AUSTRALIAN JOURNAL OF BASIC AND APPLIED SCIENCES

ISSN:1991-8178 EISSN: 2309-8414
Journal home page: www.ajbasweb.com

An Overview of DNS Amplification Attack Defense via Flow-Based Analysis and SDN

[1]Ahmad Ariff Aizuddin Mohd Atan, [1]Megat Norulazmi Megat Mohamed Noor and [2]Mohd Nazri Ismail

[1] University of Kuala Lumpur, Malaysian Institute of Information Technology, 50300 Kuala Lumpur, Malaysia
[2]National Defence University of Malaysia, Faculty of Defence Science and Technology, Jalan Raja, Kem Sungai Besi, Kuala Lumpur, Malaysia

Address For Correspondence:
Ahmad Ariff Aizuddin Mohd Atan, University of Kuala Lumpur, Malaysian Institute of Information Technology, 50300 Kuala Lumpur, Malaysia

ARTICLE INFO

Article history:
Received 3 March 2016
Accepted 2 May 2016
published 26 May 2016

Keywords:
DNS Amplification Attack; Flow-Based Analysis; SDN; Flexible Flow; RRL

ABSTRACT

Distributed Denial of Service (DDoS) attack is a growing issue in the Information Age. Following the rise of IPv6 and Internet of Things (IoT), these attacks are undeniably becoming more abusive. Relatively, Domain Name System (DNS) amplification attack is one of the biggest DDoS to date. The attack leverages the fact that a single DNS request (small investment) can generate large DNS responses (amplified returns). This paper reviewed existing countermeasures and its deficiencies against DNS amplification attack that eventually lead to the need of flow-based analysis as detection approach. This paper discussed related works as well and pointed out possible directions in further research with emphasis on the application of Software-Defined Networking (SDN) for mitigation purpose. Technically, the refinements focused on the utilization of flexible flow, immediate cache, extended flow values involving DNS attributes, and client-side Response Rate Limiting (RRL) practice. Both flow-based analysis and SDN are expected to play an increasingly major role in today's rapid networks.

INTRODUCTION

High availability plays a vital role in the Information Age. Ensuring availability involves the protection against network threats that could lead to unavailability, such as DDoS (Bhuyan *et al*., 2014). DDoS is a type of cyber-attack that aims to disrupt network services. The attack typically works by flooding a target with bogus network traffic thru various ways to overwhelm its resources and consequently causing cascading failures. An attacker usually controls a group of computers, also known as botnets or zombies, to distribute the attack. The motives may range from hacktivism, vandalism, political agenda or personal vendetta, and the implications may range from a nuisance to tons of money in lost revenue (Bhuyan *et al*., 2014). A reflected DDoS is more detrimental because the attack is more distributed and mostly involves servers (amplifiers) that are capable of sending amplified responses. On March 18 2013 (Anagnostopoulos *et al*., 2013), a cyber-attack caused by DNS amplification has resulted in what many security experts are calling one of the largest reflected DDoS so far. The attack works by distributing spoofed DNS requests to numerous open resolvers. An open resolver is a DNS server that recursively replies to random DNS queries from anyone all over the cyberspace. Due to spoofing, these servers then would assume as if the DNS requests were sent from the intended target and all DNS responses are redirected to that target instead (see Fig. 1). The attack takes advantage of the circumstance that a small DNS query can cause amplified DNS replies, where Amplification Factor (AF) is defined as: response

To Cite This Article: Ahmad Ariff Aizuddin Mohd Atan, Megat Norulazmi Megat Mohamed Noor and Mohd Nazri Ismail., An Overview of DNS Amplification Attack Defense via Flow-Based Analysis and SDN. ***Aust. J. Basic & Appl. Sci.,*** *10(11): 47-57, 2016*

size / request size. Average DNS sessions roughly involve a 40 bytes request and a 512 bytes response. This alone yields an AF of 12.8. In relation, DNS Security Extensions (DNSSEC) attempts to ensure data integrity by adding cryptographic signatures into DNS responses, but it also inadvertently creates larger messages which contribute to the AF of 100 or more (van Rijswijk-Deij *et al.*, 2014). Conventionally, a solid DDoS attack demands an excessive use of botnets. It takes time and effort to set up a botnet, and money to rent one. Thus, attackers are in favor of reflected DDoS as substitutes. Since open resolvers naturally have comparatively high bandwidth (up to 10Gbps) to handle manifold requests, they have no problem pumping out loads of traffic. Technically, there are various types of amplification attacks involving different kinds of protocols, but DNS amplification is the most widely used because of its readily available exploit and ease of execution (see Fig. 2). Cisco (2014) highlighted that DNS amplification attack remains a concern in 2015.

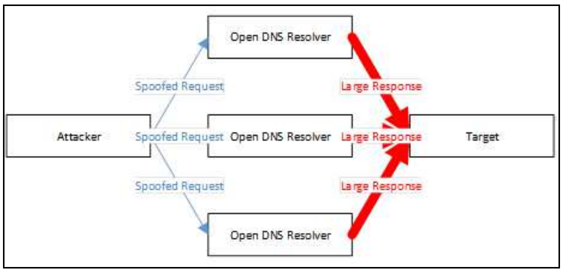


Fig. 1: Theoretical concept of DNS amplification attack.

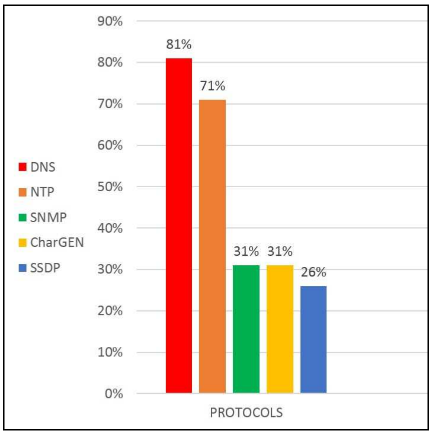


Fig. 2: Protocols used for amplification (Anstee *et al.*, 2014).

Countermeasure:

A. Key Countermeasure:

DNS amplification attack was preliminary reported by Vaughn and Evron (2006), and has gained much attraction ever since. US-CERT (2013) stated that there are two main solutions for DNS amplification attack. First option; ISP should implement packet filtering policies (BCP38) to check the validity of IP addresses (Damas and Neves, 2008). However, BCP38 is demanding to deal with and suffers from lack of global participation since the implementation cost exceeds its benefit. A large number of ISP customers means a large number of separate filters must be created, maintained, and updated, all of which pose the risk of misconfigurations (Wheeler, 2010). Second option; reconfigure preexisting open resolvers or amplifiers (Hudaib and Hudaib, 2014). Open Resolver Project (2015) discovered about 28 million publicly exploitable DNS servers, and to reconfigure each of them would be a very time-consuming procedure as the numbers will only continue to rise since recursions are enabled by default in the deployment of new DNS servers. Likewise, most DNS providers manage their capacities by adding more DNS servers and/or bandwidth as well (Silva, 2014). Furthermore, the impending growth of IoT would worsen the scenario due to the increasing need for scalable DNS services in the IoT architecture (Muncaster, 2015; McMillan, 2015).

B. Conventional Countermeasure:

Conventional countermeasures like Firewall and IPS mostly rely upon predetermined attack patterns or malicious contents (Bauer and Adams, 2012). This however does not apply to protocols like DNS since it transparently involves legitimate traffic. Moreover, these methods cannot act as the first line of defense since they are often deployed deeper in a network hierarchy. They also do not scale well due to their stateful design, where each packet is compared against the state table for deep packet inspection (Sheth and Thakker, 2013). Hence, the approach would not be able to keep up with inspecting all data at high throughput levels. Overprovisioning (Silva, 2014) is another conventional countermeasure to protect against DDoS attacks. It can be as simple as deploying more servers to support flash crowd and absorb attack traffic. Nonetheless, overprovisioning also means more resources to operate and manage the substructure; a big capital outlay. At the end of the day, it is fairly equivalent to creating more targets (Bhuyan *et al*., 2014).

C. Alternative Countermeasure:

Conservatively, both Kambourakis *et al.* (2008) and Sun *et al.* (2008) built a low-cost hardware solution to protect against amplification attacks, but since it is hardware-based, it is unfeasible for updates and aggressive expansion. Vixie and Schryver (2012) described a modus to blunt the impact of amplification by limiting the rate of DNS responses sent by a DNS server. The default configuration is five identical responses per second for a single DNS client/block of network address. Else, the client will be ignored for a default of five seconds. The effectiveness of this method is questionable since the attacker might be able to avoid it by distributing the attack over a large number of DNS servers while staying under the limits of each server. Donnerhacke (2012) advocated a slightly similar way by dampening the rate of DNS queries received on a DNS server. The idea is to collect penalty points per requester, where each point is based on the query type. When the same query is repeated, additional points per repeat count are applied, and if the accumulated points reach a limit, dampening is activated. The downside of it is high chances of misjudgments, where the dampened traffic might be legit. Ye and Ye (2013) presented a detection scheme based on IPTraf tool to differentiate between benign and malicious DNS traffic. Lexis (2013) developed a visualization tool to identify strange activities in DNS queries and responses. All of these measures however were applicable to DNS servers rather than on targeted clients. The same goes for a BCP38-like solution cited by Anagnostopoulos *et al*. (2013), which involves policies to spot forged DNS requests, but mandates wide scale deployment between open resolvers. Another server-centric solution is provided by Dolmans (2013); using a global whitelist containing all DNS resolvers. Though the mitigation seems feasible, it perpetually demands enough DNS resolvers' addresses to be collected. Sattar *et al.* (2013) proposed a solution which works by storing all outgoing queries and incoming responses. Whenever an unmatched reply is received, a threat counter increments until it reaches the risk threshold (administrator-specified) and generates attack alert. The pattern matching is based on TTL value; if a response is not within the given interval of time (e.g. 64 seconds) then the IP address is blocked. The problem with this is its scalability for large networks as it needs to store every DNS requests and replies.

Subsequently, an incentive-driven approach is required and can be achieved through Network Behavior Analysis (NBA) (Nitin *et al*., 2012).

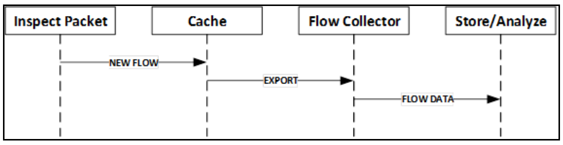


Fig. 3: Standard flow-based operation (Li *et al*., 2013).

NBA is utilized to tackle the issues that are overlooked by customary approaches, such as internal threats or zero-day exploits. Technically, a benchmark of normal network activity must first be established over a period of time. Once certain parameters have been defined as normal, any departure from one or more of it is considered to be suspicious (Li *et al*., 2013; Bhuyan *et al*., 2014). In terms of application, NetFlow and sFlow are widely regarded as ideal tools for flow-based analysis (Miltenburg and Veelenturf, 2015).

Flow-Based Analysis:

A flow is a unidirectional sequence of packets that share the values of ingress interface, source/destination IP address and port number, protocol, and type of service. Other values that may be included are the non-key fields, i.e. durations, packets, bytes, and flows. Fig. 3 shows the principle of a flow-based operation. It typically works by caching incoming packets in the flow cache for aggregation purpose. The aggregated flows are then exported as flow records to a dedicated server (collector) for traffic analysis. Flow-based analysis is not a replacement but rather an addition to the protection supported by aforesaid countermeasures (van Rijswijk-Deij *et al*., 2014).

A. Types of Analysis:

Flow-based analysis (Li *et al*., 2013) can generally be categorized as:

- Network Monitoring: Provide information about routers/switches and is used for troubleshooting

- Application Monitoring: Provide information about application usage over a network and is used for resource allocation
- Host Monitoring: Provide information about user utilization of network and is used for access control/policy violation
- Accounting: Provide information about bandwidth usage over a network and is used for billing
- Network Security: Provide information about changes in network behavior and is used to identify anomalies

B. Analysis – Network Security:

The main advantage of flow-based analysis is its feasibility within high-speed networks since it simply relies on packet headers (Golling *et al*., 2014). While lacking the actual payload, flow-based analysis still provides an adequate amount of timely information for network security purposes. Recent research approaches have concentrated more on the detection/mitigation of network threats that are quantifiable via packet headers, such as port scans, worms, botnets, and DDoS. Li *et al*. (2013) asserted that real-time analysis conservatively demand large storage space for the collector due to constant update of new traffic, but a trade-off can be achieved depending on the selection of flow-based application.

C. Fixed Flow vs. Flexible Flow:

As of today, there are two well-known types of flow-based application; NetFlow (fixed) and sFlow (flexible). NetFlow (Chen *et al*., 2015) is a feature on Cisco routers that collects flows as it enters an interface, while sFlow is a flow-based monitoring technology developed by InMon Corporation (Afaq *et al*., 2015). The main difference between these two versions is the flow values defined in both of them (see Fig. 4). NetFlow is in need of flexibility and extensibility (Patterson, 2013; Hofstede *et al*., 2014). NetFlow records are rather limited to the predefined flow values. In terms of DNS traffic analysis, the limitation is evidenced by the lack of DNS application data. These data consist of, but not limited to, DNS query ID, query type, query name, return code, and number of returned answers. sFlow permits users to add extended flow values that suited the analysis goals (Golling *et al*., 2014). Conversely, extended flow values might unnecessarily increase the size of flow records. However, not all DNS application data are required. As long as the optimally selected values (e.g. DNS query ID) fulfill the detection criterion, it would not excessively increase the size of flow records. Apart from the flow values, NetFlow is highly subjected to cache timeout (Patterson, 2013; Hofstede *et al*., 2014).

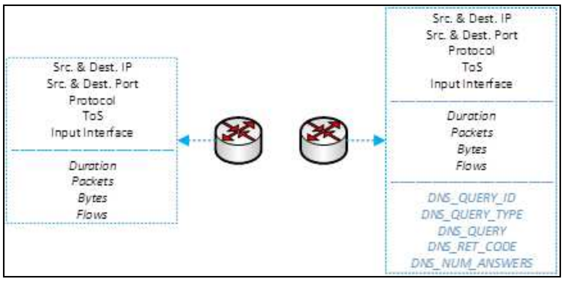


Fig. 4: Fixed-flow vs. Flexible-flow (Afaq *et al*., 2015).

The timeout value ranges from a minimum of 1 minute to a maximum of 60 minutes. Shorter timeout will cause lots of short-lived flows exportations while longer timeout will cause high occupation of cache memory. The cache size is modifiable on most Cisco high-end routers up to 524,288 entries. sFlow permits users to choose between normal and immediate cache (Golling *et al*., 2014). Immediate cache enables flows to be exported instantly based on certain flow values (e.g. port 53). If a flow is parsed and does not meet the criteria, it is stored in the normal cache. Most of the flow-based researches and analysis rely upon NetFlow since it was formerly considered as a de-facto industry standard and has large community support back then (Li *et al*., 2013; Hofstede *et al*., 2014). Nowadays, sFlow is gaining considerable attention due to its association with recent technology like SDN and alternative flow records' options (Afaq *et al*., 2015).

D. Flow Records:

Naccache and Sauveron (2014) gathered real network traffic exported as flow records, and considered 11 candidate attributes that correlate with protocol (e.g. DNS) anomalies. These attributes are TCP/UDP octet count per flow, TCP/UDP packet count per flow, source/destination port variance for TCP/UDP traffic, source IP address variance, and TCP/UDP traffic ratio. Nevertheless, a reduced number of attributes is preferable for resource efficiency as it would also be beneficial for a lightweight detection approach. Comparatively, Sadre *et al*. (2012) and Hoogesteger *et al*. (2014) have previously parameterized the flow attributes that are considerably affected during DDoS attacks. The authors utilized simple theoretical reasoning to label any possible fluctuations in the flow records. The reasoning is then forwarded to simple queuing model that states the performance of flow application. The attributes comprise average flow duration, flow record creations per second, average number of bytes per flow, and average number of packets per flow. However, time-series analysis revealed that smaller computation time (shorter timeout) of the attributes results in too much noise, where the diurnal pattern is vaguely measurable and consist of many peaks that are not distinguishable.

In relation, both Rossow (2014) and Zaalouk *et al*. (2014) are among the current and closely related researches in the detection of DNS amplification attack via flow-based analysis. In details, Rossow (2014) focused on identifying and delivering lists of potential amplifiers by analyzing flow data obtained from a large European ISP. The author hypothesized that a targeted network shall receive huge sums of traffic from DNS servers while there is no request involved. Based on the hypothesis, a pairflow for each client/server pair is created. The amplifiers are defined by the number of received bytes-per-pairflow, and ratio of sent and received bytes. The detection methods depend on fixed flow records and require a high threshold, where the threshold values are subjected to optimization and mainly depends on different types of network traffic profiles (e.g. enterprise-network or campus-network). Zaalouk *et al*. (2014) focused on detecting and mitigating the malicious traffic in DNS amplification attack by analyzing simulated flow data. The author further reviewed about existing approaches and its lack of reactive action or automated mechanism once an attack is detected. The detection logic is based upon total number of packets per flow and average number of bytes per flow. Using fixed flow records, the detection methods also require a high threshold and an optimization, and it is a proof-of-concept since there is no real network traces involved.

In all of these researches, the related analysis is done after a predetermined cache timeout. In other words, the related analysis require at least 5 to 10 minutes of aggregated flows before the traffic can be determined as benign or malicious. Hence, there is a delay in the detection. The main issue with shorter timeout is that not enough flow data is collected, thus resulting in insufficient and unreliable info. The reduction of timeout value would as well raise inaccuracy or false positive rates. Then again, 5 to 10 minutes duration is contradicting the concept of a near real-time environment. Empirical studies (Vitali *et al*., 2012) advised that one minute is a good compromise between reactivity and sensibility. The sooner the flow data is analyzed, the quicker the network threat is detected. In a comprehensive survey of DDoS attacks, Bhuyan *et al*. (2014) discussed that emphasis should be given more to detection speed over accuracy (e.g. up to 90% accuracy for less than 1 minute) in order to facilitate a practical flow-based anomaly mitigation within the DDoS defense mechanism.

Various efforts have been made by many researchers in order to put forward a flow-based solution that would manage and alleviate DDoS attacks (Kreutz *et al*., 2015). However, most of the researches have focused on utilizing forwarding devices and using proprietary hardware appliance as DDoS defense, where the mitigations involved are traffic blocking and rerouting. These solutions also require the implementation of protocol like BGP for traffic diversion and the establishment of tunnel like GRE for traffic forwarding. According to Nayana *et al*. (2015), such intricacies within traditional networking indicate an absence of autonomic properties (e.g. centralized control), hence lacking the support for automatic attack handling. Therefore, a holistic-cum-automated mitigation system is desired, and it is achievable via SDN.

SDN:

SDN is an emerging network architecture that aims to simplify networking by decoupling the control planes from the underlying network devices/data planes (Kreutz *et al*., 2015). In principle, control planes are used to decide the forwarding of packets, and data planes are used to forward those packets based on the decision (path). In SDN, control plane (controller) is realized in a centralized server, while data plane (i.e. router or switch) becomes a simple forwarding device (see Fig. 5). Accordingly, the separation of both planes allow more programmability and flexibility in a sense that a controller can run various applications from different programming languages, and streamline policy implementation or network reconfiguration in any forwarding device despite its vendors or platforms (Afaq *et al*., 2015; Kreutz *et al*., 2015).

SDN generally uses flow-tables (list of flow entries) to forward packets (see Fig. 6). Each entry in the flow-table has match-fields counters to retrieve per-flow, per-table or per-packet statistics, and actions to decide how to process the packets. If there is a match, the packets are processed based on the action associated with that entries. Else, the packets are forwarded to the controllers (Afaq *et al*., 2015). Technically, there are two types of security researches in SDN; securing SDN (analyzing its vulnerabilities) and security-centric SDN (using its features for defensive reasons).

A. SDN Security Features:

SDN provides the following network security features (Afaq *et al*., 2015):

- Network Visibility: SDN controller has a network-wide view that allows it to make security decisions based on the traffic seen in the entire network

- Decoupled Planes: Network devices can focus more on forwarding, hence reducing overhead as no processing decision needs to be made
- Security Applications: No hardware based solution is needed as all logic is executed on the controller in the form of SDN applications
- Integrated Changes: No changes are required for all the hosts in the network as all the changes (if any) are done in the controller
- Traffic Steering: Since SDN controller has a network-wide view, it can selectively steer/drop malicious traffic in an automated manner

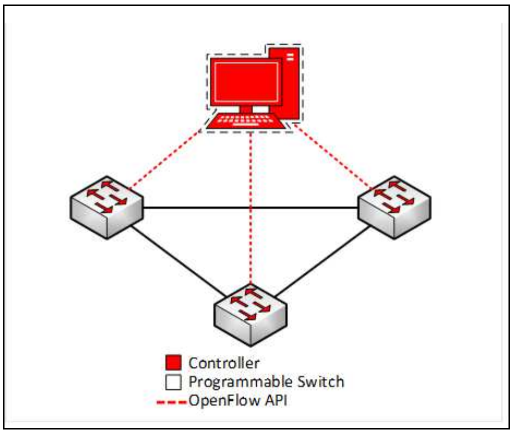


Fig. 5: Standard SDN component (Kreutz *et al*., 2015).

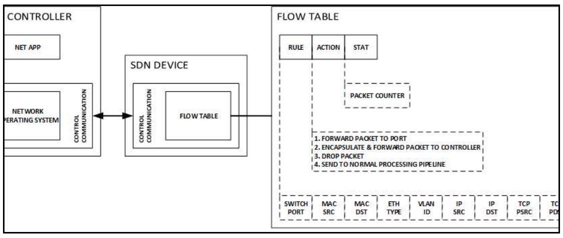


Fig. 6: SDN flow-table (Lara *et al*., 2014).

B. SDN Architecture:

As shown in Fig. 7, a typical SDN architecture consists of SDN applications, Northbound Interface (NI), controllers (control plane), Southbound Interface (SI), and forwarding devices (data plane). NI defines the Application Program Interface (API) that enables the interaction between controllers and SDN applications (e.g. REST), while SI defines the API that permits the communication between controllers and forwarding devices (i.e. OpenFlow). One might confuse OpenFlow to be the same as NetFlow or sFlow for its name, but in practice, they are primarily different. Technically, both NetFlow and sFlow are flow-based monitoring technologies, while OpenFlow is a flow-based configuration technology (Hofstede *et al*., 2014; Lara *et al*., 2014; Afaq *et al*., 2015). Thus, these two technologies are essentially required in order to have a pragmatic SDN architecture.

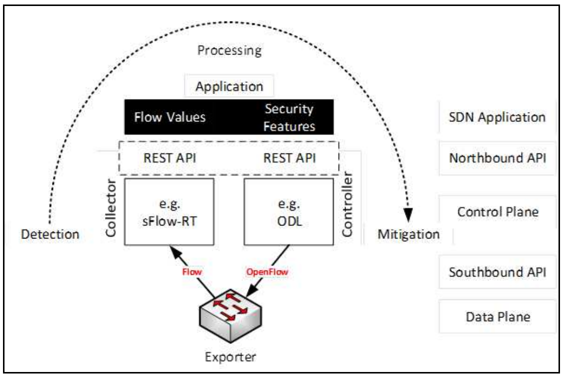


Fig. 7: SDN architecture (Afaq *et al*., 2015).

Table I: Countermeasures for Security Threats in OpenFlow Networks.

Measure	Description
Access Control	Provide strong authentication/authorization mechanism on devices
Event Filtering	Allow/block certain types of event to be handled by special devices
Flow Aggregation	Coarse-grained rules to match multiple flows to prevent information disclosure and DDoS attacks
Forensic Support	Allow reliable storage of traces of network activities to find the root causes of different problems
Intrusion Tolerance	Enable control platforms to maintain correct operation despite intrusions
Shorter Timeouts	Useful to reduce the impact of an attack that diverts traffic

As previously mentioned, SDN applications are usually unified with the controllers. In terms of network security, the applications comprise both detection and mitigation logic due to the flow-based nature of SDN and its central point of knowledge (Kreutz *et al*., 2015; Afaq *et al*., 2015). Table I summarizes a number of countermeasures that can be applied to different elements of SDN architecture, depending on the OpenFlow versions. Conventionally, some of these countermeasures are already applicable to middleboxes (e.g. Intrusion Detection System) since they are commonly special purpose devices and more robust (Bauer and Adams, 2012). However, middleboxes deployment are also known for their challenges and criticism due to poor interaction with higher layer protocols, apart from additional costs and complexities in network management (Sheth and Thakker, 2013). Though SDN is preferred, further research remains open for discussion.

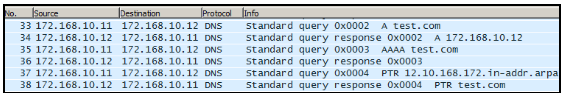
Discussion:

Li *et al*. (2013) explored the primary flow-based detection of network threats via Top N analysis. Top N refers to a set of statistics distributed by flow-enabled router for detecting heavy-hitters or big talkers. Basic Top N data itself is not sufficient to comprehend complex network security situations. Then again, Li *et al*. (2013) pointed out that there is no silver bullet in network security. It is indicated that Top N analysis would evidently contribute to the countermeasures if it is combined with other perspectives of flow-based anomaly detections specified in Section III-D.

As discussed by (Bhuyan *et al*., 2014), a faster detection scheme adversely affects accuracy. An offline classifier normally ensures better performance compared to a real-time detection, but accurately detecting all attacks after interrupting the services are impractical. One cannot just store captured packets and analyze them later, clearly increasing the lag between the start of an attack and its subsequent detection. All phases of detection need to run in a parallel manner. As mentioned in Section III-B and Section III-C, most related researches rely upon fixed flow records. Although cache timeout influences detection time, it is not the only factor of delays. This is because fixed flow itself is restricted to give enough information for real-time detections. On top of delays, flow cache has explicit size limits. In the event of an attack, the amount of incoming traffic would rapidly intensify and lead to cache exhaustion. Existing flows will not be exported until the timeout is reached, and new flows will not be cached until the cache is cleared. This is considered as intrusive if the new flows consist of normal traffic (Hofstede *et al*., 2013). Therefore, further research should leverage the application of flexible flow instead.

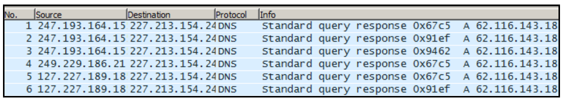
A. Prospective Detection:

The importance of information in flexible flow was initially a secondary discovery, but now it has been a primary finding for detecting/mitigating network threats (Chen *et al*., 2015; Golling *et al*., 2014). In the context of cache timeout, immediate cache helps in achieving a near-real time environment by expediting the exportation process, and averting memory overload. In the context of flow values, the analysis works by statistically observing the simulated data, and heuristically choosing optimal DNS information that would fluctuate from the standardized frequencies of flows.



```
No.  Source         Destination    Protocol  Info
  33 172.168.10.11  172.168.10.12  DNS       Standard query 0x0002  A test.com
  34 172.168.10.12  172.168.10.11  DNS       Standard query response 0x0002  A 172.168.10.12
  35 172.168.10.11  172.168.10.12  DNS       Standard query 0x0003  AAAA test.com
  36 172.168.10.12  172.168.10.11  DNS       Standard query response 0x0003
  37 172.168.10.11  172.168.10.12  DNS       Standard query 0x0004  PTR 12.10.168.172.in-addr.arpa
  38 172.168.10.12  172.168.10.11  DNS       Standard query response 0x0004  PTR test.com
```

Fig. 8: Benign DNS packet capture via Wireshark.



```
No.  Source          Destination     Protocol  Info
   1 247.193.164.15  227.213.154.24  DNS       Standard query response 0x67c5  A 62.116.143.18
   2 247.193.164.15  227.213.154.24  DNS       Standard query response 0x91ef  A 62.116.143.18
   3 247.193.164.15  227.213.154.24  DNS       Standard query response 0x9462  A 62.116.143.18
   4 249.229.186.21  227.213.154.24  DNS       Standard query response 0x67c5  A 62.116.143.18
   5 127.227.189.18  227.213.154.24  DNS       Standard query response 0x67c5  A 62.116.143.18
   6 127.227.189.18  227.213.154.24  DNS       Standard query response 0x91ef  A 62.116.143.18
```

Fig. 9: Malicious DNS packet capture via Wireshark.

To do so, a set of benign dataset must be reiterated and served as a benchmark of normal network activity (Santanna *et al*., 2015). After a simulation of DNS amplification attack, both benign and malicious traffic are then compared/analyzed for any departure from normal parameters. It is best to keep the number of chosen DNS attribute to a minimum since it would also be beneficial for a lightweight detection approach. Out of the five metrics (DNS query ID, query type, query name, return code, and number of returned answers), the first appears suitable to fulfill the requirement of being lightweight as only a single counter is needed. DNS Query ID is mainly used for tracking queries and responses to that particular queries, and the value ranges from 0 through 65535 (Eastlake, 2013). As conjectured by (Rossow, 2014), a targeted network shall receive huge sums of traffic from DNS servers while there is no or minimal request involved. Conventionally, this can be computed through the ratio between received and sent bytes. If the ratio is larger than a threshold or equals to 0, then it will be marked as suspicious. However, the computational process can only be done after a predetermined cache timeout as shorter period will lead to deficient outcomes (false positives). In terms of Query ID, benign DNS traffic should have equivalent Query ID for both queries and replies. Else, there is an attempt of DNS amplification attack. Compared to conventional approaches, Query ID does not require excessive computation of flow values. Fig. 8 shows an example of benign DNS traffic captured from simulated data, where for every standard query (e.g. 0x0002 A test.com), there is a standard query response that matches the transaction ID, in hex value, (e.g. 0x0002 A 172.168.10.12). While Fig. 9 shows a snippet of malicious DNS traffic captured from a virtualized target, where multiple standard query responses with multiple occurrences, do not have its corresponding standard query.

B. Prospective Mitigation:

Traffic dropping or blocking is not a definitive solution when handling attacks involving legitimate network services like DNS protocol. This is because attackers generally hide behind, for instance, recursive DNS servers or open resolvers, instead of generating the attack themselves.

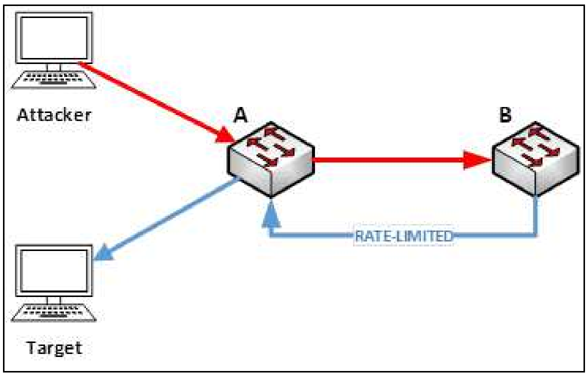


Fig. 10: Ping-pong application (Zaalouk *et al*., 2014).

These servers still serve for normal DNS operation, where there are possibilities that a targeted network and the servers having common DNS sessions (Rossow, 2014). So this is also considered as intrusive if traffic from the servers are plainly blocked; preventing normal DNS operation. Another potential rule action for DDoS mitigations is to rate-limit the response received at a targeted network (Bhuyan *et al*., 2014). Rate-limit has in fact been practiced by authoritative name servers where it also poses significant drawbacks (Vixie and Schryver, 2012). While it is not a definitive solution on server-side, the option seems to be a reasonable approach on client-side (targeted network) since it reduce the amplification effect while preserving normal DNS operation. One way to perform a rate-limit is by using the Ping-Pong application (Zaalouk *et al*., 2014). As stated in Section III-C, Zaalouk *et al*. (2014) is one of the current and closely related researches in the detection of DNS amplification attack via flow-based analysis. Relatively, it is also the only research that is currently using SDN features for mitigating this type of attack (security-centric SDN). As stated in Section IV-B, the type of countermeasures listed on Table I are highly dependable on the version of OpenFlow. Zaalouk *et al*. (2014) used OpenFlow v1.0 since it is supported on major SDN controllers (e.g. OpenDaylight, Floodlight, POX) compared to other versions (up to 1.5). However, v1.0 does not include rate-limit option so an alternative application (Ping-Pong) was implemented instead. Fig. 10 shows the topology where one of the links between network device A and B is configured to have a lower capacity than it usually has, hence rate-limiting the link. Based on Fig. 16, when an attack is detected, device A forwards the malicious traffic to device B. Device B, which is instructed by a controller, will resend the malicious traffic back to device A over the rate-limited link. Once received, device A forwards the traffic normally to its intended destination. OpenFlow v1.3 provides meter-tables which can be used to implement various QoS operations, such as rate-limiting (Lara *et al*., 2014). Additionally, malicious traffic could also be marked with special labels, and forwarded by OpenFlow to a scrubber (e.g. Snort) for further analysis (Lara *et al*., 2014; Kreutz *et al*., 2015).

Conclusion:

Nowadays, where there is a need for network modernization, there is a need for innovative security. In conjunction with IPv6 and IoT, flow-based analysis and SDN are expected to promote a paradigm shift in today's rapid networks. This paper evaluated related studies that foster further research to improve the interval and accuracy of DNS amplification attack detection via flow-based analysis, and the automation of attack mitigation via SDN. Accordingly, the robustness of detection/mitigation can be justified with real network traces that vary in total volume sent, attack sources, queries, AF, speed, and normal to abnormal ratio.

REFERENCES

Afaq, M., S. Rehman and W.C. Song. 2015. Large Flows Detection, Marking, and Mitigation based on sFlow Standard in SDN. Journal of Korea Multimedia Society, 18 (2): 189-198. DOI:10.9717/kmms.2015.18.2.189.

Anagnostopoulos, M., G. Kambourakis, P. Kopanos, G. Louloudakis and S. Gritzalis, 2013. DNS Amplification Attack Revisited. Computer Security, 39 (0167-4048): 475-485. DOI:10.1016/j.cose.2013.10.001.

Anstee, D., J. Escobar, C.F. Chui and G. Sockrider, 2014. Worldwide Infrastructure Security Report.Arbor Networks, Massachusetts, USA.http://www.arbornetworks.com/resources/infrastructure-security-report (Accessed on March 17, 2015)

Bauer, E., R. Adams, 2012. Denial of Service Attacks. In: Reliability and Availability of Cloud Computing, Wiley, Hoboken, NJ, pp: 159-160. ISBN: 9781118177013.

Bhuyan, M. H., H.J. Kashyap, D.K. Bhattacharyya and J.K. Kalita, 2014. Detecting Distributed Denial of Service Attacks: Methods, Tools and Future Directions. The Computer Journal, 57(4): 537-556. DOI:10.1093/comjnl/bxt031.

Chen, Z., Y. Wen, J. Cao, W. Zheng, J. Chang, Y. Wu, G. Ma, M. Hakmaoui and G. Peng, 2015. A Survey of Bitmap Index Compression Algorithms for Big Data. Tsinghua Science and Technology, 20(1): 100-115. DOI:10.1109/TST.2015.7040519.

Cisco, 2014. Annual Security Report.Cisco Public Information, California, USA.http://www.cisco.com/web/offers/lp/2014-annual-security-report/index.html (Accessed on March 17, 2015)

Damas, J., F. Neves, 2008. Preventing Use of Recursive Nameservers in Reflector Attacks.Internet Engineering Task Force, California, USA.https://www.ietf.org/rfc/rfc5358.txt (Accessed on March 18, 2015)

Dolmans, R., 2013.Preventing DNS Amplification Attacks using white- and greylisting. Unpublished dissertation in partial fulfillment of the requirements for the degree of Master of System and Network Engineering, University of Amsterdam, Amsterdam, Netherlands

Donnerhacke, L., 2012. DNS Dampening.Jena, Thuringia, German.http://lutz.donnerhacke.de/eng/Blog/DNS-Dampening (Accessed on March 19, 2015)

Eastlake, D., 2013. Domain Name System (DNS) IANA Considerations.Internet Engineering Task Force, California, USA.https://tools.ietf.org/html/rfc6895 (Accessed on March 23, 2015)

Golling M., R. Koch and R. Hofstede, 2014. Towards multi-layered intrusion detection in high-speed networks. 2014 6th International Conference on Cyber Conflict, June 3-6, IEEE Xplore Press, Estonia, pp: 191-206. DOI: 10.1109/CYCON.2014.6916403.

Hofstede, R., P. Celeda, B. Trammell, I. Drago, R. Sadre, A. Sperotto and A. Pras, 2014. Flow Monitoring Explained: From Packet Capture to Data Analysis With NetFlow and IPFIX. Communications Surveys & Tutorials, 16 (2037-2064): 1553-877X. DOI:10.1109/COMST.2014.2321898.

Hofstede, R., V. Bartos, A. Sperotto and A. Pras, 2013. Towards Real-Time Intrusion Detection for NetFlow and IPFIX. 9th International Conference on Network and Service Management, Oct. 14-18, IEEE Computer Society, Switzerland, pp: 227-234. URL: http://doc.utwente.nl/87853/.

Hoogesteger, M., O. Schmidt, Ricardo de, A. Sperotto and A. Pras, 2014. ReFlow: Reports on Internet Traffic. TERENA Networking Conference, May 19-22, TERENA, Netherlands, URL: http://doc.utwente.nl/91455/.

Hudaib, A.A.Z. and E.A.Z. Hudaib, 2014. DNS Advanced Attacks and Analysis. Computer Science and Security, 8 (63-74): 1985-1553. URL:http://www.cscjournals.org/manuscript/Journals/IJCSS/volume8/Issue2/IJCSS-905.pdf.

Kambourakis, G., T. Moschos, D. Geneiatakis and S. Gritzalis, 2008. Detecting DNS Amplification Attacks. Computer Science, 5141 (185-196): 0302-9743. DOI:10.1007/978-3-540-89173-4_16.

Kreutz D., F.M.V. Ramos, P.E. Verissimo, C.E. Rothenberg, S. Azodolmolky and S. Uhlig, 2015. Software-Defined Networking: A Comprehensive Survey. Proceedings of the IEEE, 103 (1): 14-76. DOI:10.1109/JPROC.2014.2371999.

Lara A., A. Kolasani, and B. Ramamurthy, 2014. Network Innovation using OpenFlow: A Survey. Communications Surveys & Tutorials, 16(1): 493-512. DOI:10.1109/SURV.2013.081313.00105.

Lexis, P., 2013.Identifying Patterns in DNS Traffic - Using Visual Analytics to Discover DNS Abuse. Unpublished dissertation in partial fulfillment of the requirements for the degree of Master of System and Network Engineering, University of Amsterdam, Amsterdam, Netherlands

Li, B., J. Springer, G. Bebis and M.H. Gunes. 2013. A survey of network flow applications. Computer Applications, 36(567-581): 1084-8045. DOI:10.1016/j.jnca.2012.12.020.

McMillan, L., 2015. The DNS of Things.SYS-CON Media, New Jersey, USA.http://lizmcmillan.sys-con.com/node/3241349 (Accessed on March 27, 2015)

Miltenburg, W. and K. Veelenturf, 2015. Preventing Common Attacks on Critical Infrastructure. Unpublished dissertation in partial fulfillment of the requirements for the degree of Master of System and Network Engineering, University of Amsterdam, Amsterdam, Netherlands

Muncaster, M., 2015. Home Routers and IoT Devices Set to Drive DNS DDoS Attacks.Infosecurity Magazine, London, England.http://www.infosecurity-magazine.com/news/home-routers-iot-devices-drive-dns/ (Accessed on March 27, 2015)

Naccache, D. and D. Sauveron, 2014. Information Security Theory and Practice. Securing the Internet of Things. 8th IFIP WG 11.2 International Workshop, June 30 - July 2, Springer Science+Business Media, Germany, pp: 249-256. DOI: 10.1007/978-3-662-43826-8.

Nayana Y., J. Gopinath and L. Girish, 2015. DDoS Mitigation using Software Defined Network. International Journal of Engineering Trends and Technology, 24 (5): 258-564. DOI:10.14445/22315381/IJETT-V24P246.

Nitin, T., S. Rajdeep Singh and P.G. Singh. 2012. Intrusion Detection and Prevention System (IDPS) Technology- Network Behavior Analysis System (NBAS). Engineering Sciences, 1 (51-56). URL:http://www.isca.in/IJES/Archive/v1i1/9.ISCA-JEngS-2012-032.pdf.

Open Resolver Project, 2015.http://openresolverproject.org/ (Accessed on February 16, 2015)

Patterson, M., 2013. NetFlow v5 Vs. NetFlow v9.Extreme Networks, California, USA.http://www.extremenetworks.com/netflow-v5-vs-netflow-v9 (Accessed on March 26, 2015)

Rossow, C., 2014. Amplification Hell: Revisiting Network Protocols for DDoS Abuse. Proceedings of the 2014 Network and Distributed System Security, Feb. 23-26, Internet Society, USA, pp: 1-15. DOI: 10.14722/ndss.2014.23233.

Sadre, R., A. Sperotto and A., 2012. The effects of DDoS attacks on flow monitoring applications. Network Operations and Management Symposium, Apr. 16-20, IEEE Xplore Press, USA, pp: 269-277. DOI: 10.1109/NOMS.2012.6211908.

Santanna, J.J., R. van Rijswijk-Deij, A. Sperotto, R. Hofstede and M. Wierbosch, 2015. Booters - An analysis of DDoS-as-a-Service Attacks. Proceedings of 14th IFIP/IEEE Symposium on Integrated Network and Service Management, May 11-15, IEEE Xplore Press, Canada. URL: http://im2015.ieee-im.org/content/technical-sessions-0.

Sattar, U., T. Naqash, M.R. Zafar, K. Razzaq and F. bin Ubaid, 2013. Secure DNS from amplification attack by using modified bloom filters. Digital Information Management, Sept. 10-12, IEEE Xplore Press, Pakistan, pp: 20-23. DOI: 10.1109/ICDIM.2013.6694018.

Sheth, C., and R. Thakker. 2013. Performance Evaluation and Comparison of Network Firewalls under DDoS Attack. Computer Network and Information Security, 5 (60-67): 2074-9090. DOI:10.5815/ijcnis.2013.12.08.

Silva, P., 2014. Five Key Issues for the DNS of Things.Cloud Expo, California, USA.http://cloudexpo2014west.sys-con.com/?q=event/session/2573 (Accessed on March 27, 2015)

Sun, C., B. Liu and L. Shi, 2008. Efficient and Low-Cost Hardware Defense Against DNS Amplification Attacks. Global Telecommunications Conference, Nov. 30 - Dec. 4, IEEE Xplore Press, USA, pp: 1-5. DOI: 10.1109/GLOCOM.2008.ECP.397.

US-CERT, 2013.Alert (TA13-088A).DNS Amplification Attacks.United States Computer Emergency Readiness Team.https://www.us-cert.gov/ncas/alerts/TA13-088A (Accessed on February 17, 2015)

Van Rijswijk-Deij, R., A. Sperotto and A. Pras, 2014. DNSSEC and Its Potential for DDoS Attacks - A Comprehensive Measurement Study. Proceedings of the 14th ACM Internet Measurement Conference, Nov. 5-7, ACM, USA, pp: 449-460. DOI: 10.1145/2663716.2663731.

Vaughn, R. and G. Evron, 2006. DNS Amplification Attacks.http://crt.io/DNS-Amplification-Attacks.pdf (Accessed on August 31, 2014)

Vitali, D., A. Villani, A. Spognardi, R. Battistoni and L.V. Mancini, 2012. DDoS Detection with Information Theory Metrics and Netflows - A Real Case. Proceedings of the International Conference on Security and Cryptography, July 24-27, SCITEPRESS, Canada, pp: 172-181. DOI: 10.5220/0004064501720181.

Vixie, P. and V. Schryver, 2012. DNS Response Rate Limiting.Internet Systems Consortium, California, USA.http://ss.vix.su/~vixie/isc-tn-2012-1.txt (Accessed on March 17, 2015)

Vizváry, M. and J. Vykopal, 2014. Future of DDoS Attacks Mitigation in Software Defined Networks. Lecture Notes in Computer Science, 8508: 123-127. DOI:10.1007/978-3-662-43862-6_15.

Wheeler, D. A., 2010. Planning for the Future of Cyber Attack Attribution.Institute for Defense Analyses, Virginia, USA.http://archives.democrats.science.house.gov/Media/file/Commdocs/hearings/2010/Tech/15jul/Wheeler_Testimony.pdf (Accessed on March 20, 2015)

Ye, X. and Y. Ye, 2013. A Practical Mechanism to Counteract DNS Amplification DDoS Attacks. Computational Information Systems, 9 (265–272): 1553–9105. URL:http://www.jofcis.com/publishedpapers/2013_9_1_265_272.pdf.

Zaalouk, A., R. Khondoker, R. Marx and K, Bayarou, 2014. OrchSec: An Orchestrator-Based Architecture For Enhancing Network-Security Using Network Monitoring And SDN Control Functions. Network Operations and Management Symposium, May 5-9, IEEE Xplore Press, Poland pp: 1-9. DOI: 10.1109/NOMS.2014.6838409.