Spark-Based Port and Net Scan Detection

Antonia Affinito
University of Napoli Federico II
Napoli, Italy
antonia.affinito@unina.it

Alessio Botta
University of Napoli Federico II
Napoli, Italy
alessio.botta@unina.it

Luigi Gallo
Cyber Security Lab TIM S.p.A. *
University of Napoli Federico II +
*Turin,+Napoli, Italy
luigi.gallo3@unina.it

Mauro Garofalo
University of Napoli Federico II
Napoli, Italy
mauro.garofalo@unina.it

Giorgio Ventre
University of Napoli Federico II
Napoli, Italy
giorgio.ventre@unina.it

ABSTRACT

Network security is more and more important today. Port and net scan are the typical preliminary steps an attacker makes to find victims in a certain network, and are currently the most spread network anomalies. In this work, we focus on traditional approaches for port and net scan detection, previously abandoned due to their limited speed, and we use Big Data Analytics to speed them up and cope with current high-speed networks. The paper describes our approach and presents an experimental analysis in terms of detection performance and execution time of a threshold-based algorithm on Apache Spark. We use real traffic traces from MAWI archive and MAWILab anomaly detectors to compare with our results. The analysis shows that i) the threshold-based algorithm is already able to achieve detection performance higher than MAWILab (in 95% of the considered cases with the best threshold value), currently considered the gold standard in the field; ii) the execution time can be as low as 25 seconds for a 24h traffic trace collected over a 10Gbps link, which makes it usable also in real time. Moreover, we bridge an important gap in literature providing the research community with a new labeled dataset, validated using MAWILab and extended with other anomalies not detected by it.

CCS CONCEPTS

• **Security and privacy** → **Intrusion/anomaly detection and malware mitigation**; **Network security**; • **Networks** → *Network measurement*;

KEYWORDS

Anomaly Detection, Port and Net Scan, Apache Spark

ACM Reference Format:
Antonia Affinito, Alessio Botta, Luigi Gallo, Mauro Garofalo, and Giorgio Ventre. 2020. Spark-Based Port and Net Scan Detection. In *The 35th ACM/SIGAPP Symposium on Applied Computing (SAC '20), March 30-April 3, 2020, Brno, Czech Republic.* , 8 pages. https://doi.org/10.1145/3341105.3373970

SAC '20, March 30-April 3, 2020, Brno, Czech Republic

ACM ISBN 978-1-4503-6866-7/20/03…$15.00
https://doi.org/10.1145/3341105.3373970

1 INTRODUCTION AND MOTIVATION

The pervasive use of the Internet has led to a significant increase in the amount of traffic that crosses the network every day. On the other hand, network security and the related, necessary step of traffic analysis, are becoming more and more important [2, 5, 11, 17]. However, the amount of data that has to be analyzed is higher and higher, especially in current high-speed networks. Two typical steps attackers perform to find vulnerable hosts on the network are port and net scan. Such anomalous events represent a good fraction of all the anomalies in current traffic [1]. Detecting them represents an important yet difficult task due to the high volume of traffic to analyze.

Network traffic can be analyzed at several layers of the protocol stack: packet, flow, application, etc. At flow-level, packets relating to the same TCP or UDP communication (e.g. all packets related to an HTTP communication from and to a single host and a web server) are aggregated, and a summary of such group of packets is considered. These summaries can now be provided directly by network devices such as switches and routers, and standard protocols have been defined for this aim (e.g. Internet Protocol Flow Information Export or IPFIX [12]). Working at flow level seems the most promising approach for coping with the high-speed of current and future networks. However, even at the flow-level, the analysis of traffic for the detection of anomalies in high-speed networks requires huge computational power or data reduction techniques as flow records still represent a huge quantity of data.

In this paper, we analyse traffic from high-speed networks using a flow-level approach. The aim is to detect two of the most spread network anomalies i.e. port scan and network scan (or simply net scan). In the former case, an attacker probes a host (the victim) on various TCP/UDP ports to find active and vulnerable services. In the latter case, the attacker scans a group of victim hosts on a single or a small number of ports. Such scanning activities are also associated with worms and botnets [7].

Port and net scans generate a real specific pattern in network traffic. One of the most popular methods for detecting scanning activity is based on the fan-in fan-out proportion of the hosts, i.e. counting the number of incoming and outgoing flows, and

[1] Mawilab Documentation: http://www.fukuda-lab.org/mawilab/documentation

comparing their ratio with a threshold [16, 23]. With this approach, the performance in terms of detection capacity can be high, but the performance in terms of execution times can be very low [16, 20, 23]. Sampling is typically applied to solve this problem, but this involves a significant loss of information. To overcome this problem, we have used Big Data analysis techniques to analyze the whole volume of traffic with a threshold-based algorithm in the shortest possible time. In particular, we use the Apache Spark framework (Spark in the following), which is comparable to Hadoop Map/Reduce but it provides faster results working entirely in the memory.

We concentrated on a simple threshold-based detection algorithm, implemented it in Spark, and performed a large experimental evaluation of its performance by using several real traces. Our results in terms of execution time show that we achieve a processing rate down to 25 seconds for a 24h traffic trace collected on a 10Gbps link, which makes the approach able to easily run in real time, also in much faster links. Comparing our detection performance with MAWILab [1], we show that our approach achieves fewer false negatives, which is to say that we can uncover more anomalies than the gold standard. For this reason, we also provide an improved dataset publicly on the web with labeled traffic traces, including more port and net scan events than the ones from MAWILab.

Our contributions can be summarized as follows: i) we report the results of a longitudinal analysis on MAWI archives that shows how port and net scans contribute to the total of network anomalies; ii) we show that the threshold-based approach can achieve higher detection performance than the available gold standard based on much more complex and therefore less observable approaches; iii) we show that threshold-based approaches can be applied to current network traffic using Big Data technologies, achieving very high processing speeds; iv) we provide an improved and constantly updated dataset which can be used by the research community for further studies on this important topic. The new dataset is updated daily on our project website http://spada.comics.unina.it.

We believe this represents an important step forward for the research community focused on port and net scans.

The remainder of the paper is organized as follows. Sec. 2 motivates our choice to focus on scanning activities. Sec. 3 positions our work in the context of previous research. Sec. 4 describes the threshold algorithm and the Apache Spark framework adopted by our system. Sec. 5 gives an overview of the dataset and the methodology we use. Sec. 6.1 presents the evaluation and related results obtained in different scenarios. Finally, Sec. 7 summarizes our conclusions and discusses future work.

2 SCANNING ACTIVITIES OVER TIME

Our first question in this work was if net and port scan anomalies are still actual today, so to justify further studies on this topic. To answer this question, we have conducted a longitudinal analysis of the number of anomalies detected by MAWILab over time. We collected and analyzed data regarding the anomalies detected by MAWILab in the last years and dissected them according to the type of anomaly. Fig. 1 shows the box plot of the ratio of scanning anomalies over the total number of anomalies, for the years from 2007 to 2017: the x-axis represents the year, and the y-axis represents the percentage of port and net scans over the total number of anomalies detected by MAWILab. The results are aggregated per year, starting from the information available day-by-day. The graph presents a growing trend, with a large increase in the ratio starting from 2014. This result indicates that the scanning anomalies are increasingly present in the traces, an increase that has been evident especially in recent years. This behavior motivates our choice to focus on these types of the anomalies and calls the research community for always updated data, techniques, and tools for port and net scan detection.

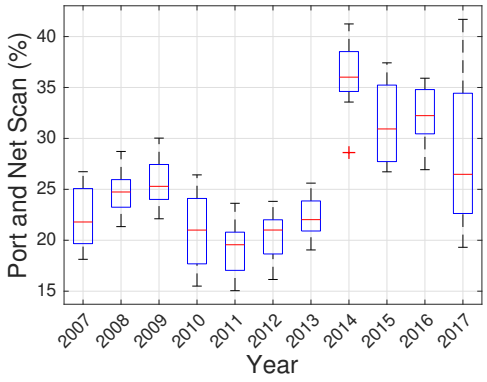


Figure 1: Ratio of scanning over all anomalous activities over time during the last 11 years.

3 STATE OF THE ART

Scientific literature has been focusing on network and port scan for several years. Generally, these anomalies are examined considering that a source of scanning activity shows a very high number of outgoing connections [20].

Zhao et al. [23] proposed an approach based on this consideration. They also proposed a standard hash-based flow sampling algorithm to cope with high connection speeds (10-40 Gb/s). *Dainotti et al.* [8] used wavelet to detect network anomalies and to precisely locate their position inside the traffic, while *Balram et al.* [3] proposed a technique based on packet count through neural networks.

Sridharan et al. [21] compared the performance of Snort and Bro on backbone traffic and proposed a new approach based on sequential hypothesis testing. *Kim et al.* [16] described a scanning activity in terms of traffic models, working at flow-level and detecting the scanning anomalies through the analysis of variations in the models. *Chan et al.* [18] proposed two machine learning methods, useful for the constructing of models detecting network anomalies starting from past behavior.

The approach described by *Wagner et al.* [22] is based on probabilistic measurement of entropy, used to indicate regularity in traffic of network flows. Traffic models used in the three last works can be sensitive to changes in the type of traffic and network. Threshold-based approaches have been widely and successfully used in the literature [14]. In this work, we want to update these approaches to the current transmission rates and network technologies, and without linking the analysis to a specific point in the network.

MAWILab team proposed a system to detect attacks or anomalous events, applying a combination of four detectors with different

theoretical backgrounds (see Sec. 5.1). They calculate a measure of distance from normal traffic using a combination of four techniques, each based on different theoretical backgrounds: Principal Component Analysis (PCA), Gamma distribution, Kullback Leibler (KL) divergence, and Hough transformation. They then use such distance to detect anomalies in MAWI traffic traces, publishing the result of the anomaly detection every day, together with the related traffic trace. This important dataset is currently used as a gold standard in the literature [6]. *Casas et al.* proposed the combined use of a Big Data framework and machine learning algorithms to achieve high performance in terms of speed of execution and detection performance. They analyzed five types of anomalies. We focus on the entire class of port and net scan and use a much simpler detection algorithm. Moreover, we uncover that MAWILab - the ground truth they, as many other works in the literature, use - is incomplete. This clearly jeopardizes the results obtained. Therefore, we also propose an improved dataset, obtained through a combination of MAWILab and our algorithm.

4 OUR APPROACH

4.1 A simple detection algorithm

The algorithm we use for port and net scan detection is described below. Fig. 2 shows its flow chart. Input data are flow-level information. In particular, we focus on the timestamp, the IP addresses, and the transport-layer ports of each flow. For our experiments, we derived the flow-level information processing the packet traces from MAWI with a tool named TIE (Traffic Identification Engine) [2]. This tool combines the packets into flows using five fields: Source IP Address, Destination IP Address, Source Port, Destination Port, Protocol. Then, our algorithm divides the flow-level trace into time intervals of custom duration (e.g. 30 seconds). Afterward, we use Spark SQL to calculate the ratio between generated and received flows in each interval and from each IP address. This proportion is then compared with a **threshold value**. The IP addresses whose proportion is larger than the threshold are marked as anomalous (see Fig. 2).

It is worth noting that, even if very simple, the algorithm is still quite robust. For instance, it will not mark as anomalous hosts that generate a large amount of, even unbalanced, flows (e.g. servers serving popular applications) because a few responses from the other hosts (e.g. the clients) is sufficient to re-balance the equation. As we will see in Sec. 5.2, this simple algorithm is able to detect port and net scan anomalies with high precision and recall and in very short execution time if run on Apache Spark. It is also worth specifying that the algorithm cannot detect other types of anomalies or more sophisticated attacks by design in its current formulation. But 1) it does not require traffic sampling or modeling, and 2) it is able to detect all types of port and net scans unlike others that work only for a subset of them [6].

4.2 Using Apache Spark

Apache Spark is a platform for fast and efficient distributed processing of Big Data which has almost substituted Hadoop [3]. It is very fast both in data storage and processing because it supports

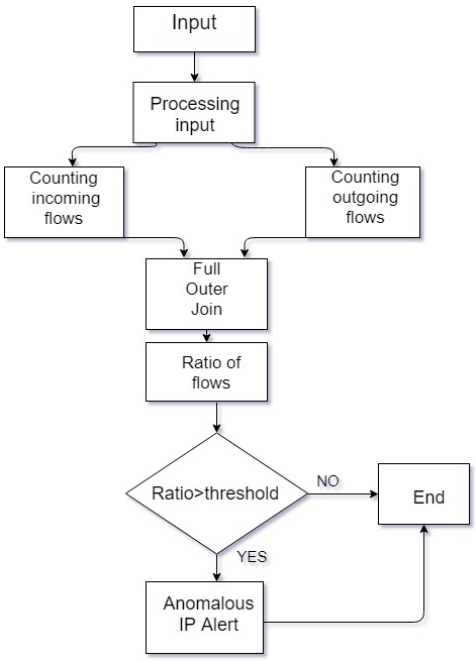


Figure 2: Flow chart of the threshold-based algorithm.

in-memory processing, i.e. analyzing data directly in main memory without recurring to mass memories [19].

Spark can work in two different ways: **Batch** and **Streaming**. Both modes [4] have been used in this work. Apache Spark allows data storage in three different types of data structure: Resilient Distributed Dataset (RDD), Dataframe, and Dataset. In this work, the DataFrame structure is used, which is basically equivalent to a table in a relational database. In fact, it is also possible to execute SQL queries on DataFrames.

Apache Spark supports different programming languages, e.g. Java, Python, and Scala. We used Scala for two main reasons: i) Apache Spark is built on Scala and so debugging is easier; ii) Scala is about 10 times faster than others (e.g. Python) to analyze and to process data due to the presence of the Java Virtual Machine.

5 DATA AND METHODOLOGY

5.1 Data used

We used several real traffic traces from the MAWI (Measurement and Analysis of the Wide Internet) dataset, an archive of real traffic traces provided by the MAWI Working Group [5].

Traces are captured since 2007, and they constitute a very rich dataset that includes different applications and network conditions, and also comprise various known anomalies with global or local impact, periods of congestion, and network reconfiguration. Traffic traces considered are captured on a transoceanic link between Japan and United States of America. Each trace contains packets captured every day from 14:00 to 14:15 in different locations inside the WIDE

[2] Traffic Identification Engine http://tie.comics.unina.it
[3] Welcome to Apache Hadoop!https://hadoop.apache.org
[4] Spark Streaming - Spark 2.3.0 Documentation. https://spark.apache.org/docs/latest/streaming-programming-guide.html
[5] Mawi working group traffic archive. http://mawi.nezu.wide.ad.jp/mawi/

network. Traces of 24 and 48 hours are also occasionally collected. A typical 15-minute trace contains 300k-500k unique IP addresses [9]. We use traces captured at Samplepoint-F, a link working at 1 Gbps with a current average load of 650 Mbps that has largely increased in recent years [4].

The MAWI group also created the MAWILab project: a novel approach for network anomaly detection, also implemented in a system that automatically runs every day on a traffic trace from MAWI repository. MAWILab defines a distance from normal traffic to recognize anomalies in MAWI traffic traces. This distance is calculated though the combination of four anomaly detectors based on different theoretical backgrounds: Principal Component Analysis (PCA), Gamma distribution, Kullback Leibler (KL) divergence, and Hough transformation. These detectors only work on the IP header [10, 15].

The results of these detectors are combined to classify the anomalies in four types:

- **Anomalous** - assigned to all abnormal traffic and should be identified by any efficient anomaly detector;
- **Suspicious** - assigned to all traffic that is probably anomalous but not clearly identified by their method;
- **Notice** - assigned to all traffic that is not anomalous, but has been reported by at least one detector;
- **Benign** - all the rest of the traffic where no detector has labeled it as abnormal.

MAWILab provides the results of the analysis in two files, *Anomalous* and *Notice*. After detecting the anomalous behaviors, MAWILab applies a heuristic to assign a label related to the type of anomaly. Possible labels are represented in a tree-based taxonomy, where the root is a generic event and nodes contain an anomaly label.

In the first part of this work, we used MAWILab archive as a ground truth [13] to validate our method (see Sec. 5.2). Afterwards, we verified that many anomalies were not detected by MAWILab, and built a new dataset on which we performed further analysis (see Sec. 6.2).

It is worth noting that MAWILab database helped and still helps a lot of researchers to evaluate the performance of novel anomaly detectors. The availability of traffic traces is already scarce. Labeled traces, including an indication of anomalies inside them, are very very rare in our research community, and this is a great obstacle to further studies on this topic. For this reason, we decided to also evaluate MAWILab accuracy, and we finally managed to improve it. In particular, our dataset [1] includes a larger set of port and net scans not detected by MAWILab, which we believe is an important contribution for the research community.

5.2 Methodology

The execution of the anomaly detection algorithm, described in Sec. 4.1, provides in output the IP addresses that are sources of port or net scans. In the first part of our experimentation we have used MAWILab as ground truth, i.e. we have compared the addresses, detected by our algorithm, with the ones in the *Anomalous* and *Notice* files provided by MAWILab.

The IP addresses detected by our algorithm that are also in one of these two files are considered true positives. The results of this analysis are reported in Sec. 6.1. We considered as false positives the ones detected by us and not by MAWILab and false negatives the ones detected by MAWILab and not by us. For our positives, besides comparing with MAWILab, we also manually verified that such IP addresses are actually source of an anomaly. This analysis evidenced the limitations of MAWILab: several anomalies we detected were not present in both MAWILab files (i.e. were also not considered suspicious by MAWILab). As explained in more details in Sec. 6.2, we confirmed this result with several manual inspections and automatic processing of the pcap files. Starting from this important result, we constructed a new dataset extending MAWILab with other anomalous flows and used such dataset as a ground truth for another experimental analysis, reported in Sec. 6.2.

The detection performance of our anomaly detector is evaluated using two main metrics: **Recall**, also called True Positive Rate, and **Precision**. Several traffic traces have been analyzed in our experimentation. Results reported in the following refer to 60 traces, characterized by a duration of 15 minutes and collected from December 2017 to September 2018. Similar results have been obtained on the other traces. We have carried out multiple tests on different values of the threshold ranging from 20 to 200. A sensitivity analysis for this important parameter is reported in Sec. 6.3. Most of the following results are then presented for the three threshold values that are more interesting: 50, 100, 200.

6 RESULTS OF THE EXPERIMENTS

6.1 Using MAWILab as a ground truth

In this section, we analyze the results we obtained using MAWILab as a ground truth. In particular, we used the *Anomalous* and *Notice* files from MAWILab and considered only scanning anomalies, which are the ones our detector has been designed for. All anomalies that are not part of scanning activities have been removed from MAWILab results (e.g. normal events, Denial of Services, Distributed Denial of Services). Since the aggregation of packets into flows does not work well for ICMP, due to unbalanced reduction compared to TCP/UDP, ICMP anomalies are not considered.

Fig. 3 shows the values of the Recall and Precision obtained. In particular, we report the box plot of the precision and recall obtained for all the traces considered. Such figures illustrate the median value of the recall ranges from about 0.65 to about 0.4 increasing the threshold value. The median value of the precision ranges from 0.3 to 0.65 increasing the threshold value. Using a threshold value of 100 we can obtain about 0.55 for the recall and about 0.5 for the precision.

These results may seem to indicate that a threshold-based approach does not allow to obtain satisfactory results and cannot be used to detect such anomalies. We then analyzed the false positives in more details to validate this hypothesis. We performed manual inspection of the pcap trace starting from the false positives. Such inspection revealed that the most part of the false positives was actually a source of scanning activity and MAWILab was unable to detect them. For example, we noticed that several IP addresses generate flows with one or two packets, mostly with the TCP SYN and ACK flags set, and they receive zero or a very small number of answers. In addition, the number of useful bytes, i.e. bytes of the TCP payload, is usually zero. These results have led to reconsider them as IP addresses generating port and net scan.

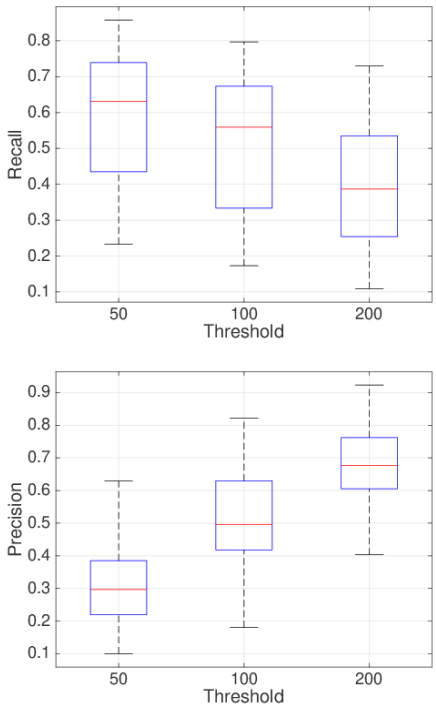


Figure 3: Recall (above) and Precision (under) of the threshold-based approach using MAWILab as ground truth.

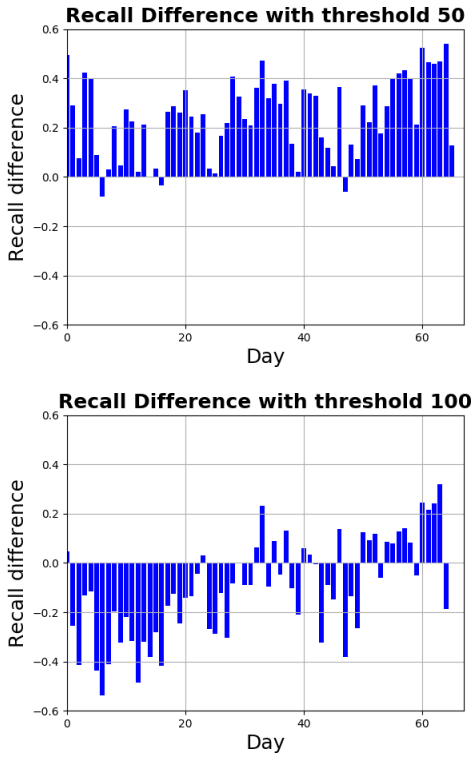


Figure 4: Difference between the Recall of the threshold-based approach and the one of MAWILab using the new dataset as ground truth.

We confirmed this result using several traces. An important implication of this result is that we can not consider MAWILab as a ground truth as done up to now in different works in the literature.

6.2 Constructing and using the new ground truth

We built a new dataset expanding MAWILab with the IP addresses that have been found abnormal. In particular, based on the results of the previous analysis, we implemented two heuristic rules to complement MAWILab results for all the cases in which a source of the scanning activity was not detected by MAWILab.

IP addresses that are false positives and trigger such rules are reintegrated into the true positives. The rules are the following: i) An IP address is a generator of **Net Scan** if it generates at least 20 flows towards different IPs of the same subnet; ii) An IP address is a generator of **Port Scan** if it contacts the same destination IP address on more than 10 ports. Using these rules on several traces, we have built a new dataset to evaluate the performance of the threshold-based algorithm detector. This new dataset is obtained by the union of MAWILab results and the list of IP addresses that are considered a source of scanning activities after the application of the rules implemented.

In the experiments described in the following, we compared MAWILab with the threshold-based approach and used the new dataset as a ground truth. Fig. 4 shows the difference between the recall of the threshold-based algorithm and the one of MAWILab as a function of the different traces analyzed. The figure shows that the recall of the former algorithm is larger than the one of MAWILab in 95% of the cases when the threshold value is 50 and in 33% of the case with a threshold value of 100. When we increase the threshold value, MAWILab starts to obtain better performance in terms of Recall.

The values of recall for the threshold-based algorithm are reported in Fig. 5 (a). The precision of MAWILab is clearly equal to 1 because there are no false positives.Fig. 5 (b) shows the precision of the threshold-based algorithm. We can see that for the intermediate threshold value (i.e. 100) the precision is larger than 0.85 in about 70% of the cases, and the median value is about 0.88. Higher precision values can be obtained with higher threshold values.

Comparing the results in Fig. 3 and Fig. 5 we can see the improvement of the performance of the threshold-based approach with the new dataset. Moreover, we can say that such very simple approach allows obtaining very high performance, higher than MAWILab which uses a much more complicated and therefore less observable approach.

Summarizing, in this section we have shown that a very simple detection algorithm can obtain an even better Recall than MAWILab (in 95% of the cases with a threshold 50 and in 33% of the cases with a threshold 100), and this is because MAWILAb is not able to detect all scanning activities in MAWI traces.

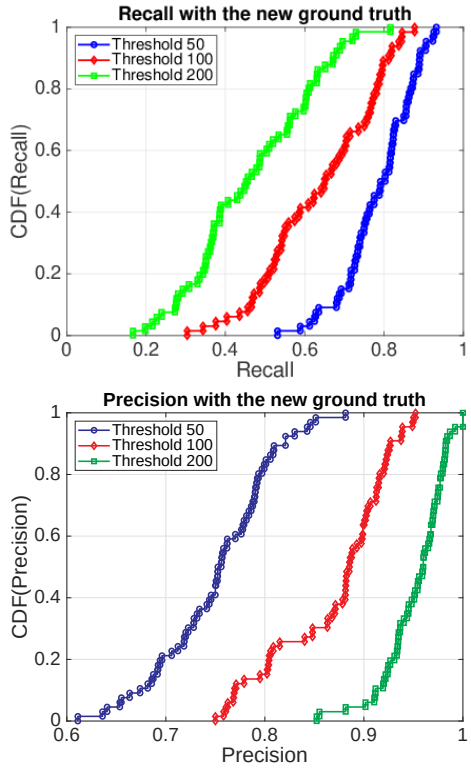


Figure 5: Recall (a) and Precision (b) of the threshold-based approach using the new dataset as ground truth.

6.3 Analysis of sensitivity to the threshold

Fig. 6 shows the average values of recall and precision obtained for several threshold values ranging from 20 to 200 using the new dataset as a ground truth.

This figure shows that the best threshold value depends on which metric you want to maximize. For example, if false positives are annoying for a human intervention in the security pipeline, a good threshold value is about 125. For such value, an average precision of about 0.9 can be obtained. On the other side, if false negatives are more a problem, 50 is the best threshold value to have a high recall without losing too much precision. An optimal value for both metrics is 80, which allows obtaining an average recall and precision of about 0.85.

6.4 Speed analysis on Amazon EMR

We carried out several experiments on Amazon Elastic Map Reduce[6] to analyze the execution time of the algorithm using various cluster configurations and to understand the impact of the different variables under our control (number of workers, resources per VM, availability zone) on this important performance parameter.

For this analysis, we have chosen MAWI traffic traces that were captured for 24-hours on the Samplepoint-G, the main IX link of WIDE to DIX-IE with a speed link of 10 Gbps. On Samplepoint-G, MAWI provides two 24-hours traces (from the 9th of May 2018 and

[6]Amazon EMR – Amazon Web Services https://aws.amazon.com/it/emr/

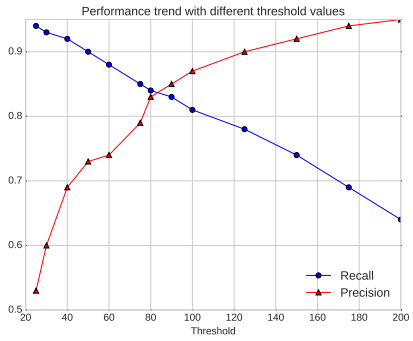


Figure 6: Average Recall (blue) and Precision (red) using the new dataset as ground truth for several threshold values.

the 9th of April 2019), where the first one is characterized by a smaller average bit rate with respect to the second one.

Three Cloud Availability Zones have been considered: Ohio (US), Ireland (Europe), and Tokyo (Asia Pacific). We have instanced four configurations in each of them as shown in Tab. 1. Fig. 7, related to the trace collected on the 9th of April 2019, shows that the two availability zones with the best execution times in terms of average and variance are Ohio and Tokyo. Ireland, on the other hand, has high average execution times and high variance. Moreover, OHIO and Tokyo show a clear improvement in performance as computing resources increase. Ohio has better results than Tokyo for this trace.

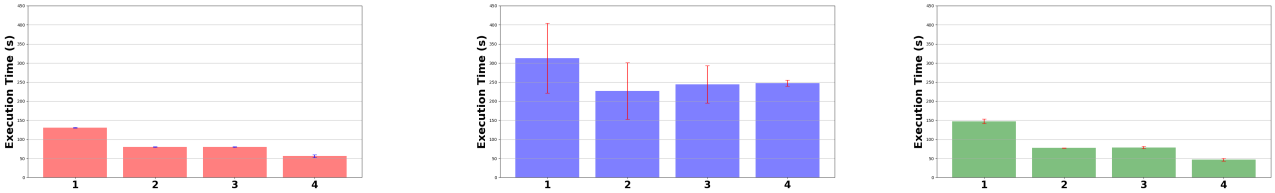
We can observe the opposite behavior for the trace collected on the 9th of May 2018, which results are reported in Fig. 8. Also in this case, Ireland has low execution times in terms of both average and variance. Like in the previous case, there is an improvement in performance when computing resources increase, although less marked than in the previous case. A significant result of this analysis is the best average time of execution of our algorithm on a 24-hour track, processed in about 25 seconds. This average value is obtained in the second track in the availability Zone of Tokyo with the configuration m5.2xlarge.

	Model	#Master	#Worker	vcpu	Memory (GiB)
1	m5.xlarge	1	2	4	16
2	m5.xlarge	1	4	4	16
3	m5.2xlarge	1	2	8	32
4	m5.2xlarge	1	4	8	32

Table 1: Configurations of EMR Clusters

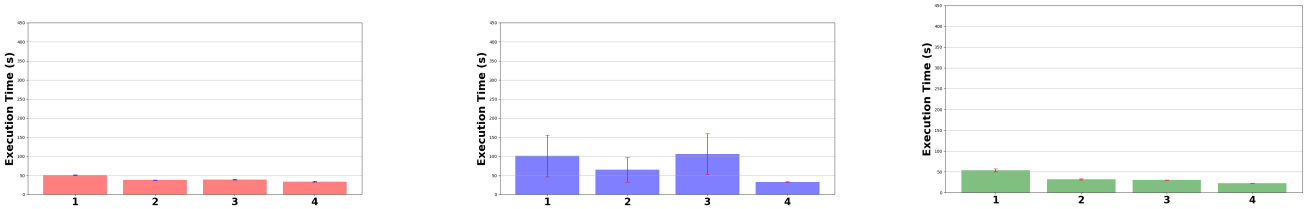
7 CONCLUSIONS

Anomaly detection in network traffic is a hot topic in scientific literature, and it is in continuous evolution. Many proposals have been presented in this important research area, but still today these kinds of anomalies are difficult to be detected efficiently and with high precision. In this paper, we have firstly shown that port and net scans, which are well known malicious activities, are still increasingly spread in recent years.



(a) Ohio - USA (b) Ireland - Europe (c) Tokyo - Asia Pacific

Figure 7: Execution Time on Amazon EMR - first trace



(a) Ohio - USA (b) Ireland - Europe (c) Tokyo - Asia Pacific

Figure 8: Execution Time on Amazon EMR - second trace

Several research efforts are currently put on complex techniques for anomaly detection, such as deep learning. These techniques can provide good detection performance, but their observability is limited. Our idea, instead, was to recover traditional detection approaches and resort to novel computing frameworks for obtaining the performance required by current network traffic.

In particular, we used a threshold-based algorithm, working at flow-level. The basic idea of this algorithm is to recognize malicious hosts looking at the ratio of their fan-in and fan-out (i.e. the number of outgoing and incoming flows). Though very simple, this approach can obtain good detection performance, but it has scarce performance in terms of processing time. To cope with this problem, exacerbated in high-speed networks, we used Big Data Analytics and Apache Spark in particular. We conducted an experimental analysis with several real traffic traces from the MAWI archive. We also used MAWILab anomaly detection results as a ground truth in the first part, and a comparison in the second one, after recognizing that MAWILab fails to detect several scans.

We firstly evaluated the precision and recall of the threshold-based algorithm using MAWILab as a ground truth. Results were not satisfactory, in particular in terms of false positives, and pushed us to investigate more in deep. Through manual inspections of several traffic traces, we verified that such positives were actually true rather than false. This drove us to create a new dataset starting from MAWILab and complementing it with several other anomalies identified through heuristic rules devised thanks to the analyses described above.

We then evaluated the performance of the threshold-based algorithm using the new dataset as ground truth and compared obtained results with the ones from MAWILab. This analysis shows that the simpler algorithm can easily achieve higher recall than MAWILab, which is already based on much more complex algorithms. We executed the algorithm also on Amazon EMR to analyze the average execution time on three Availability Zones (Ohio, Ireland, and Tokyo) and four different cluster configurations. We showed that the fastest availability zones are Ohio and Tokyo and that our algorithm can process 24h of traffic collected over a 10Gbps link in about 25 seconds in Tokyo Availability Zone with m5.2xlarge configuration.

Finally, we also set up a system that processes the traces available from MAWI every day and publishes a new dataset, including a comparison with MAWILab [1]. We believe this was an important yet missing contribution for further studies on this research topic.

ACKNOWLEDGMENTS

The work is partly supported by GRISIS project (CUP: B63D180002800079), DD MIUR prot.368 of 24/10/2018, Programma Operativo FESR Campania 2014-2020. The work is also supported by Amazon through AWS Research Credits.

REFERENCES

[1] 2019. The Spada Research Project. http://spada.comics.unina.it. (March 2019).
[2] G. Aceto, A. Botta, A. Pescape, and C. Westphal. 2013. Efficient Storage and Processing of High-Volume Network Monitoring Data. *IEEE Transactions on Network and Service Management* 10, 2 (June 2013), 162–175. https://doi.org/10.1109/TNSM.2013.011713.110215
[3] S. Balram and M. Wiscy. 2008. Detection of TCP SYN Scanning Using Packet Counts and Neural Network. In *2008 IEEE International Conference on Signal Image Technology and Internet Based Systems*. 646–649.
[4] P. Borgnat, G. Dewaele, K. Fukuda, P. Abry, and K. Cho. 2009. Seven Years and One Day: Sketching the Evolution of Internet Traffic. In *IEEE INFOCOM 2009*. 711–719.
[5] A. Botta, W. de Donato, A. Pescape, and G. Ventre. 2007. Discovering Topologies at Router Level: Part II. In *IEEE GLOBECOM 2007 - IEEE Global Telecommunications Conference*. 2696–2701. https://doi.org/10.1109/GLOCOM.2007.511

[6] P. Casas, F. Soro, J. Vanerio, G. Settanni, and A. D'Alconzo. 2017. Network security and anomaly detection with Big-DAMA, a big data analytics framework. In *2017 IEEE CloudNet*. 1–7.

[7] Cliff C. Zou, Don Towsley, and Weibo Gong. 2006. On the performance of Internet worm scanning strategies. *Performance Evaluation* (2006), 700 – 723.

[8] A. Dainotti, A. Pescape, and G. Ventre. 2006. NIS04-1: Wavelet-based Detection of DOS Attacks. In *IEEE Globecom 2006*. 1–6.

[9] R. Fontugne, P. Abry, K. Fukuda, D. Veitch, K. Cho, P. Borgnat, and H. Wendt. 2017. Scaling in Internet Traffic: A 14 Year and 3 Day Longitudinal Study, With Multiscale Analyses and Random Projections. *IEEE/ACM Transactions on Networking* (Aug 2017), 2152–2165.

[10] Romain Fontugne, Pierre Borgnat, Patrice Abry, and Kensuke Fukuda. 2010. MAWILab: Combining Diverse Anomaly Detectors for Automated Anomaly Labeling and Performance Benchmarking. In *Proceedings of the 6th International Conference*. ACM Conext 2010, New York, USA, Article 8.

[11] Luigi Gallo, Alessio Botta, and Giorgio Ventre. 2019. Identifying Threats in a Large Company's Inbox. In *Proceedings of the 3rd ACM CoNEXT Workshop on Big DAta, Machine Learning and Artificial Intelligence for Data Communication Networks (Big-DAMA '19)*. ACM, New York, NY, USA, 1–7. https://doi.org/10.1145/3359992.3366637

[12] J. Quittek, T. Zseby, B. Claise, S. Zander. 2004. RFC3917: Requirements for IP Flow Information Export (IPFIX). (2004).

[13] Jurgen A H R Claassen. 2018. The gold standard: not a golden standard. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC557893/. (August 2018).

[14] Maciej Korczynski et al. 2011. An Accurate Sampling Scheme for Detecting SYN Flooding Attacks and Portscans. In *ICC'11*.

[15] J. Mazel, R. Fontugne, and K. Fukuda. 2014. A taxonomy of anomalies in backbone network traffic. In *IWCMC 2014*. 30–36.

[16] Myung-Sup Kim, Hun-Jeong Kong, Seong-Cheol Hong, Seung-Hwa Chung, and J. W. Hong. 2004. A flow-based method for abnormal network traffic detection. In *2004 IEEE/IFIP Network Operations and Management Symposium (IEEE Cat. No.04CH37507)*, Vol. 1. 599–612 Vol.1.

[17] Valerio Persico, Alessio Botta, Pietro Marchetta, Antonio Montieri, and Antonio Pescapé. 2017. On the performance of the wide-area networks interconnecting public-cloud datacenters around the globe. *Computer Networks* 112 (2017), 67 – 83. https://doi.org/10.1016/j.comnet.2016.10.013

[18] Philip K. Chan, Matthew V. Mahoney, and Muhammad H. Arshad. 2005. LEARNING RULES AND CLUSTERS FOR ANOMALY DETECTION IN NETWORK TRAFFIC. MACO, volume 5. (July 2005).

[19] Abdul Ghaffar Shoro and Tariq Rahim Soomro. 2015. View of Big Data Analysis: Apache Spark Perspective. Global Journals Inc. (USA). (2015).

[20] A. Sperotto, G. Schaffrath, R. Sadre, C. Morariu, A. Pras, and B. Stiller. 2010. An Overview of IP Flow-Based Intrusion Detection. *IEEE Communications Surveys Tutorials* (Third 2010), 343–356.

[21] A. Sridharan, T. Ye, and S. Bhattacharyya. 2006. Connectionless port scan detection on the backbone. In *2006 IEEE International Performance Computing and Communications Conference*. 10 pp.–576.

[22] A. Wagner and B. Plattner. 2005. Entropy based worm and anomaly detection in fast IP networks. In *14th IEEE International WETICE'05*. 172–177.

[23] Q. Zhao, J. Xu, and A. Kumar. 2006. Detection of Super Sources and Destinations in High-Speed Networks: Algorithms, Analysis and Evaluation. *IEEE Journal on Selected Areas in Communications* (Oct 2006), 1840–1852.