Scalable Rule Management for Data Centers

Aris Paphitis
CUT

Summary Of

Summary of the article. Keypoints, contributions and comments.

1 Background

Cloud operators increasingly need more and more fine grained rules to better control individual network flows for various traffic management policies. Many novel traffic policies have been proposed by recent research, to improve network utilization, application performance, fairness and cloud security among tenants in multi-tenant data centers.

Tenants in data centers like Amazon or whose servers run software from VMware, place their rules at the servers that source traffic. However, multiple tenants at a server may install too many rules at the same server causing unpredictable failures. Given the scale of today's data centers, the total number of rules can be hundreds of thousands or even millions. In perspective, rule processing of future data centers can hit CPU or memory resource constraints at servers and switches.

2 Motivation

In this paper, the authors argue that future data centers will require automated rule management in order to ensure rule placement that respects resource constraints, minimizes traffic overhead, and automatically adapts to dynamics. They propose vCRIB, a virtual Cloud Rule Information Base, which provides the abstraction of a rule repository, and automatically manages rule placement without intervention by operator. To achieve this they propose offloading rule processing to other devices in the data center.

3 Challenges

The authors acknowledge the need for many fine-grained rules to achieve different management policies; typical examples are data centers that provide computing as a service. They observe that a simple policy can result in a large number of fine grained rules, especially when operators wish to control individual VMs and flows. Additionally, in a data center where multiple concurrent policies might co-exist, rules may have dependencies between them, so may require carefully designed offloading. vCRIB is focused on flow-based rules (drop, rate limit, count) that can be processed either at servers (hypervisors) or programmable network switches. These rules are usually based on IP addresses, MAC addresses, ports, VLAN tags etc. While software-based hypervisors at servers can support complex rules and actions, they may require committing an entire core at each server in the data center. Some hypervisors may offload rule processing to the NIC, but it can only handle limited number of rules due to memory constraints.

Rule management is further complicated by heterogeneity in hardware equipment and the highly dynamic environment with policy changes existing in today's data centers.

The central challenge of vCRIB is to design a collection of algorithms that manages to keep the traffic overhead induced by rule offloading low, while respecting the resource constraints.

4 Contributions

To address these challenges vCRIB provides an abstraction of a centralized repository of rules for the cloud. Tenants and operators simply install rules in this repository and vCRIB uses network state and traffic information to proactively place rules in hypervisors.

4.1 System Design

vCRIB partitions the rule space to break dependencies among rules. Each partition contains rules that can be co-located. Rules that span multiple partitions are replicated rather than split; this reduces rule inflation. In vCRIB this is called `rule partitioning with replication` and it allows a flexible and efficient placement of rules at both hypervisors and switches. Different types of rules may be best placed at different places. vCRIB uses `per-source partitions` such that within each partition, all rules have the same VM as the source. This way only a single rule is required to redirect traffic when that partition is offloaded. Similarities in co-located rules are treated with care not to double resource usage. vCRIB uses resource usage functions that deal with different constraints (CPU/memory) in a uniform way, accommodating device heterogeneity. The process of rule offloading is split in two steps: a novel bin-packing heuristic for `resource-aware partition placement` which leverages similarity and respects resources and a fast `online traffic-aware refinement` algorithm which migrates partitions to reduce traffic overhead. The split enables quick adaptation to small scale dynamics. Finally, the resource-aware placement of rules is achieved through an heuristic algorithm called `First Fit Decreasing Similarity` and is optimized via a `traffic-aware refinement` to reduce traffic overhead while still maintaining feasibility, in the sense that it respects resource constraints.

5 Evaluation

vCRIB's ability to minimize traffic overhead given resource constraints and rule replacement were studied through simulations on a large fat-tree topology. To calibrate vCRIB's performance it was compared against `SourcePlacement`, which stores the rules at the source hypervisor. It has been demonstrated that when Source-Placement is infeasible, vCRIB can successfully fir all the rules in the servers by leveraging similarities of partitions and balancing the rules. vCRIB is able to find placements with low traffic overhead. Additionally it can optimize rule placement when CPU allocated for rule processing is constrained, effectively managing heterogeneous resource constraints.

Resource usage and traffic spatial distribution vCRIB is shown to be effective in leveraging on-path and nearby devices and most traffic overhead introduced is within the rack. Finally it is demonstrated that vCRIB reacts satisfactory to cloud dynamics, such as VM migrations, and maintains the traffic levels low.

Notes

[1] Remember to use endnotes, not footnotes!