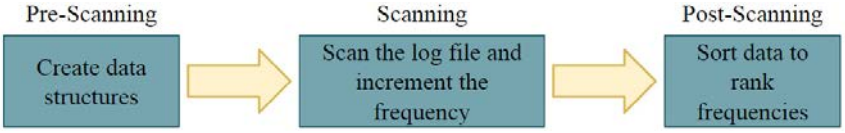

Fig (2). Main stages of binary-based frequency analysis technique.

The above Fig (2) shows the three main stages of the technique. To make the discussion clearer, the binary data structures and algorithms are presented in the next section using the MapReduce model.

5. The Binary-Based MapReduce Analysis

In order to conduct frequency analysis of the log file, we present our Big Data enabled binary-based technique. In the context of our implementation, we will operate with the JSON data that we generated as specified in section 3, which contains user requests, in order to detect web application attacks. However, the same technique can be used for other purposes. There are two data structures that will be created, the first is an array named nameTab which contains the names of the different attacks (described in section 2) detected from the log file. The other is an array named arrFreq which will be used to store the frequency of certain combinations of attacks detected. Fig (3) shows the three stages of the technique, which are the pre-scanning phase, the map phase and the reduce phase.

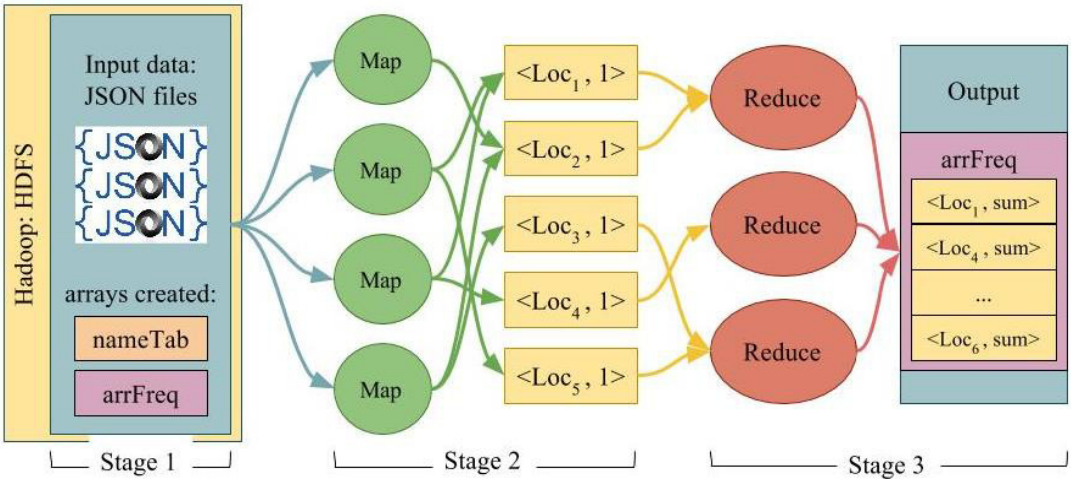

Fig (3). The main stages involved in our approach.

5.1. The Pre-Scanning Phase

In this stage, the data structures nameTab and arrFreq will be created. The size of arrFreq depends on the number of attacks that we can detect. For example, if the number of recognizable attacks n, is 5, then the size will be $2^n = 2^5 = 32$, corresponding to the possible combinations of the 5 attacks.

Considering that attacks A, B and C are stored in the array nameTab in locations 0, 1 and 2 respectively, If we find a log record containing attacks A and C, this combination will have the index 5 in the array arrFreq, which is determined by binary conversion. In this case A and C refers to 101 in binary, which is the binary conversion of 5. Then, the arrFreq index is determined as follows: $2^0 + 2^2 = 1 + 4 = 5$.

5.2. The Map Phase

In this stage, the algorithm starts by scanning each JSON variable which is stored in HDFS[24]. Attacks can be detected by comparing the content of a JSON variable against a series of particular regular expressions which are sequences of characters designed to recognize patterns of different attacks. For each attack detected in the record, its numeric identifier can be found accordingly in nameTab. The identifier will be used in the following formula to determine the corresponding arrFreq index, named 'Loc', where 'i' is the attack index in nameTab:

$$Loc = \sum_{i=0}^{n} 2^i \tag{1}$$

The following algorithm presents the process of the map phase, where 'i' is the index of the current attack in nameTab. The output of the map step is a key and value pair: the arrFreq index, and the frequency number to be used in the reduce step:

```
Begin
loc ← 0
For each i in nameTab
    If i is detected in log record
        loc ← loc + 2^i
    End if
End for
Output [loc, 1]
End
```

Algorithm 1. Scanning log file in the map phase.

5.3. The Reduce Phase

At this stage, Hadoop worker nodes will redistribute the data based on the output keys that were produced by the map function. The reduce method will, subsequently, perform an adding operation on each group of the output data, per key, in parallel. The arrFreq array will be populated based on the results of the reduce method, which can be sorted to rank the frequencies along with their array indexes from highest to lowest.

In the end, finding the combination of attacks that are most frequently attempted against the web server will require a conversion of the top indexes to their binary equivalent. For example, if the arrFreq index 10 is among the most frequent in the log file, we can figure out the combination of attacks by converting to binary: 1010. We can conclude that the combination of attacks represented by the index 10 is B and D.

6. Experimental Results

6.1. Implementation Environment

In our implementation, we undertook a number of stages to develop predictions based on attack pattern detection in log files. The implementation was conducted on Intel® core™ i5-3340 CPU, 2.70 GHz, and 4 GB of RAM computer. We devised our implementation on two parts. The first part consisted of extracting structured information from unstructured log data as a JSON file containing variables that correspond to log file records.

The second part of our implementation consisted of storing the JSON variables in HDFS and passing them as input data to the MapReduce model exactly as detailed earlier. The attacks that we were able to detect in the log data are: A-Cross Site Scripting (XSS), B-Injection Flaws, C-Insecure Direct Object Reference and D-Information Leakage and Improper Error Handling[18,25]. This part was executed on Hadoop 2.6, using Java programming language for its high performance and compatibility to the Hadoop framework. The dataset used in the experiment is from The Honeynet Project, challenge Scan 31[26].

6.2. A-priori Algorithm to find patterns

Now that the detection of several web attacks from log files has been completed, applying various strategies of data mining will be helpful to generate patterns and associations rules. The approach that we opted for is based on A-priori algorithm[17], which is used to identify association rules over a transactional database, considering a predefined minimum support, and minimum confidence, it locates correlations between items. Considering the rule A=>B, a support is an indicator of "reliability", and means that attacks A and B are present together in the transaction set. P(A∩B) is the probability of having attacks A and B at the same time.

$$Supp\{A,B\} = \mathrm{P}(\mathrm{A} \cap \mathrm{B}) \tag{2}$$

Confidence is an indicator of accuracy. This rule, A=>B, holds that if the attack A is detected, we have a probability that the attack B will happen.

$$Conf\{A => B\} = \frac{\mathrm{P}(\mathrm{A} \cap \mathrm{B})}{\mathrm{P}(\mathrm{A})} = \frac{Supp\{A,B\}}{Supp\{A\}} \tag{3}$$

6.3. Our Results

In this section, we applied support and confidence measures to locate association rules and to predict incoming attacks based on their probability to materialize. In the following table, a "transaction" refers to the index of a JSON variable, in which an attack or more were detected, and "Frequency" is the number of the JSON variables that verify the transaction. To illustrate, in transaction 3, the attacks A and B were detected in the same JSON variable.

Table 1. Set of attack combination frequency.

Transactions	D	C	B	A	F (Frequency)
2	0	0	1	0	77
3	0	0	1	1	1
4	0	1	0	0	129
6	0	1	1	0	21
8	1	0	0	0	953
9	1	0	0	1	1
10	1	0	1	0	1

In this case, the support (Supp{X}) is calculated as shown in equation 4, where n is the number of transactions, x_i takes the value of 1 if attack X exists in the current transaction and the value of 0 otherwise, F_i is the frequency of log records verifying the transaction i:

$$Supp\{X\} = \sum_{i=1}^{n} (x_i F_i) \tag{4}$$

We applied equation 4 to determine the support of each element of the attack combination. Then, we applied equation 2 to determine the support of groups with cardinality of two, and we stopped at this step because we don't have more associations. In our case, the support of B and C is equal to: Supp{B,C} = 21/1183.

After that, we calculated the confidence measure applying equation 3, and the results (Conf{B=>C} = 0,21) confirm that detecting the attack B (Injection Flaws) means that we have 21% of probability that the attack C (Insecure Direct Object Reference) will follow.

7. Conclusion

Analysis of machine-generated log data helps greatly in identifying several security issues. Although, the size of this data in a cloud environment can be overwhelming, it is important to consider the larger advantages of log management and to exploit MapReduce and A-priori algorithm to provide better analysis and to allow us make a prediction of several attacks at a percentage of probability.

In this paper, binary-based and A-priori methods perform frequency analysis to predict attacks in a log. By implementing these techniques, mining log data for different attacks can be done even if the amount of data is huge. In the end, we presented our experimental results.

References

1. Mell, Peter, Grance, "The NIST definition of cloud computing", 2011.
2. Sosinsky, Barrie, "Cloud computing bible", John Wiley and Sons, Vol.762, 2010.
3. Vaquero, Luis, et al, "A break in the clouds: towards a cloud definition", ACM SIGCOMM Computer Communication Review 39.1, Pages: 50-55, 2008.
4. Lemoudden, El Ouahidi, "Managing Cloud-generated Logs Using Big Data Technologies", In Proceedings of the 3rd Edition of the International Conference on Wireless Networks and Mobile Communications (WINCOM'15). IEEE, 2015
5. Ben Bouazza, Lemoudden, El Ouahidi, "Surveying the challenges and requirements for identity in the cloud", Proceedings of the 4th Edition of National Security Days (JNS4), IEEE, Pages: 1-5, 2014.
6. Hashem, Abaker Targio, et al, "The rise of 'big data' on cloud computing", Information Systems 47, Pages: 98-115, 2015.
7. Lemoudden, Ben Bouazza, El Ouahidi, "Towards achieving discernment and correlation in cloud logging", Applications of Information Systems in Engineering and Bioscience, Pages 202-207, 2014
8. Nagappan, Meiyappan, Robinson, "Creating operational profiles of software systems by transforming their log files to directed cyclic graphs", Proceedings of the 6th International Workshop on Traceability in Emerging Forms of Software Engineering. ACM, Pages: 54-57, 2011.
9. McAfee, Andrew, Brynjolfsson, "Big data: the management revolution." Harvard business review 90, Pages: 60-6, 2012
10. Zikopoulos, Paul, Harness, "the Power of Big Data the IBM Big Data Platform", McGraw Hill Professional, 2012.
11. White, Tom, "Hadoop: The definitive guide", O'Reilly Media Inc, 2012.
12. Dean, Jeffrey, Ghemawat, "MapReduce: simplified data processing on large clusters", Communications of the ACM 51.1, Pages: 107-113, 2008.
13. Wang, Lizhe, et al, "Cloud computing: a perspective study", New Generation Computing 28.2, Pages: 137-146, 2010.
14. Rewatkar, Liladhar, Lanjewar, "Implementation of Cloud Computing on Web Application", International Journal of Computer Applications 2.8, Pages: 28-32, 2010.
15. Armbrust, Michael, et al, "A view of cloud computing", Communications of the ACM 53.4, Pages: 50-58, 2010.
16. Fox, Armando, et al, "Above the clouds: A Berkeley view of cloud computing", Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS 28, Page: 13, 2009
17. Agrawal, Rakesh, Ramakrishnan, "Fast algorithms for mining association rules", In Proc. 20th int. conf. very large data bases, VLDB (Vol. 1215, Pages: 487-499, 1994.
18. OWASP, Top, "The Ten Most Critical Web Application Security Risks", 2013.
19. McGuire, Paul, "Getting started with pyparsing", O'Reilly Media Inc, Pages: 1-13, 2007.
20. Nurseitov, Nurzhan, et al, "Comparison of JSON and XML Data Interchange Formats: A Case Study", Caine 9, Pages: 157-162, 2009
21. Fageeri, Osman, Rohiza, "An Efficient Log File Analysis Algorithm Using Binary-based Data Structure", Procedia-Social and Behavioral Sciences 129, Pages: 518-526, 2014.
22. Luotonen, "The common log file format." CERN httpd user manual 176, 1995.
23. Monteith, Yates, McGregor, Ingram, "Hadoop and its Evolving Ecosystem", IWSECO@ ICSOB, Pages: 57-68, 2013.
24. Shvachko, Konstantin, et al, "The hadoop distributed file system", IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST), Pages: 1-10, IEEE, 2010.
25. Meyer, Roger, "Detecting attacks on web applications from log files", Sans Institute, InfoSec reading room, 2008.
26. "The Honeynet Project: Discover how an OpenProxy is abused", http://www.honeynet.org/scans/scan31/, 2004.