Adaptive Monitoring using Openflow

[Extended Abstract]

Author One*
Institute One
Address One
author.one@emails.com

Author Two
Institute Two
Address Two
author.two@emails.com

Author Three
Institute Three
Address Three
author.three@emails.com

ABSTRACT

This paper present a network monitoring architecture which take advantages of Software Defined Network and traditional monitor methods. Our architecture support the a number of features: 1)monitoring all the flows with low cost, 2)adjust the work condition automatically according to the flow status, 3)accept instructions from users and other applications to control the flow, 4)combined with traditional detection methods.

optional: 1)support long-term flow analysis...

General Terms

Application

Keywords

network monitor, openflow

1. INTRODUCTION

In modern datacenters, collecting the network flow statistics information and DPI information is important for attack detection and resource allocation. By monitoring the network flows, we could know whether there's irregular flows, which part is the most active and when to upgrade the whole system. If the bandwidth demands of servers and applications are changing dynamically, it is too difficult to give reasonable predictions and only real-time monitoring is feasible.
However, the procedure of monitor all the network flows usually cost too much and we have to make a compromision between efficiency and accuracy.
(TBA) More details
(TBA)Our main contribution.

2. RELATED WORK

(TBA)Related Work
openflow: [2]
B4: [1]

3. OUR METHOD

(TBA)Brief summary of related work. Discuss the advantages and disadvantages of Sample, Mirror and so on.
(TBA, motivation)Now we consider the cost when we try to get the network flow information. If we use openflow table to monitor an point-to-point flow, the cost is one flow entry. If we mirror a flow to a server, the cost is the bandwidth. In our system, we use openflow entry to monitor large flow, and mirror small flows to the server.
The rest of the sections is arranged as follows. The ystem architecture part described the hardware architecture and flow table design, and why we build our system like this. The section about elephant flows describes the algorithm and feedback control on filtering large flows

3.1 System Architecture

Hardware Structure

(image)
user –> controller
controller <–> switch
switch —> servers
switch –> mirror server
switch –> snort node(third-part tools)
mirror server –> controller
snort node –> controller
(add port information, and link to controller is inband)

Briefly introduce the structure of different part and their relationships.
Briefly intrudoce the detailed structure of the controller.
Flow table Struction:
(image)
flow table 1:
for high level monitor (user different priority)
flow table 0:
for determinate flow monitor while can make routing decision by oneself (high priority)
fow learning switch(low priority), inlucding the lowest priority packet-in

*sample titlenote

(TBA)Explain why use the high level part to monitor and routing, but not add another table. (Due to the hardware implentation of multi-table on Pica8...–Can we write like this?–)
(TBA)Explain why use table one for high level monitor. (It can't make routing decisions by one self. eg. all flow come from a given IP.)

3.2 Flow Bandwidth Monitoring

Bandwidth Detection:
Use flow entris to filter elephant flow.
Use feedback to adjust monitoring entries. If the mirror bandwidth extend the port bandwidth, times the factor for feedback result

Flow Entry Replacement Algorithm
All use generated flow matrix to test. Easy to reappear.
Random vs Greedy vs Damping using long-term history and dft
compare target: mirror bandwidth, monitor precision, entry change time and so on.
Use dft to detect whether the flow is mutable or stable. Combine it with the average flow size to determine whether to add the corresponding monitor entry.
Store long-term information also help to ddos detection.

3.3 DDos attack detection

Store all the flow statistics information of a period time. For each src node and dst node, use the relavent flows to determine whether it is a potential attacker or a target.

3.4 Deep Package Information Detection

The shortage of our SDN monitor is we can't look up into the packets on the switch. Even we can look into the information in a packet when packet-in event happens, we can't affort the cost of packet-in. A general controller could only deal with 1k-10k packet-in request, which we will test later.
snort

3.5 Operation for User Command

Monitor Flows, Block Ports
Change work status.

4. EVALUATION

4.1 Environment Setup

Hardware parameters introduction. Pica8-3295 and server models
Real System connection.

4.2 Basic Environment Test

Test on Switch and Controller:
Basic Test.
Use cbench or other tools. Test on standard learning switch between our switch.

Compare the Bandwidth and Latency under L2 and Openflow:
use scripts (got help from liyiran)
Test bandwidth between servers under L2 model and Openflow model.
(image)
Test bandwidth between servers under Openflow model with insert/delete monitor flows on different frequency.
(image)

Test the Bandwidth and Latency influence of inserting and deleting monitor flows.
Also test whether it lead to packet-loss when insert and delete flow entries.

4.3 Precision

Bandwidth(Elephant FLow):
Compared with sFlow.
Use generated flows, Compare the precision.
When bandwidth is not full, when mirror port is full.
absolute precision, precision with sflow.

DDoss:
Compared with? absolute precision, precision with sflow.
long-term feature

4.4 Cost, Cost includes flow entry, mirror bandwidth

Use generated flows
As all are real result, not necessary to test active time.

4.5 DPI and User's command

4.6 Real Applications (eg. Spark cluster.))

5. CONCLUSION

Conclusion

6. REFERENCES

[1] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu, et al. B4: Experience with a globally-deployed software defined wan. In *Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM*, pages 3–14. ACM, 2013.

[2] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner. Openflow: enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review*, 38(2):69–74, 2008.