Detecting DDoS Attacks with Hadoop

Yeonhee Lee
Chungnam National University
Daejeon, 305-764, Republic of Korea
yhlee06@cnu.ac.kr

Youngseok Lee
Chungnam National University
Daejeon, 305-764, Republic of Korea
lee@cnu.ac.kr

ABSTRACT

Recent distributed denial-of-service (DDoS) attacks have demonstrated horrible destructive power by paralyzing web servers within short time. As the volume of Internet traffic rapidly grows up, the current DDoS detection technologies have met a new challenge that should efficiently deal with a huge amount of traffic within the affordable response time. In this work, we propose a novel DDoS detection method based on Hadoop that implements a HTTP GET flooding detection algorithm in MapReduce on the distributed computing platform.

1. INTRODUCTION

HTTP GET flooding is one of the typical DDoS attacks that exploit normal TCP connections between a client and a target web server. As the volume of Internet traffic increases explosively year after year, the Intrusion Detection Systems (IDSes) have faced the issue on how to assure both scalability and accuracy of analyzing the DDoS attack from these huge volume of data.

In recent years, several approaches have been proposed to solve this issue. Dimensionality reduction methods such as Principal Component Analysis (PCA) enables to classify large volume of traffic by separating the normal behavior from anomalies [2]. However, these schemes usually require too excessive computing cycles to apply to actual systems. Sketch-based studies focus on memory efficiency by utilizing hash tables. Though Liu et al. [3] proposed a two-level sketch approach to reduce memory consumption and searching complexity while boosting accuracy, their technique still needs sufficient memory space and complex computation.

Hadoop is an open-source distributed cluster platform that includes a distributed file system, HDFS and the programming model, MapReduce. In our previous research [4], we have developed a Hadoop-based packet processor (P^3) [4]. This paper extends P^3 to devise a DDoS anomaly detection method on Hadoop that implements a MapReduce-based detection algorithm against the HTTP GET flooding attack.

2. MAPREDUCE-BASED DETECTION ALGORITHM

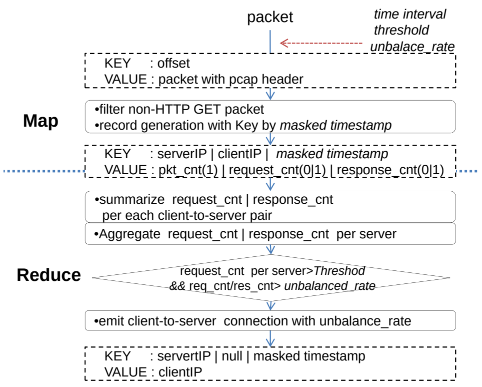
2.1 Counter-based method

Counter-based detection is a simple method that counts the total traffic volume or the number of web page requests [1]. Since the DDoS attack with the low volume of traffic such as the HTTP GET incomplete attack is prevalent in these days, the frequency of page requests from clients will be a more effective factor. Figure 1 illustrates our MapReduce algorithm to detect DDoS with URL counting. To lower the false positive rate, we adopted response rate against page requests as secondary regulation as well as traffic volume, which was proposed by Liu et al [3]. This algorithm needs three input parameters of *time interval*, *threshold* and *unbalance ratio*, which can be loaded through the configuration property or the distributed cache mechanism of MapReduce. *Time interval* limits monitoring duration of the page request. *Threshold* indicates the permitted frequency of the page request to the server against the previous normal status, which determines whether the server should be alarmed or not. The *unbalance ratio* variable denotes the anomaly ratio of response per page request between a specific client and a server. This value is used for picking out attackers from the clients.

In our MapReduce algorithm, the map function filters non-HTTP GET packets and generates key values of server IP address, masked timestamp, and client IP address. The masked timestamp with *time interval* is used for counting the number of requests from a specific client to the specific URL within the same time duration. The reduce function summarizes the number of URL requests, page requests, and server responses between a client and a server. Finally, the algorithm aggregates values per server. When total requests for a specific server exceeds the *threshold*, the MapReduce job emits records whose response ratio against requests is greater than *unbalance ratio*, marking them as attackers. While this algorithm has the low computational complexity and could be easily converted to the MapReduce implementation, it needs a prerequisite to know the *threshold* value from historical monitoring data in advance.

2.2 Access pattern-based method

The access pattern-based detection method assumes that clients infected by the same bot conduct the same behavior and that attackers could be differentiated from normal clients. This method requires more than two MapReduce jobs: the first job obtains access sequence to the web page between a client and a web server and calculates the spending time and the bytes count for each request of the URL; the second job hunts out infected hosts by comparing the access sequence and the spending time among clients trying to access the same server. The drawback of this approach is the highly computational complexity to spot the DDoS pattern.



packet
time interval
threshold
unbalace_rate
KEY : offset
VALUE : packet with pcap header
Map
•filter non-HTTP GET packet
•record generation with Key by masked timestamp
KEY : serverIP | clientIP | masked timestamp
VALUE : pkt_cnt(1) | request_cnt(0|1) | response_cnt(0|1)
•summarize request_cnt | response_cnt per each client-to-server pair
•Aggregate request_cnt | response_cnt per server
Reduce
request_cnt per server>Threshod && req_cnt/res_cnt> unbalanced_rate
•emit client-to-server connection with unbalance_rate
KEY : servertIP | null | masked timestamp
VALUE : clientIP

Figure 1: The MapReduce algorithm for counter-based DDoS detection

2.3 Performance Evaluation

For experiments of our algorithm in 1, we configured a small Hadoop testbed consisting of one master node and ten slave nodes. Each node is equipped with quad-core 2.93 GHz Intel i7 CPU, 16 GB memory, 1 TB hard disk and 1 Gbps Ethernet cards. To manifest the scalability of our algorithm, we measure the performance of the counter-based DDoS detection method by varying cluster nodes. Figure 2 shows that the detection job with ten worker nodes was completed within 25 minutes for 500GB and 47 minutes for 1TB, which is over 8 times faster than one node's and 2.9 times faster than 3 nodes' respectively. From evaluation results we could observe the increased performance enhancement as the volume of input traffic becomes large.

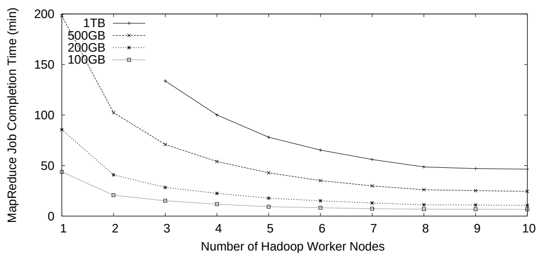


Figure 2: Completion time of a counter-based DDoS detection job regarding various # of Hadoop datanodes

3. CONCLUSIONS

In this paper, we focused on a scalability issue of the anomaly detection and introduced a Hadoop-based DDoS detection scheme to detect multiple attacks from a huge volume of traffic. Different from other single host-based approaches trying to enhance memory efficiency or to customize process complexity, our method leverages Hadoop to solve the scalability issue by parallel data processing. From experiments, we show that a simple counter-based DDoS attack detection method could be easily implemented in Hadoop and shows its performance gain of using multiple nodes in parallel. It is expected that a signature-based approach could be well suited with Hadoop. However, we need to tackle a problem to develop a real time defense system, because the current Hadoop is oriented to batch processing.

4. ACKNOWLEDGEMENT

This research was partly supported by the MKE (Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program supervised by the NIPA (National IT Industry Promotion Agency) (NIPA-2011-(C1090-1031-0005)) and partly by the IT R&D program of MKE/KEIT [KI001878, "CASFI : High-Precision Measurement and Analysis Research"].

5. REFERENCES

[1] J. Mirkivic and P. Reiher, *A Taxonomy of DDoS Attack and DDoS Defense Mechanisms*, ACM SIGCOMM CCR, 2004

[2] H. Sun, Y Zhaung, and H. Chao, *A Principal Components Analysis-based Robust DDoS Defense System*, IEEE ICC, 2008

[3] H. Liu, Y Sun, and M. Kim, *Fine-Grained DDoS Detection Scheme Based on Bidirectional Count Sketch*, IEEE ICCCN, August 2011

[4] Y. Lee, W. Kang, and Y. Lee, *A Hadoop-based Packet Trace Processing Tool*, TMA, April 2011