Computer Communications 56 (2015) 60–73

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Mitigating congestion based DoS attacks with an enhanced AQM technique

Harkeerat Bedi [a,*], Sankardas Roy [b], Sajjan Shiva [a]

[a] Dept. of Computer Science, University of Memphis, Memphis, TN 38152, United States
[b] Dept. of Computing and Information Sciences, Kansas State University, Manhattan, KS 66506, United States

ARTICLE INFO

Article history:
Received 8 August 2013
Received in revised form 5 July 2014
Accepted 10 September 2014
Available online 19 September 2014

Keywords:
Congestion control
Active queue management
Denial of service
Flow protection

ABSTRACT

Denial of Service (DoS) attacks are currently one of the biggest risks any organization connected to the Internet can face. Hence, the congestion handling techniques at the edge router(s), such as Active Queue Management (AQM) schemes must take into account such attacks. Ideally, an AQM scheme should (a) ensure that each network flow gets its fair share of bandwidth, and (b) identify attack flows so that corrective actions (e.g. drop flooding traffic) can be explicitly taken against them to further mitigate the DoS attacks. This paper presents a proof-of-concept work on devising such an AQM scheme, which we name Deterministic Fair Sharing (DFS). Most of the existing AQM schemes do not achieve the above goals or have significant room for improvement. DFS uses the concept of weighted fair share (*wfs*) that allows it to dynamically self-adjust the router buffer usage based on the current level of congestion, while aiding in identifying malicious flows. By using multiple data structures (a comprehensive repository and a cache) for keeping state of legitimate and malicious flows, DFS is able to optimize its runtime performance (e.g. higher bandwidth flows being handled by the cache). We demonstrate the performance advantage of DFS via extensive simulation while comparing against other existing AQM techniques.

1. Introduction

As widely evidenced, Denial of Service (DoS: regular as well as distributed) is one of the most prominent attack mechanisms on the Internet. In a recent study conducted by Radware and Ponemon Institute that consisted of surveying 705 IT practitioners, it was observed that 65% of the represented organizations suffered from three DoS attacks on average in 2012 [1]. Their average downtime lasted 54 min, resulting in an estimated cost of $22 K per minute, including the loss in revenue, traffic and end user productivity [1].

As computer hardware becomes cheaper and social networking becomes more accessible through the cyberspace, organization and execution of such distributed attacks become significantly easier. For example, botnets capable of performing DoS attacks of throughput ranging from 10–100 Gbps can be rented on the Internet for $200 per 24 h [2]. Moreover, the bandwidth used by these attacks is constantly growing. In 2014, a DDoS attack approximating 400 Gbps was observed by CloudFlare [3].

In general, DoS attacks can be classified into two categories depending on the layer of the OSI model they target: infrastructure-based attacks and application-based attacks. The former targets the layers 3 and 4 (i.e. network and transport layers) and the latter targets the application layer. Infrastructure-based attacks include SYN floods, UDP floods, ICMP floods, and IGMP floods, while application-based attacks include HTTP/SSL GET and POST floods, and NTP floods. Quarterly reports by Prolexic for the last several years show that DDoS attacks on the network infrastructure (Layer 3 and 4) far surpassed those that occurred on the application layer [4–6]. During the first quarter of 2014, over 87% of DDoS attacks were focused on the network infrastructure with UDP floods being one of the most popular attacks [6].

Congestion based attacks still dominate the denial of service landscape. Arbor Networks report that 61% of DDoS attacks observed by their survey respondents in 2014 were congestion based (the remaining included state-exhaustion and application layer attacks) [7]. Academia and the Industry have performed considerable amount of work towards understanding and mitigating network congestion based DoS attacks [8,9]. Active Queue Management (AQM) techniques are one of the most prominent approaches used for this purpose. Major network equipment providers including Cisco [10], Juniper [11] and Huawei [12] offer built-in support for several of them.

In this work, we design a novel congestion identification and mitigation technique that works at the infrastructure level. It aims

* Corresponding author.
E-mail addresses: hsbedi@memphis.edu (H. Bedi), sroy@ksu.edu (S. Roy), sshiva@memphis.edu (S. Shiva).

http://dx.doi.org/10.1016/j.comcom.2014.09.002
0140-3664/

to (a) ensure that each network flow gets its fair share of bandwidth, and (b) identify attack flows so that corrective actions (e.g. drop flooding traffic) can be explicitly taken against them. In particular, we develop an AQM technique called Deterministic Fair Sharing (DFS) that maintains per-flow state such that DoS attack traffic can be precisely identified and effectively mitigated while ensuring fairness.

Internet traffic can be broadly categorized into two major categories. The first category consists of flows that are inherently responsive in nature. That is, when they observe congestion due to packet loss or receive Explicit Congestion Notification (ECN) [13] marked packets, they reduce their sending rate. Examples of these include flows that use TCP as their transport protocol. They are known as *responsive* flows. The second category consists of flows that do not respond to congestion notifications. That is, they do not change their sending rates when they observe a packet loss or an ECN marked packet. Examples of these include flows that use UDP as their transport protocol. They are known as *unresponsive* flows.

1.1. Attacks

DoS attack flows can be defined as: (i) the set of unresponsive flows that use more than their fair share of the bandwidth; (ii) the set of responsive flows that do respond to congestion notifications, but not in a fair manner. Examples include flows using a hacked version of the TCP protocol with the intent to make the flows behave selfishly (or unresponsively); and (iii) the set of responsive flows that originate from a single client where their cumulative share is more than the fair share per client. In this case, the attacker generates a number of parallel fair TCP connections to the target server with the intent to use a major portion of the bandwidth as a whole. Examples include download accelerator tools that create multiple parallel connections to a target server requesting for the same file (different parts) in order to increase the client download speed.

1.2. Main idea

Prior research at large focused on using heuristic and probabilistic measures for creating AQM techniques. Choosing such measures give the benefit of low operational overhead, at the expense of fairness. DFS uses a deterministic approach to address router-based congestion and therefore is able to provide higher fairness. By using multiple data structures, DFS is able to reduce its operational overhead.

DFS keeps track of incoming traffic on a per-flow basis. By performing such granular analysis, it is able to provide a higher degree of fairness to well-behaved flows and accurately identify and throttle misbehaving flows by either dropping their packets or marking them using ECN (see Fig. 1).

To reduce its operational expense, DFS uses two kinds of data structures to manage per-flow information. Most of the flows, including all responsive flows (e.g. TCP flows), are stored in a comprehensive repository. In this paper we use an in-memory B-tree data structure for this purpose. B-tree is chosen since it can have a large fan-out while bounding its height. This in turn helps in reducing the maximum time required in retrieving or updating a stored flow. Flows marked as unfair (e.g. high bandwidth UDP flows) tend to require a higher frequency of updates and are stored separately in an array of fixed size that acts as a cache, for faster access. It should be noted that the fairness, attack identification and mitigation capabilities provided by DFS are not tied with the data structures it uses. B-tree is just one of many data structures that can be used for storing flow states.

Motivation: This work explores the feasibility of the following objectives:

- The potential of handling per-flow processing in modern routers. Per-flow processing techniques can provide unique benefits such as guaranteed bandwidth for legitimate flows and zero miss-classification of such flows as malicious. It can also ensure accurate identification of DoS attack traffic so that corrective actions can be taken explicitly against them [14]. This is necessary since such attacks are primarily performed with malicious intent and they most often lead to tangible economic damage to their victims [1]. Moreover, newer DoS defense techniques should not only focus on attack mitigation but also on explicit identification of malicious flows (e.g. [15]).
- Use of efficient data structures and approaches for storing state or managing lookup tables [16–18].

Note that a shorter and preliminary version of this work has been published in [19].

2. Related work

AQM techniques can be broadly classified in two categories based on the type of traffic they can handle. The first category aims to provide fairness when the incoming traffic consists of only responsive flows (e.g. TCP flows). Typical techniques include RED [20], BLUE [21], and AVQ [22]. The second category aims to provide fairness when the incoming traffic consists of both responsive and unresponsive flows (e.g. TCP and UDP flows). Well known techniques include CHOKe [23], SFB [24], RED-PD [25], and FRED [26].

Dischinger et al. [27] studied the deployment of RED in residential broadband networks. Their study consisted of 1894 broadband hosts from 11 cable and DSL providers in North America and Europe. They observed that 26.2% of the DSL hosts demonstrated a RED-style drop policy on their upstream queues. Moreover, the three providers owned by AT&T (i.e., Ameritech, BellSouth, and PacBell) exhibited deployment rates ranging from 50.3% to 60.5%.

Random Early Detection (RED) estimates the level of congestion in the router's buffer and drops packets accordingly by maintaining an exponentially weighted moving average (EWMA) of the queue length. One of the limitations of RED is that it requires significant parameter tuning to obtain optimal results. Several techniques have been proposed to address this limitation. Adaptive RED (ARED) [28] and its revisions [29,30] extend the effectiveness of RED by allowing the RED parameters to be dynamically self-adjusted based on traffic load. BLUE [21] uses link utilization and packet loss information to handle buffer congestion instead of monitoring the queue length. Another limitation of RED includes its incapability to provide fairness against unresponsive flows. Various AQM techniques have been proposed that are based on (or function with) RED and aim to address this limitation. Examples include FRED, CHOKe, xCHOKe [31], RECHOKe [32], StoRED [33], RRED [34], and RED-PD. Zhang et al. propose CPR (Congestion Participation Rate) [15], a metric and a congestion control technique that can be deployed in routers to identify and mitigate low-rate distributed DoS (LDDoS) attacks. It requires RED as part of its operation.

Random Exponential Marking (REM) [35] measures congestion by a quantity defined as *price*, instead of performance measures like loss, delay or queue length. AVQ uses a virtual queue that simulates the router buffer. If the virtual queue overflows, then the incoming packet either ECN-marked or discarded. CHOKe is a stateless technique that also tries to handle unresponsive flows. For each incoming packet at a congested router, a random packet is chosen from the router queue and if they both belong to the