ISSN (Print) : 0974-6846
Indian Journal of Science and Technology, *Vol 9(26), DOI: 10.17485/ijst/2016/v9i26/97255, July 2016* ISSN (Online) : 0974-5645

Immediate Detection of DDoS Attacks with using NetFlow on Cisco Devices IOS

Mahmoudreza Tahmassebpour

Computer Science and Information Technology, University of Malaya, Kuala Lumpur, Malaysia; mahmoudtahmasebpour@gmail.com

Abstract

Background/Objectives: DDoS attacks are usually detected by analysis of the applications that are installed in or close to the current system are carried out. **Methods/Statistical Analysis:** Although this method is easy to deploy, but non-urgent and sensitive detection of DDoS attacks that reasons are first, the fact that the write current by interrupting the current collector is normally the data for application analysis creates pieces that caused a delay of several minutes to be recognized. Second, if the attack traffic may be strengthened by the process of sending the original package small enough to be part of a small stream. **Findings:** In this research paper will show how to detect DDoS attacks on the sender instead of the current collection, the data close to the source and immediate fashion, which had access to a streaming surveillance infrastructure with development needs. In this study, to examine whether the detection system may operate on the same network platform is widely deployed Cisco IOS devices. Since the ultimate goal of the research is to identify the attackers and its objectives, the use of NetFlow. **Applications/Improvements:** In this paper, the DDoS attack detection prototype has been shown to produce a constant load on the underlying platform, even under attack, stressing that detects DDoS attack can be a Cisco Catalyst 6500 models used in production networks.

Keywords: Computer Network, Cisco IOS, Detection, DDoS Attack, NetFlow

1. Introduction

The Distributed Denial of Service (DDoS) attacks are becoming a major limitation of technical and economic, overload networks and servers with large amounts of traffic on the network. The Internet is an open architecture susceptible to various forms of network attacks. A primary example is DDoS attack. There are many DDoS attack methods, and flooding packets attack is the most common and powerful tool for hackers[1,2]. In early 2014, the site CloudFlare by a UDP flood attack reinforced by reaching close to 400 Gbps bandwidth traffic was disrupted[3]. Although UDP flooding attacks typically target overload by high congestion bytes, as well as other attacks such as TCP SYN flood attacks are the result of the large number of connections. The definition of a stream, "a set of packets through a network point during a certain timeframe, so that all packets belonging to a particular flow have a set of common properties"[4], it is also the result of a lot going on. This makes it possible for flow-based technologies for the detection of such a volume-based attacks makes[5,15]. In addition, the use of current technology exports, such as NetFlow and just trying to IETF standard IPFIX, specifically useful for the production of traffic. This approach significantly reduces the amount of data for analysis[6] and as well as the necessary processing power for export and collected. Moreover, these technologies are widely available on devices and data flow packets are readily available for deployment in existing networks makes.

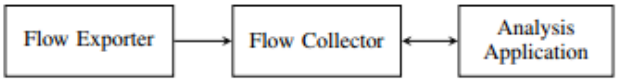

Figure 1. Typical flow monitoring architecture.

In general, flow-based intrusion detection, traditionally done by analysis applications other than DDoS attack[7-9] as shown in Figure 1. The applications based on flow data exported by exporters flow and collectors gathered by the act. Since the export flow data collection relies heavily on the time lag is often designed to operate at intervals

* *Author for correspondence*

of a few minutes, the application analysis process has a different delays diagnosis[10]. Especially in the case of attack detection DDoS, that it can happen very quickly overload the network infrastructure, this is what should be avoided.

Recent work has shown that motion detection closer to the data source; reducing significantly delay will be the detection of at least 10 seconds to 165 seconds[11]. Offer DDoS attack detection algorithm on a target platform with non-active export-based data streams, the platform INVEA-TECH's FlowMon applicable. The purpose of this paper is to examine whether recognition algorithm presented in[11] can be deployed in a platform-wide network available. In this context, the Cisco IOS platform and is targeted for the Catalyst 6500, which is one of the most widely deployed packet forwarding devices[12]. Especially in the experience of operating packet network intrusion detection devices production is concentrated.

2. Export and Flow Measurement

In this section, amendments to the Export and measurement introduced that will be used throughout this article. For a comprehensive overview of NetFlow and IPFIX, referring to the tutorial[6].

Export and measurement of double duty by an exporter flow[6], as shown in Figure 1. Depending on the current network are collected by the measurement process. When a new flow can be seen, for entry into the flow of a stream is created secret. This cache is a table that stores information in network-enabled[6]. Apart from the key process of identifying areas of a stream, some additional information usual, such as the number of packets and bytes were in a current account. In the event that the cache is full and a current input cannot be made secret visit, which can occur during periods of high traffic if the current cache size, is going to be a learning failure[13]. When a flow cache entry expires, for example, when the active or has been unemployed for a long time or because of resource constraints, the current record is issued, NetFlow or IPFIX the message and put it in a collector for storage and preprocessing sent.

3. Detection Algorithm

An existing algorithm to meet the requirements for lightweight, precise and immediate in detecting DDoS attacks were used, in described[11]. Algorithm on a fixed time interval and secretly created by measuring the number of input flow, as measures can be used in more than four criteria applicable to[11]. Under this measure, a provision for measuring the next interval is created. The number of entries in the cache created compared to past measurements is too high, the measurement sample is considered unusual. However, due to Internet traffic that daily pattern, such as a strong increase and decrease in the number of the cache created at the beginning and end of the day, respectively, the algorithm learns the normal behavior of the network in a 24-hour period. Predictive value in this case is defined as:

$$\hat{x}_{t+1} = b_t + s_t \quad (1)$$

Where $\hat{x}_{t+1}$ value is forecast for the next interval, b_t the base component, sometimes as a permanent component, which reflects the seasonal trend of Internet traffic and St components that represent the daily pattern.

Several enhancements to the algorithm in[11] discussed. First, to reduce memory use, the seasonal patterns used worth preserving. S_t, at any time and for estimating the value stored for a given time is inserted. Second, to avoid learning algorithm of malicious traffic patterns, such as S_t and b_t values are discarded during an attack. Finally, the weekend traffic patterns are different from the usual patterns during weekdays, weekends and weekdays for a distinction between seasonal memories are made. This result in two training models, one for weekdays and weekends is another.

4. Monitoring Information Available on IOS

Detection algorithms in this article are intended in section 3, in a single size, the number of current cache entries in each time slot will be created. This size is easily accessible on the monitoring platform in the core business of the prototype[11], INVEA-TECH's FlowMon. Because the platform is designed with extensibility in mind, this information is directly available from the Platform API. However, the information available on IOS is highly dependent on the path is closed, or the stream will depend on the router and switch. More precisely, depending both on the hardware and the software switch, although many packages are hardware switches. For example, in the network of the University of Twente (UT), 99.6% of traffic-related hardware switch[13]. The

reason for this is that depending on the software switch, packet fragmentation, the packets are sent to their own devices, for example, packages that need to have ARP[14]. For stream processing hardware, data on the number of current cache entries made directly available. To get closer to this metric, metric information available and export processing flow used:

- The number of entries in the cache (F_c).
- The number of records issued during the software switch (F_e).
- The number of errors during learning (F_f). This metric rather than the flow of packets.

The number of current cache entries created from the last measurement can be estimated using the following definition:

$$F = \Delta F_c + \Delta F_e + \Delta F_f / c_f \quad (2)$$

When the cache entry is issued, F_c will decrease, which will be less accurate approximation if measured distances are too long. For example, in Figure 2, if the measure is to cover two periods, from t = 2 to t = 4, ΔF_c in spades t = 3 will not be considered. F_c by sampling at greater distances, it can be more accurately seen changes, so that positive ΔF_c at t = 3 and negative ΔF_c at t = 4 can be seen, which is created by export. Then, if ΔF_c is negative, an estimate of amounts previously ΔF_c used instead. When the cache is nearing its capacity limit, the issuer performs an emergency expiry[6]. In Figure 2 it is depicted in the shaded area. F_c is also tacit C_m flow capacity reaches that should be expired from the cache entry. If a measurement is performed between t = 6 and t = 7, the algorithm may take it as an attack for a measured distance due to the large increase in the number of cache entries identified with respect to t = 6. To counteract this, the implementation waits for the next measurement if it suspects an attack rather than a real attack. This diagnosis is delayed.

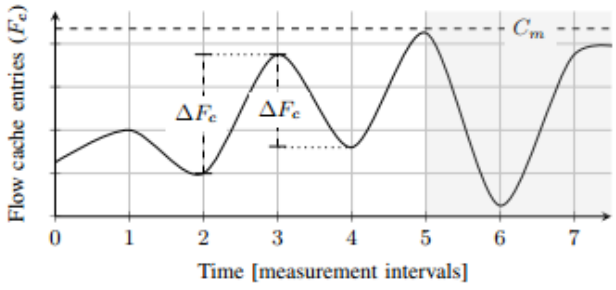

Figure 2. Flow cache entry creations in Cisco IOS over time.

Because the number of entries in the cache (F_c) is just about the hardware switch, the change current software exports (F_e), which can be acquired directly from IOS to be added. The add F_f be allowed for the flows that have been created but have not, especially in the case of DDoS attacks with high intensity. For example, to compensate for the fact that F_f expressed in packets that flow between the other metrics, F_f divided by the average number of packets in the flow, is provided by cf in Equation 2.

5. Implementation

EEM is part of Cisco IOS that uses real-time detection of network events, define policies can be used to allow the applet to run a script when events occur or applied. For example, when the network load reaches a certain limit or change occurs in the network to provide network administrators can be emailed. Another time-based event. Among other events, this event can be scheduled at fixed intervals. In this work, using time-based policy runs like TCL scripts:

- Policy measures: Determine the first component to estimate the flow-based metrics. The number of entries in the cache (F_c), described in Section 4.
- Recognition policy: Recover the remaining components. Total current software exports (F_e) and the number of errors during learning (F_e). As well as the implementation of a real DDoS attack detection algorithm.

To get all three components, all of which are made available using SNMP, using the EEM feature that provides access to SNMP objects. Divide the measurement politics of policy more clearly recognize the need for a more accurate diagnosis changes that are described in Section 4.

Policy invocations are memory less, and because we want to share that both between politics and policies are implemented, a method for sharing data required running. Due to the fact that the file system is flash-based, we generally want to avoid excessive write actions that will shorten the memory's lifespan. The EEM environment for this purpose offers a library of text; it changes TCL memory for storage instead of writing them to disk allows. In addition to keeping track of data between policy runs, also use this feature to exchange information between the two policies are used, as a result of policy measures are needed recognition by politics.

Talk to EEM policies by their respective distances based on the appropriate policies is implemented. When the switch is under a heavy load with the use of more powerful CPU usage policy is current. To prevent this policy from executive Jump over when policy during the prototype away enjoying the EEM can adjust the maximum runtime. If you take too much time to end the policy of forced loss of information. In recognition policy, the algorithm starting up again with a learning phase for all the lost data from the action. If policy measures prematurely terminated, the current measurement inputs less secret created will be lost as a measure of the accuracy of the algorithm will be little affected. To avoid loss recognition policy, a margin is added to the distance which allows it to run longer if necessary, but not more than the distance at which it is executed. The average time policies distinguish between 2 to 3 seconds in normal circumstances and pressures between 7 and 8 seconds. So are the ultimate choices for the diagnosis policy is 10 seconds. For policy measure, the measure appears that 2 seconds optimal balance between precision measurements and provides data loss to end.

6. Validation

In this section, the validity of the work described, start by identifying the requirements in Section 6, Part 6.1, followed by a description of the setting up of credit as well as details about deploying in Part 6.2 and the results discussed in Part 6.3 Takes.

6.1 Requirements

Three requirements were defined for the original detection algorithm:

- It must be very light in terms of CPU and memory use,
- The accuracy should be high enough for the low number of false positive and negative,
- Delayed diagnosis has almost 10% of the common intrusion detection method[11] is less.

However, because Cisco devices with high-speed packet forwarding Catalyst 6500 models are not designed for intrusion detection tasks. Proper care should be taken to not exceed the load limit device and have not been cut off. So the need for immediate assessment of detection is 30 seconds or less 10% of the CPU and should be used.

6.2 Setup and Deployment

The implementation described in Section 5 on a Cisco Catalyst 6500 supervisor engine 720 and IOS version 15.1 (2) SY1 has been developed. The WS-X6708-10G-3C card to the device with Ethernet connectivity is added. The traffic used for validation is mirrored from the uplink of the UT campus network to the Dutch National Research and Education Network SURFnet and consists of both educational traffic, i.e., traffic generated by faculties and students, and traffic of campus residences. The link speeds of 10 Gbps with average throughput of 1.8 Gbps is during working hours. In addition, data collected during the current export deals so that attacks can be detected by manually approved prototype.

The network traffic used in[11] used in this work is different from the traffic, so obviously you have to set the parameters for the accurate detection algorithm in[11] as well. The optimal parameter values (The parameters used in this work are: $c_{threshold}$ = 4.0, M_{min} = 7000,c_{cusum} = 6.0, α = 2 / (N+1), where N = 540, and γ = 0.4) for the selected observation point. Parameter c_f, to estimate the number of secret input current is generated, which is described in Section 4. c_f= 59.8133 depending on the size of the average setup.

6.3 Results

The most important requirement in this work is validated by running style that should be done so that the core business of routers and switches, the devices send a packet enters disorder. Measurement of CPU and memory resource consumption takes place. Figure 3 CPU load and memory usage on mobile devices is shown. An average of more than 150 seconds. Using SNMP CPU load on the three components of the CPU that handles traffic is L3 routing and switching the CPU to process the L2 traffic measures.

In Figure 3, the policy is active across the entire measurement period. Since in most processes, CPU usage to zero percent and only the top one percent. To measure this, switch to re-boot and all the memory and CPU usage is erased. During the measurement of a load on the CPU and memory routing by 4%, which is 31.3 percent after the activation of the policy, an increase of 20% CPU usage and increased 0.2 percent in the memory can be seen.

During the period in which the detection algorithm based authentication is a network attack on 25 August.

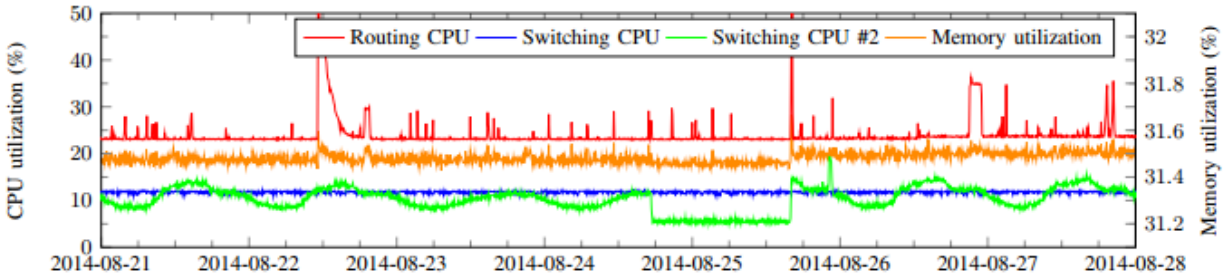

Figure 3. Load of the Cisco Catalyst 6500 over time.

The attack time is 20 minutes and includes DNS traffic and TCP. According to measurements above it was concluded that the estimated storage requirements by 10% or less. However, 20% of the CPU load of the overall performance of the 10% not satisfied.

The second requirement is delayed diagnosis. This requirement must be accurate and the prototype is validated[11]. The results of the diagnosis of multiple delays that minimum are 10 seconds. In the third attack, seen in Figure 4 are detected and diagnosed in a 30-second operation was performed. The final requirement DDoS attack detection accuracy. In Figure 4, the number of entries made in the unseen distance measurements show that the average distance of more than 5 minutes.

7. Discussion

The prototype presented in Section 5 platforms background information is retrieved using SNMP. Data recovery can be performed by any other device using SNMP, even a Raspberry Pi, by maximizing the available processing power of the device used for routing and switching. However, since the final aim of reducing attacks that require information about attackers and their targets is normally done on your diagnosis device (where NetFlow is available), for rapid deployment in production environments at no additional cost.

The detect attacks is a very important first step, which only serves the ultimate goal of reducing the attacks. In[11], discussed not only detect attacks, but it also is reduced. The specific source IP address is issued; the source IP address is blacklisted. Blacklisted IP addresses added to a firewall to block traffic striker added. In addition, to avoid overloading the gravure process, to collect flow records with this IP address could not be sent. When the algorithm detects the end of the attack, the firewall rules will be deleted.

Information used to identify the attackers is not available in IOS[11], only total latent export inputs are available. An alternative approach is to identify the attackers, analyze the contents of the cache is current. However, attackers IP addresses in the cache during a DDoS attack will be weakened as a result of a lot of input from the unseen attackers that they generate a lot of traffic. However, the time required to retrieve and process all the time hidden under that consists of at least 128 kilograms

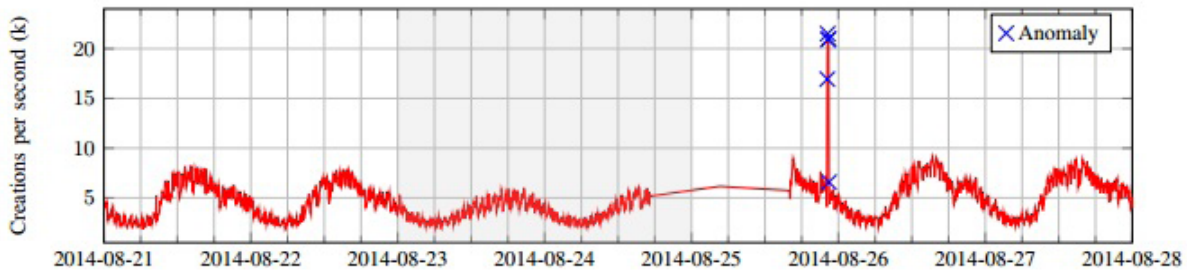

Figure 4. Flow cache entry creations per second (averaged per 5 minutes), as processed by the detection algorithm over time.

of input, depending on the hardware used, it can take ten seconds, reducing it is hardly possible.

A different approach for reducing the use of IOS features that track high scores x (0, 200) stream contains the largest amount keeps, both in terms of packets or bytes, referring to the NetFlow Top Talkers. It cannot be hosted by a number of flows generated by the show, which will be too much for a DDoS attack sources. Moreover, it is likely that legitimate users are on the list, they can only generate many packets and bytes. So it follows that identify the attackers alongside the decline in this difficult task.

8. Conclusions

The results show that the detection of flood attacks possible in ten seconds, widely available instant recognition features depends on the switch platform. However, the sample has also shown it can cause interference routing and switching processing CPU load is 20%. According to various network operators with available capacity can detect DDoS attack packets in their production environment. While enabling the deployment of its implementation with only 20 to 30 percent of CPU capacity is available, for example, requires a lower priority to non-interference in routing and switching processes. As this may cause instability to our prototype, it is advised to have at least 40% CPU capacity available.

There are a few requirements to identify: first, a small footprint necessary to implement the detection algorithm validation. Results have shown that visible increase in CPU and memory usage during raids there. However, when the monitor runs increase in CPU, 20% and 0.2% memory usage visible in the sample. While memory usage is satisfying due to the use of 10 percent or less of the available resources, requirement of load on the CPU used is not satisfactory. Second, the validation of samples has shown that the delay UT campus network to detect attacks with high intensity for 30 seconds, to detect than satisfying the above requirements. This corresponds to three times 10 seconds is measured. Measuring distances smaller may reduce the amount of delayed diagnosis, but will make it more likely that our detection runs overtime and is killed by a management process. The last recognition accuracy is required to run. The validation results show that the number of false positives is low, while the detection rate is high, then the result is high accuracy.

Reducing the next step after diagnosis. The research has shown that while it is possible to obtain sufficient information to identify the attackers, the command used to obtain this information for 10 seconds when the switch is under heavy load conditions torrential attacks occur. A result of the decrease in real time on the hardware used in this work is not possible.

Future work includes a review of alternative implementations on different hardware will be included with additional capabilities are stronger hardware. The more powerful hardware will certainly reduce the time of diagnosis.

9. References

1. Carl G, Kesidis G, Brooks R, Rai S. Denial-of-service attack-detection techniques. IEEE Internet Computing. 2006; 10(1):82-9.
2. Peng T, Leckie C, Ramamohanarao K. Survey of network-based defense mechanisms countering the dos and ddos problems, ACM Computing Survey(CUSR), 2007, 39(1), pp. 3.
3. Cloud Flare, Inc. Technical Details Behind a 400Gbps NTP Amplification DDoS Attack. 2014. Available from: http://blog.cloudflare.com/technical-details-behind-a-400gbps-ntp-amplification-ddos-attack
4. Claise B, Trammell B, Aitken P. Specification of the IP Flow Information Export (IPFIX) protocol for the exchange of flow information. Internet Engineering Task Force. 2013 Sept; 1-76.
5. Sperotto A, Schaffrath G, Sadre R, Morariu C, Pras A, Stiller B. An overview of IP flow-based intrusion detection. IEEE Communications Surveys and Tutorials. 2010; 12(3):43-56.
6. Hofstede R, Celeda P, Trammell B, Drago I, Sadre R, Sperotto A, Pras A. Flow monitoring explained: From packet capture to data analysis with Netflow and IPFIX. IEEE Communications Surveys and Tutorials. 2014; 16(4):2037-64.
7. Galtsev AA, Sukhov AM. Network attack detection at flow level. Proceedings of the 11th International Conference and 4th International Conference on Smart Spaces and Next Generation Wired/Wireless Networking; 2011. p. 326-34.
8. Nguyen HA, Tam Van Nguyen T, Kim D, Choi D. Network traffic anomalies detection and identification with flow monitoring. 5th IFIP International Conference on Wireless and Optical Communications Networks, Surabaya, WOCN'08; 2008. p. 1–5.
9. Muraleedharan N, Parmar A, Kumar M. A flow based anomaly detection system using chi-square technique. Proceedings of IEEE 2nd International Advance Computing Conference, IACC'10; Patiala. 2010. p. 285–9.
10. Hofstede R, Pras A. Real-time and resilient intrusion detection: A flow-based approach. Proceedings of the 6th IFIP

WG 6.6 International Conference on Autonomous Infrastructure, Management, and Security, AIMS'12; 2012. p. 109–12.
11. Hofstede R, BartosV, Sperotto A, Pras A. Towards real-time intrusion detection for NetFlow/IPFIX. Proceedings of the 9th International Conference on Network and Service Management (CNSM'130); Zurich. 2013. p. 227–34.
12. Follett J. Cisco: Catalyst 6500 the most successful switch ever [Online]. 2006. Available from: http://www.crn.com/news/networking/189500982/cisco-catalyst-6500-the-most-successful-switch-ever.htm
13. Hofstede R, Drago I, Sperotto A, Sadre R, Pras A. Measurement artifacts in NetFlow data. Proceedings of the 14th International Conference on Passive and Active Measurement (PAM'13); 2013. p. 1–10.
14. Cisco Systems, Inc., Catalyst 6500/6000 switch high CPU utilization [Online]. 2012. Available from: http://www.cisco.com/c/en/us/support/docs/switches/catalyst-6500-series-switches/63992-6k-high-cpu.html
15. Wong DH, Chee CM. Usable, flexible and adaptive network data visualization design for multiple levels of computer users. Indian Journal of Science and Technology. 2015; 8(15):1-7.