Vol.2014-CSEC-66 No.50
Vol.2014-SPT-10 No.50
2014/7/4

A user mode implementation of filtering rule management plane on virtualized networking environment

Ruo Ando[1,a)]

Abstract:
With the rapid advance of virtualization and big data technology which realizes SDN (Software Defined Network) and Cloud computing, today's computing environment becomes more flexile, diversified and complex. In this paper we present a user mode support for centralized filtering rule management base. Proposed system enables us to handle fine grained traffic engineering functionality for diversified environment of Cloud and SDN. Our architecture adopts NoSQL data store for handling a large scale of filtering rules. By leveraging data store for centralized access control of instances on virtualized environment, we can provide alternative access control framework for reducing the burden of managing complicated and dynamic filtering policy on instances (virtual machine) on virtualized networking environment. In experiment, we have prototyped a lightweight management plane for IP filtering. Access filtering rules including target IP address, prefix and gateway is represented as radix tree. It is shown that proposed method can achieve reasonable utilization in filtering IP packets.

1. Introduction

The rapid advance of virtualization technology have realized SDN (Software Defined Network) and Cloud computing, today's computing environment becomes more flexible, diversified and complex. Actually, innovative open-source products such as open vSwitch and KVM, users and administrators have obtained flexible, tight and elastic control over their networking environment.

Besides, these innovative technologies causes a situation that diversified service and applications is running. With a rapid popularization of advanced virtualized networking technology such as Cloud and SDN, diversified organizations start adopting data centers. For example, universities, private companies as well as commercial corporations heavily utilize cloud services such as IaaS for running a lot of kinds of applications and cloud based services.

As a result, this kind of virtualized networking infrastructure has been important and hence become mission-critical services. these services range from financial, medical applications which is internet facing sensitive applications to computing resource intensive services guch as big data analysis and indexing huge web contents.

Based on these backgrounds, SDN(Software Defined Network) has been emenged for more flexible and elastic hetworking. Basically, software dined networking enables us to make it simple to manage networks by providing developers a wide range of network visibility and tight control over underlying switches from a logically centralized controller. This type of architecture provides more flexibility and fine-grained traffic functionaility.

However, these promissing technologies currently hare several points to be improved and therefore causes unfortunate shortcomings. Particularly, in the aspects of information security, current access control framework is not suitable for advanced virtual networking. For example, administrators of Cloud infrastructure need lots of fine-grained rules to keep adequate control over indivisual network flows in order to handle various management tasks. For this purpose, conventional access control methods need to be improved for better centralized filtering rule management.

For alternative access control model, Virtual Cloud Information Base [2] is proposed in order to enable adminstrators to configure different management policies independently from considering underlying resource constraints. For the most part previous research efforts have advocated putting rule processing module On either on switches ov hypervisors. However, huge network such as future data center would benefit from taking advantage of rule processing capabilities both ot scalabity and performance.

For one step further concerning flexibility and manageability, we present a new centralized access control framework which can be deployed in user-mode. Besides, Proposed system adopts fast and scalable datastore module for fine-grained traffic engineering.

2. Design requirements

As we mentioned in pvevious section, current access control framework has gowe short comings for more fine-grained, flexible and distributed architecture of Cloud computing and SDN. Many of existing access control techniques have been originally aimed to handle task management in enterprise environments. These environments are not affected by the innovations of virtual

[1] Network Security Institute, National Institute of Information and Communications Technology 4-2-1 Nukui-Kitamachi, Koganei, Tokyo 184-8795, Japan
[a)] ruo@nict.go.jp

networking and therefore do not share with new challenges concerning cloud and SDN. To put it simply, the majority of traffic control techniques are poorly suited for virtual networking environments. The requirements for improving are scalability and robustness better than existing network based techniques.

Specifically, network level access control should be considered. For preserving security in migrating the Cloud computing model, network based traffic filtering policies are critical issue. Today, access control in Clould environment is typically enabled by adopting techniques such as VLANs and firewalls. These techniques, unfortunately, originally'implemented for single tenant environment such as company network and hence ill suited to hinder the problem which is specific for Cloud and SDN environment.

2.1 Scalability and diversity

One of the most important features of virtualized networking is scalability. Many existing Cloud systems are designed for coping with scalability. Garfinkel[15] pointed that creating a new virtual instance is far easier than physical environment. We can increase the number of virtual machines like copying files while the increase in physical machines is ultimately bounded by configuration time, budget and so on. In Cloud computing environment, we can often have several or even dozens of special purpose virtual machines. To make things worse, ali administrative tasks are hardly automated inthe case that the number of VM kan increase at an explosive rate. For example, updating systems, patch management and configurations on a large number of VMs will impose a great burden on administrators.

Scalability. conventionally, increasing physical machines has a limit which is determined by setup time and equipment budget. virtualized networking technology nullifies these limits. Like copying files, users will easily have several or even dozens of special purpose VMs. In worst cases, the number of instances can grow at explosive rate which rarely administrators can handle. Consequently, the rapid and unpredictable growth can exacerbate management tasks and in worse case the impact of catastorophic events can be multiplied where all instances should be patched.

Diversity. Enforcing homogenity is common way to cope with security problems for many IT organizations. However, as we mentioned before, under current situation that users cau hare their own special purpose VMs easily, diversity can sometimes a burden for security administrators. Specifically, a hard problem has been occurred about maintaining patches a protecting for a wide range OS, which is vulnerability management. In this case, some risks will be imposed by having so mary VMs unpatched in the network.

2.2 Flexibility

For a long period, from active networks to advanced network technologies like cloud and SDN, one of the general goals of networking research has been arrived at a network which is flexible. Ideally, the allure of more flexibility could be that it enables us to incorporate new ideas into the network with reasonably low cost. For flexibility, these ideas aims to enhance the existing network in the point of manageability and reachability. AIso, these ideas could be altogether new business models and applications.

2.3 Fine-grained functioning

As we discussed before, there are more requments for new type of network architecture compared with conventional enterprise networks. Hence network administrators should achiere fine-grained traffic engineering. Specifically, for alternative access control, we have new challenges for providing three properties: flexibility, network independence and scalability. Flexibility is required specifically in company with tenant isolation, fair sharing and rate limiting operation. For decoupling access control, network should be independent from the network topology and addressing. Also, in Clould environment, we need scalability in handling hundreds of thousands of machines. Fine-grained functioning is important because the performance of Cloud services and network applications much depends on efficient functioning of data centers infrastructure.

3. Architecture

Our system architecture is roughIy identical to generic openflow implementation such as open vswitch. the thrust of proposed system is adopting NoSQL datastore for achieving scalability.

3.1 System overview

Ganeric software defined networks basically consists of two planes: control plane and data plane. control plane is responsible for deciding whether the packet should be forwarded or dropped. Also, control plane play a role for monitoring and propagating event notification to distributed controller such as monitor performance, faults and alarms. Data plane works as conventional forwarder without filtering functionality. In some cases, data plane is responsible for program the HW pipeline to create/update/release circuits/Lsp.

One of the main features of proposed system is implementing access control module ontside both data and control plane. By doing this, filtering rules distributed in each instance can be handled in centralized manner.

3.2 Forwarding and control plane implementation

Essentially, for every fundamental layers, implementing bridge module is necessany. Because in OSI model, bridging acts in the low two layers. Network bridging enables function to communicate between two ormore segments and create an aggragate network. Bridging is different from routing in the pointthat routing enables the network to communicate independently to manage as separate networks. A network bridge is an interface for connecting multiple segments in the network.

For implementing forwarding plane, we have two bridging decide of which mode is either POLLIN and POLLERR. When the packet is arrived at one bridge interface, it is forwarded to another packet after validating it as shown in the snippet below.

Listing 1

Main loop

```
1
2 main loop.
3
```

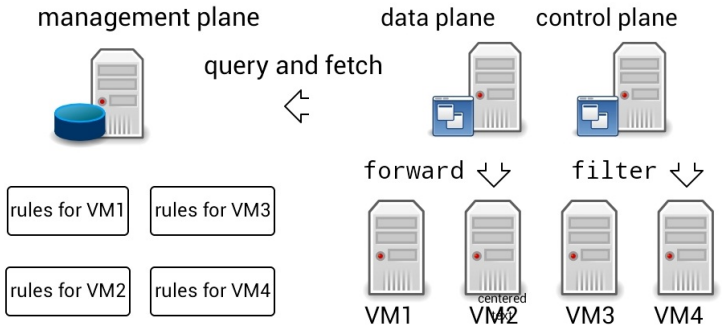

Fig. 1 System overview.

```
4  for(i=0;i<2;i++){
5  if(targets[i].revents&(POLLIN|POLLERR)){
6
7
8  inward packet.
9  if((size=read(Device[i].soc,buf,sizeof(buf)))<=0){
10
11 validating packet.
12 if(ProcessPacket(i,buf,size)!=-1){
13
14 outward packet.
15 if((size=write(Device[(!i)].soc,buf,size))<=0){
```

First argument of write system call is socket descriptor and second one is socket baffer.

3.3 POLL file descriptor

In running network devices, various events should be notified such as changing files, siginals raised and packet arrived. specifically in linux, read()and other many mechanisms exist to handle single event or any of a set of events. These mechanisms are enabled by POLL system call. For this purpose, the 〈poll.h¿ header defines the structure of poll file descriptor.

Listing 2
Polling

```
1  POLLIN
2  Same effect as POLLRDNORM | POLLRDBAND.
3  POLLRDNORM
4  Data on priority band 0 may be read.
5  POLLRDBAND
6  Data on priority bands greater than 0 may be read.
7  POLLERR
8  An error has occurred (revents only).
```

3.4 Backend datastore

For centralized filtering rule management, proposed system adopts data store mongoDB for storing a large number of rules rapidly while achieving scalability.

MongoDB which stands for humongous a cross-platfora document oriented NoSQL database.By eschewing the traditional relational DB structure in favor of JSON—like documents which is called as BSON. By introducing BSON, MongoDB makes it possible to integrate data in certain types of applications faster and easier.

According to CAP theorem, MongoDB gives up consistency for availabity and partition tolerance. CAP theorem states that a distributed system cannot satisfy these three guarantees at the same time. Consistency means that all nodes see the same data at the same time. Availability means that a guarantee that every request receives a response about whether it was successful or failed. Partition tolerance means that the system continues to operate despite arbitrary message loss or failure of part of the system.

While openflow decides which packet should be passed or rejected in control plane module, proposed system decide it on each instance and management plane stores filtering rules of each instance. when the packet arrives at data plane, its instance (VM) fetches corresponding rules from management plane.

Listing 3
Data
store

```
1
2
3          while(cursor->more()) {
4            mongo::BSONObj p = cursor->next();
5            mongo::OID oid = p[''_id''].OID(); // retrieve
                 ObjectId
6
7            string dest = p[''dest''].str();
8            int mask = p[''mask''].numberInt();
9                 string gateway = p[''gateway''].str();
10
11                     const char *p0 = dest.c_str();
12           const char *p1 = gateway.c_str();
13
14           add_rtentry(p0, mask, p1);
15
16                cout << ''retrieved rule : ObjectId: '' << oid
                       << endl;
17           cout << ''retrieved rule : dest: '' << dest <<
                  endl;
18                cout << ''retrieved rule : gateway: '' <<
                       gateway << endl;
19
20           int res;
21
22           res = find_route(dstAddress);
```

Listing 3 is an example of applying for filtering table matching. in ingress packet arriving, a process queries datastore corresponding rules. For each rule, forwarder searches next destination and checks whether the ingress packet can or should be forwarded to the next device.

3.5 Rules specification

Proposed system represents filtering rules by using the associative array which is called as key-value. Generally, map or directory is the fundamental data model of key-value store architecture. In coping with a collection of key-value Pairs, each possible key appears at most once in the collection.

The key-value store model is powerful in the point that it can

be extended to an ordered model with each key maintained in lexicographic order. Owing to the simplest non-trivial data models, efficient lsey processing, these extensions can be powerful.

Listing 4

BSON

```
1
2 { "_id" : { "$oid" : "53370eaeb1f58908a9837910" }, "
      dest" : "10.0.0.0", "mask" : 8, "gateway" : "
      192.168.0.2" }
3 { "_id" : { "$oid" : "53370eaeb1f58908a9837910" }, "
      dest" :8.8.8.8",␣"mask"␣:␣8,␣"gateway"␣:␣"
      192.168.11.2"␣}
4 {␣"_id"␣:␣{␣"$oid"␣:␣"53370eaeb1f58908a9837910"␣},␣
      "dest"␣192.168.0.1", "mask" : 8, "gateway" : "
      192.168.1.2" }
5 { "_id" : { "$oid" : "53370eaeb1f58908a9837911" }, "
      dest" : 172.1.0.1",␣"mask"␣:␣8,␣"gateway"␣:␣"
      192.168.0.2"␣}
```

MongoDB adopts its own specific format which is called as BSON. Like JSON, BSON is binary based serialization and encoded format which is designed to be lightweight,traversable and efficient. This format is also called as document based document-based which supports the embedding ot objects and arrays. In general cases, data storage and network transfer are applications of BSON.

3.6 Radix tree

Radix tree which is also called as compact prefix tree is a representation for processing tree data structure in a space-optimized manner. Radix tree is similar to binary search tree in the point that only one child node is merged with its child. For adapting radix tree for processing prefix in routing, a program code can be written as follows.

Listing 5

Radix tree

```
1
2  static int
3  find_route(const char *dst)
4  {
5          in_addr addr_dst;
6          rtentry entry;
7
8          if (inet_aton(dst, &addr_dst) == 0) {
9                  std::cout << ‘‘invalid address: dst = ‘‘ <<
                       dst
10                         << std::endl;
11                 return 1;
12         }
13
14         entry.addr = ntohl(addr_dst.s_addr);
15         entry.prefix_len = 32;
16
17         radix_tree<rtentry, in_addr>::iterator it;
18
19         it = rttable.longest_match(entry);
20         if (it == rttable.end()) {
21                 std::cout << ‘‘no route to ‘‘ << dst << std::
                       endl;
22                 return 1;
23         }
24
25
26         char *addr = inet_ntoa(it->second);
27         std::cout << dst << ‘‘ -> ‘‘ << addr << std::endl;
28         return 0;
29 }
```

4. Experiment

A concept proof code of proposed system is implemented as simple linux bridging interface with c++ code. We modified IP analyizing part of packet capture to leverage radix tree for representing filtering rules. To evaluate performance overhead of proposed system in filtering packets according IP address, we deployed our prototype into Amazon VPC(virtual private cloud) with several Micro instances. We compiled our system on ubuntu12 LTS with Linux kernel 3.2.0. proposed system is hosted on Intel Xeon E5645 with 2.4 GHZ clock. The experiment was conducted by a dedicated packet generator and main GW server of which two interfaces works as forwarding plane. Two modules are identical in the same spec.

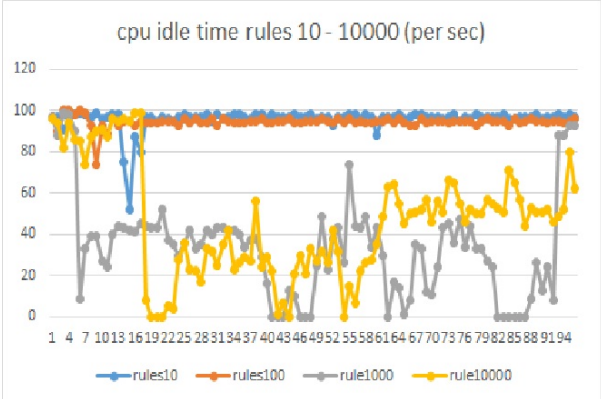

Fig. 3 CPU idle time (per sec)

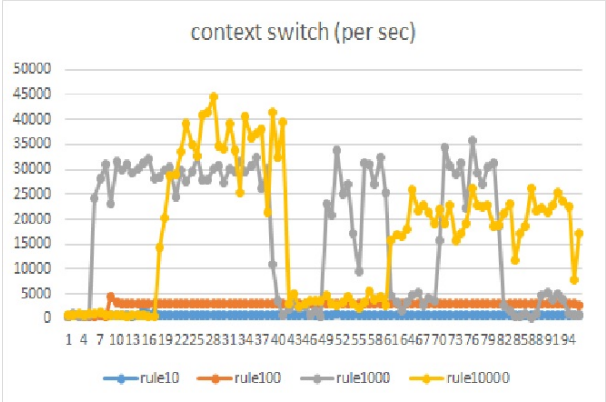

Fig. 4 Context switch (per sec)

Figure 3 and 4 depicts the system utilization of CPU idle time and context switch in applying rules 10,100,1000 and 10000. The same experimental environment was run with proposed system where we varied the number of rules. Our objective is to measure the computational impact of conducting filtering rules. It is shown that with the number of rules 10 and 100, the utilization of conducting filtering rules is reasonable. We can conclude that the computational impact is acceptable because the number of rules up to 100 is enough for micro instances in VPC.

5. discussion

The emerging network architecture such as SDN and Cloud computing make us face to new challenges for innovating current access control technologies in many aspects like multi—tenancy, distributed controller and virtualized network nodes. Besides, the rapidly expanding scale and dynamicity of hosts within the virtualized networks cannot be handled in the increasing diversity of

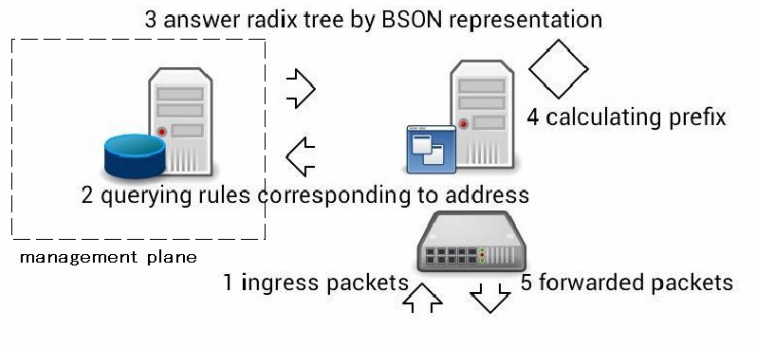

Fig. 2 Prototying management plane

virtualized networking architecture.

Basically, today's access control in virtualized networking is realized by using generic techniques such as VLAN and firewalls. Many of deployments of these techniques were designed for enterprise environments. unfortunately, the drastic change between enterprise environments and latest virtualized networking makes these techniques not to share these challenges. CloudPolice[1] is a system which implements access control within hypervisors. Although CloudPolice provides a effective way to cope with new access control problems, modules in hypervisors are sometimes hard to be installed and deployed.

Second challenge is scalability. In today's virtualized networks, tens of thousands of physical machines and even more virtual machines are generated, removed and added. in this situation, conventional access control frameworks are not suitable to handle such scale and churn. For instance, VLANs do not support dynamic configuration and for scaling to large number of instances and running across multiple gateways, firewall tends to have many problems.

Third challenge is network diversity. Among emerging network architecture, data centers has been progressed from ones of the traditional enterprises and consequently has become flut with many proposals of new systems.

6. Related work

Traditionally, VLANs are a main mechanism to realize AC (access control). Unfortunately, we are forced to cope with several shortcomings ia running the emerging new networking architecture. First, in the new architecture such as [3,4,5], vlans which couples access control and switching art not suitable. this is also the case with L3 routing as well as switching. Besides, some specific operations such as creating spanning tree and switching between VLANs has unreasonable overhead. As well as not being able to provide the flexible policies, VLANs are limited by the number of hosts in a single VLAN.

Another technology for blocking unwanted traffic is firewall. However, a mainteinance overhead of firewall is drastically increased in cloud and SDN environment since all necessary updates is increased everytime a destination changes its policy. Even in the case of adopting group-based policies, maintenance overhead of creating entry for each instance is unreasonable. Partly because this solution should face scalability limit.

OperFlow and Ethane [10] is powerful technology to provide flexible AC. However, deployment cost is not low because these require modification to the switching hardware. concerning this issue, Open vSwitch[11] can be deployed in network-independent manner, but its centrallized management require a centrallized controller.

From this viewpoint, open vswitch includes all drawbacks of centrallized approaches which has a scalIng bottlenecks. Also, centrallized controller could be single point of failure and a potential attraction for DoS attacks. In addition, compared with hypervisor based AC management, the operFlow API is poorly designed except for switches. A Problem of addness assignment which is the underlying topology independent in discussed in VL2[12]. However, VL2 is still centrallized control based and therefore inherits the same drawbacks of open vswitch.

7. Conclusions

The emerging technology of virturalized and software defined networking have made us face the challenges for alternative access control model with flexiblity, scalability and diversity. For coping with the today's more sophisticated networking environment, more fine-grained and intricated traffic engineering is necessary.

In this paper, we have presented new access filter management system which can be run on user-mode. Proposed system adopts scalable NoSQL datastore which enables us to handle fine-grained traffic functioning. By leveraging data store for centralized access control of instances on virtualized environment, we can provide alternative access control framework for reducing the burden of managing complicated and dynamic filtering policy on instances (virtual machine) on virtualized networking environment.

In proposed system, access filtering rules are represented by BSON which is JSON-like data structure. In experiment, we have prototyped a lightweight management plane for IP filtering. Access filtering rules including target IP address, prefix and gateway is handled by radix tree. It is shown that proposed method can achieve reasonable utilization in filtering IP packets in virtualized networking environment. For further work, we are enhancing backend datastore modules. Also, implementing access control specific DSL mll provide more flexibility for operating virtualized network.

References

[1] CloudPolice: Taking access control out of the network Lucian Popa, Minlan Yu, Steven Y. Ko, Ion Stoica, Sylvia Ratnasamy 9th ACM Workshop on Hot Topics in Networks (HotNets-IX). Monterey, CA, October 2010.

[2] VCRIB: Virtualized rule management in the cloud Masoud Moshref, Minlan Yu, Abhishek Sharma, Ramesh Govindan the 4th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud). Boston, MA, June 2012.

[3] M. Al-Fares, A. Loukissas, and A. Vahdat. A scalable, commodity data center network architecture. In SIGCOMM. ACM, 2008.

[4] C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang, and S. Lu. BCube: A High Performance, Server-centric Network Architecture for Modular Data Centers. ACM SIGCOMM, 2009.

[5] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, and S. Lu. Dcell: A Scalable and Fault-tolerant Network Structure for Data Centers. In SIGCOMM, 2008.

[6] P. Garimella, Y.-W. E. Sung, N. Zhang, and S. Rao. Charac terizing VLAN usage in an operational network. Workshop on Internet Network Management, 2007.

[7] Y.-W. E. Sung, S. Rao, G. Xie, and D. Maltz. Towards Systematic Design of Enterprise Networks. In ACM CoNEXT, 2008.

[8] M. Yu, X. Sun, N. Feamster, S. Rao, and J. Rexford. Virtua l LAN Usage and Challenges in Campus Networks. Princeton University Technical Report 2010
http://www.cs.princeton.edu/ jrex/papers/vlan10.

[9] The OpenFlow Switch Consortium: www.openflowswitch.org.

[10] M. Casado, M. J. Freedman, J. Pettit, J. Luo, N. McKeown, and S. Shenker. Ethane: Taking control of the enterprise. In ACM SIGCOMM, 2007.

[11] B. Pfaff, J. Pettit, K. Amidon, M. Casado, T. Koponen, an d S. Shenker. Extending Networking into the Virtualization Layer. In HotNets, 2009

[12] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim , P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta. VL2: A Scalable and Flexible Data Center Network. ACM SIGCOMM, August 17 - 21 2009.

[13] Kevin R. Fall, Gianluca Iannaccone, Maziar Manesh, Sylvia Ratnasamy, Katerina J. Argyraki, Mihai Dobrescu, Norbert Egi: RouteBricks: enabling general purpose network infrastructure. Operating Systems Review 45(1): 112-125 (2011)

[14] Lucian Popa, Ion Stoica, Sylvia Ratnasamy: Rule-based Forwarding (RBF): Improving Internet's flexibility and security. HotNets 2009

[15] Tal Garfinkel , Mendel Rosenblum, When virtual is harder than real: Security challenges in virtual machine based computing environments, HotOS 2005