IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 7, Issue 10, October 2018

A Survey on Adaptive Wildcard Rule Cache Management with Cache Replacement Algorithms for Software - Defined Networks

Kusekar Shrutika Ajaykumar[1]**, Prof. H.A.Hingoliwala**[2]

Department of Computer Engineering, Savitribai Phule University of Pune,

Jayawantrao Sawant College of Engineering, Hadapsar, Pune[1,2]

Abstract: Software Defined Networking empowers adaptable stream control by reserving rules at OpenFlow switches. Trump card run the show storing empowers administration of movement totals, diminishes stream setup questions, and disentangles strategy administration. Ternary Content Addressable Memory (TCAM) limit issue is a vital issue in Software-Defined Networking. Lead reserving is an effective method to tackle the TCAM limit issue. Be that as it may, there exists run reliance issue in special case run storing procedure. A Cover-set method to solve the rule dependency problem and wildcard-rule caching algorithm to cache rules into TCAM. Cache replacement algorithm considering both temporal and spatial localities.

Keywords: Software Defined Network, Ternary content Addressable Memory, Wild-card rule

I. INTRODUCTION

Software-Defined Networking (SDN) [1, 2] is a brand new network architecture that provides a global view of network state for network administrators to manage network services. SDN controller maintains the flow tables in the switches to comply with network policies. The flow tables are implemented by the TCAM in the modern switches. TCAM can look up a packet's header and compare the matching patterns of the packet to the match field of all rules in the flow table in parallel. In other words, it can forward packets at line rate. Although TCAM can forward packets fast, there are only 2-20K flow table sizes in commodity SDN switches which are much less than RAM based storage [3]. Therefore, the TCAM capacity problem is an important issue in SDN. Previous works on using TCAM efficiently could be classified into three categories, packet classification compression, rules distribution, and rules caching. Packet classification compression [4, 5, and 6] is a technique that combines two similar rules into a new wildcard rule, which is semantically equivalent to the original rules. As a result, we can reduce the number of required rules. Rules distribution technique [7, 8, and 9] splits the set of rules which are safe according to the network policies and distributes them along the network path. In rule caching technique [10], it treats TCAM as a cache which stores the most popular or most-likely matched rules in the future. Once cache miss happens, we should choose the victim rules and evict the victim rules out in order to cache the cache miss rule into TCAM.

One of the rule caching techniques is wildcard-rule caching. Wildcard-rule caching could keep more TCAM space than exact-match rule matching. However, there are different priorities between different rules according to the network policies. If two wildcard rules overlap with each other and we only cache the lower-priority one in the TCAM, the packets matching the overlapping field space of these two rules will improperly match lower-priority rule. In other words, only caching a wildcard rule overlapped with others would cause ambiguous matching when the packets come. Therefore, the most difficult challenge for wildcard-rule caching is to deal with the rule dependency problem. The authors use the cover-set method to solve the wildcard-rule dependency problem.
The cover-set method creates a small number of new rules that cover many low priority rules overlapped with the high-priority rule. High priority rules usually have higher weight due to their larger field space. Therefore, cover-sets help us to avoid caching high weight rules together with many low-weight rules overlapped with them. The Cover-Set Caching (CSC) algorithm calculated the contribution value of each un-cached rule and cached the rule has the maximum contribution value into the TCAM until the TCAM is full. Since the CSC algorithm only considers the contribution value of each un-cached rule, the authors proposed a Wildcard-Rule Caching (WRC) algorithm, which considers the accumulated contribution value for a set of rules. They calculated the accumulated contribution value of a set of related rules and cached the set of rules have the maximum accumulated contribution value into the TCAM until the TCAM is full.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 7, Issue 10, October 2018

II. RELATED WORK

They built Elastic Tree [1], which through data-center-wide traffic management and control, introduces energy proportionality in today's non-energy proportional networks. Also showing how a network admin might configure Elastic Tree to satisfy their needs for performance and fault tolerance, while minimizing their network power bill. Their goal is to significantly reduce this rapidly growing energy cost. Compare multiple strategies for finding minimum-power network. System is energy efficiency, best performance, and fault tolerance. Network operated close to its capacity will increase the chance of dropped and delayed packets. This is very complex system.

The DevoFlow proposal [2] allows operators to target only the flows that matter for their management problem. DevoFlow handles most micro flows in the data-plane, and therefore allows us to make the most out of switch re-sources. Also design and evaluate DevoFlow, a modification of the OpenFlow model which gently breaks the coupling between control and global visibility, aims to allow operators to specify high-level policies at a logically centralized controller. DevoFlow solves problem by allowing a clonable wildcard rule to select an output port. Multipath routing to statically load balance traffic without any use of the control-plane. These methods don't spare much vitality on elite systems. The problem adds more cost and power consumption.

Their design [3] incorporates several critical components that are necessary for enabling security applications in Open Flow networks including role-based authorization, rule reduction, conflict evaluation and policy synchronization.
Exhibit the utility of FortNOX through a model usage and utilize it to look at execution and proficiency. Manage strife examination is performed utilizing a novel calculation. When such conflicts are detected, FortNOX may choose to accept or reject the new rule, depending on whether the rule insertion requester. FortNOX is an important first step in improving the security of Open Flow networks. It demonstrates the feasibility and viability of our alias-set rule reduction approach. Base of the inherent errors. Unable to handle dynamic matching process.

OpenSketch enables [4] a simple and efficient way to collect measurement data. It utilizes information plane estimation natives dependent on item switches and an adaptable control plane so administrators can without much of a stretch execute variable estimation calculations. OpenSketch provides a simple three-stage pipeline (hashing, filtering, and counting), which can be implemented with commodity switch components. They compare OpenSketch with Net Flow and streaming algorithms using packet traces from CAIDA. It has simple, efficient way to control switches. Sketches more flexible in supporting various measurement tasks. Delay of each measurement pipeline component is large. The packet forwarding latency and throughput is reduced.

Software defined networking [5] introduces a vehicle for raising the level of abstraction for network configuration. They have designed and implemented Procera, an event-driven network control framework based on SDN. Also use the OpenFlow protocol to communicate between the Procera controller and the underlying network switches. To enable administrators to express and execute responsive abnormal state approaches in a simpler way. The technologies they describe enable network operators to implement a wide range of network policies in a high-level policy language. It provides better visibility and control over tasks for performing network. This SDN can improve common network management tasks. The evaluation on performance and scalability is less. Procera suffers from the inherent delay caused by the interaction of the control plane and the data plane.

DREAM enables [6] operators and cloud tenants to flexibly specify their measurement tasks in a network, and dynamically allocates TCAM resources to these tasks based on the resource-accuracy. Since the trade-off between resource usage and accuracy can depend upon the type of tasks, their parameters, and traffic characteristics. They design of a system for TCAM-based software-defined measurement, called DREAM. The traffic for each task may need to be measured at multiple switches; DREAM needs to allocate switch resources to each task. User-specified high level of accuracy. DREAM can support more concurrent tasks. DREAM has to reject nearly 50% of the tasks, and drop about 10%. Allocation delay increases as the tasks increases.

They presented [7] a novel control plane architecture called OpenNF that addresses these challenges through careful API design. OpenNF enables applications to settle on appropriate decisions in meeting their goals. Providing such control is challenging because they must address race conditions and accommodates a variety of application objectives and NF types. They show how to combine the two to provably ensure state updates are not lost or reordered during state moves and shared state remains consistent. NF software is always Up-to-Date. High performance on network monitoring. The average time per operation increases linearly. They cannot decrease the total move time without using more rules.

They propose [8] a new technique called Block Permutation (BP) to reduce the number of TCAM entries required to represent a classifier. The BP technique significantly improves the compression rate under the circumstances. The improvement is achieved by performing a series of permutations to change the distribution of rule elements in Boolean Space. They developed an efficient heuristic approach to find permutations for compression and have designed its hardware implementation by using a field programmable gate array. They compress the packet classification rules stored in TCAMs. FPGA is to use the stage-grouping methodology. TCAM can also be applied to other hardware implementation-based applications. Hardware cost is high. Complicated permutations would slow down the pipeline.

III. RULE CACHING SYSTEM

CacheFlow enables [9] fine-grained policies in SDNs by optimizing the use of the limited rule-table space in hardware switches, while preserving the semantics of OpenFlow. In the ongoing work, they plan to explore this trade-offs, and also evaluate our algorithms under a wider range of SDN policies and workloads. As a choice to administer pressure, we regard the TCAM as a reserve that stores the most well-known standards. CacheFlow chooses an arrangement of vital principles from among the standards given by the controller to be stored in the TCAM, while diverting the reserve misses to the product switches. Composing two rules to build a cache would simply involve merging the corresponding two sets. CacheFlow combines the high speed of hardware switches with the large rule tables of software switches, to offer the abstraction of a single, fast switch with arbitrarily large rule capacity.

In this paper [10], they present CAB, a novel reactive wildcard rule caching system. CAB resolves rule dependency while achieving efficient use of control network bandwidth, and reducing controller processing load and flow setup latency. They compare CAB with different rule caching schemes through simulations. The results show that CAB prevails in reducing flow setup requests, controlling bandwidth consumption, and introducing the lowest flow setup latency. Through CAB, they resolve the rule dependency problem with small storage overhead. The primary thought of CAB is to parcel the field space into legitimate structures called cans, and store containers alongside all the related rules.

In this paper [11], they define a hardware-software hybrid switch design called CacheFlow that relies on rule caching to provide large rule tables at low cost. In this paper, also show how to give applications the illusion of high-speed forwarding, large rule tables, and fast updates by combining the best of hardware and software processing. Their CacheFlow system “caches" the most popular rules in the small TCAM, while relying on software to handle the small amount of “cache miss” traffic. CacheFlow consists of a CacheMaster module that receives OpenFlow commands from an unmodified SDN controller.

IV. TERNARY CONTENT-ADDRESSABLE MEMORY

The match line consumes [12] the most power in TCAM search operations. Reducing both the average and peak power consumption is of great importance. The average power saving values can vary significantly with the mask cell programming style. The benefit of conditional power saving is therefore reduced for TCAM when conditional inputs are masked. There are two critical areas in TCAM design that call for innovative power-saving techniques. TCAM is often used in network equipment, which needs line-rate search capabilities for every data packet that arrives at the equipment ports. The extensive power consumption sometimes erases the advantage of the high-speed search operation and prohibits use of TCAM devices.

In this paper [13], they presented a novel memory architecture titled as HP SRAM-based TCAM that emulates TCAM functionality with SRAM memory. It is the first ever TCAM that has been successfully implemented in FPGA. It should further be noted that HP SRAM-based TCAM ensures large capacities TCAMs whereas this capability is lacked by conventional ones. Utilizing SRAM and FPGA, the proposed TCAM is a favourable choice for networking applications. The proposed TCAM may be used in networking systems for routing tables and policing where many data need to be compared in parallel at high speed. Search operation in HP SRAM-based TCAM involves two SRAM accesses followed by a logical ANDing operation.

TCAMs are widely [14] used in network infrastructure for various search functions. There have been growing interests in implementing TCAMs using reconfigurable hardware such as FPGA. This paper shares their efforts and experience on pushing the limit in implementing large TCAMs on a state-of-the-art FPGA. They formalize the ideas and the algorithms behind the RAM-based TCAM and analyze the performance thoroughly. They conduct comprehensive experiments to characterize the various performance trade-offs offered by the configurable architecture. To the best of their knowledge, this is the first FPGA design that implements a TCAM larger than 1 Mbits.

CONCLUSION AND FUTURE WORK

Wildcard-rule caching is an efficient technique to deal with TCAM capacity problem. Although there is rule dependency problem in wildcard-rule caching, we can utilize cover-set method to solve this problem. Our k-HNS algorithm can cache a set of rules into TCAM and has better capability to build the set with higher total weight for each rule. On the other hand, our NSR algorithm both considers temporal and spatial localities, which make the cache hit ratio high. To tackle this limitation, we need a novel power and Time consider wildcard rule Cache Management algorithm.

ACKNOWLEDGEMENT

I profoundly grateful to **Prof.H.A.Hingoliwala** for his/her expert guidance and continuous encouragement throughout to see that this project rights its target since its commencement to its completion. I would like to express my deepest appreciation towards Principal **Dr.M.G.Jadhav,** HOD **Dr.S.B.Chaudhari** department of computer engineering and PG coordinator **Prof.M.D.Ingle**. I must express my sincere heartfelt gratitude to all staff members of computer engineering department who helped me directly or indirectly during this course of work. Finally, I would like to thank my family and friends, for their precious support.

REFERENCES

[1]. B. Heller et al., "Elastictree: Saving energy in data center networks," in Proc. NSDI, vol. 3, pp. 19–21in 2010
[2]. A.R.Curtis, "DevoFlow: Scaling flow management for high performance networks," Comput. Commun. Rev., vol. 41, no. 4, pp. 254–265, Aug. 2011.
[3]. P. Porras et al., "A security enforcement kernel for OpenFlow networks," in Proc. 1st Workshop Hot Topics Softw. Defined Netw., 2012, pp. 121–126.
[4]. M. Yu, L. Jose, and R. Miao, "Software defined traffic measurement with OpenSketch," in Proc. NSDI, vol. 13. 2013, pp. 29–42.
[5]. H. Kim and N. Feamster, "Improving network management with software defined networking," IEEE Commun. Mag., vol. 51, no. 2, pp. 114–119, Feb. 2013.
[6]. M. Moshref, M. Yu, R. Govindan, and A. Vahdat, "DREAM: Dynamic resource allocation for software-defined measurement," in Proc. ACM Conf. SIGCOMM, 2014, pp. 419–430.
[7]. A.Gember-Jacobson et al., "OpenNF: Enabling innovation in network function control," in Proc. ACM Conf. SIGCOMM, 2014, pp. 163–174.
[8]. R. Wei, Y. Xu, and H. J. Chao, "Finding nonequivalent classifiers in Boolean space to reduce TCAM usage," IEEE/ACM Trans. Netw., vol. 24, no. 2, pp. 968–981, Apr. 2016.
[9]. Naga Katta, Omid Alipourfard, Jennifer Rexford and David Walker, "Infinite CacheFlow in Software-Defined Networks," ISBN: 978-1-4503-2989-7, August 2014.
[10]. Bo Yan, Yang Xu, Hongya Xing, Kang Xi, H. Jonathan Chao, "CAB: A Reactive Wildcard Rule Caching System for Software-Defined Networks," Association for Computing Machinery, August 2014.
[11]. Naga Katta, Omid Alipourfard, Jennifer Rexford and David Walker, "CacheFlow: Dependency-Aware Rule-Caching for Software-Defined Networks," ISBN 978-1-4503-4211-7, March 14–15, 2016.
[12]. Sanghyeon Baeg, "Low-Power Ternary Content-Addressable Memory Design Using a Segmented Match Line," IEEE Transactions On Circuits And Systems—I: Regular Papers, Vol. 55, No. 6, July 2008.
[13]. Zahid Ullah, Kim Ilgon, and Sanghyeon Baeg, "Hybrid Partitioned SRAM-Based Ternary Content Addressable Memory," IEEE Transactions On Circuits And Systems—I: Regular Papers, Vol. 59, No. 12, December 2012.
[14]. Weirong Jiang, "Scalable Ternary Content Addressable Memory Implementation Using FPGAs," IEEE Architectures for Networking and Communications Systems December 2013.