Network-resource Isolation for Virtualization Nodes

Yasusi Kanada
Central Research Laboratory,
Hitachi, Ltd.
Totsuka-ku Yoshida-cho 292,
Yokohama 244-0817, Japan
Yasusi.Kanada.yq@hitachi.com

Kei Shiraishi
Network Solution Division,
Hitachi, Ltd.
Saiwai-ku Kashimada 890,
Kawasaki 212-8567, Japan
shiraishi_kei@itg.hitachi.co.jp

Akihiro Nakao
The University of Tokyo Interfaculty, Initiative in Information Studies, Graduate School of Interdisciplinary Information Studies
Bunkyo-ku Hongo 7-3-1,
Tokyo 113-0033, Japan
National Institute of Information and Communications Technology
Bunkyo-ku Hakusan 1-33-16,
Tokyo 113-0001, Japan
nakao@iii.u-tokyo.ac.jp

In network virtualization [Nak 10a], it is important to avoid resource interference, e.g., concerning communication bandwidth and delay, so that a single *slice* (virtual network) may not disrupt the whole infrastructure. To avoid this type of interference, a method for *network-resource isolation* (*NRI*) must be developed. Two types of traffic-control mechanisms, i.e., traffic shaping and traffic policing, can be used for NRI. To more effectively implement NRI using these mechanisms, two methods are proposed in this study: the "per-slice shaping" using weighted fair queuing (WFQ) and the "per-link policing" (per-virtual-link policing).

I. Virtualization Platform, VNode, and Slice

When many users and systems share a limited amount of resources on computers or networks, virtualization technology creates the illusion that each user or system owns resources of their own. Concerning networks, WANs are virtualized by using VPNs. Nowadays, networks in data centers are virtualized using VLANs while servers are virtualized using VMs. In new-generation network research projects such as PlanetLab [Tur 07] or GENI [GEN 09], it is necessary to develop virtualization technology that makes it possible to build network environments where slices are isolated from one another so that they may develop new generation network protocols without disrupting the other slices.

Network virtualization is realized through slices and the *virtualization platform* that manages slices. A project called the *virtualization node project* (*VNP*) [Nak 10b] aims to build virtualization-platform technology and a high-performance fully functional virtualization testbed. In the virtualization-platform architecture in this project, a network domain is managed by a domain controller (DC), and each domain has two types of physical network nodes (See **Figure 1**). A type of node is *VNode* (virtualization node), which forwards packets on the platform. Each packet on the platform contains a virtualized packet on a slice. VNodes are connected by tunnels using a protocol such as GRE. Therefore, a slice with free topology can be created. Each VNode consists of the following three components.

- *Programmer* is a component that processes packets on the slices. Slice developers can program programmers.
- *Redirector* is a component that can forward or route packets on the virtualization platform and forward (redirect) packets from another VNode to a programmer or forward packets from a programmer to another VNode.
- *VNode Manager* is a software component that manages the VNode according to instructions from the DC.

Figure 2 shows a more detailed internal structure of a VNode. There exists a pair of packet encoder and decoder between the redirector and the programmer. The decoder converts the VNode-external data format to the internal format and the encoder converts vice versa. Traffic that comes to the ingress interfaces (to the left) follows the curved line and passes through the shaper, the decoder, the programmer, the policer, the shaper again, and the encoder, and is outputted through the egress interfaces (on the right).

In the model developed by VNP, a slice consists of the two types of components (See **Figure 3**) [Nak 10a].

- *Node Sliver* represents computational resources that exist in a VNode (in a programmer). It is used for node control or protocol processing with an arbitrary packet format. It is generated by slicing physical computational resources.
- *Link Sliver* represents resources of a virtual link that connects two node slivers. It is generated by slicing physical network resources such as bandwidth.

The DC of a domain receives a slice design by an XML-based *slice definition*. The DC distributes it to each VNode Manager, which sends information required for node-sliver configuration to the programmer and sends information required for link-sliver configuration to the redirector.

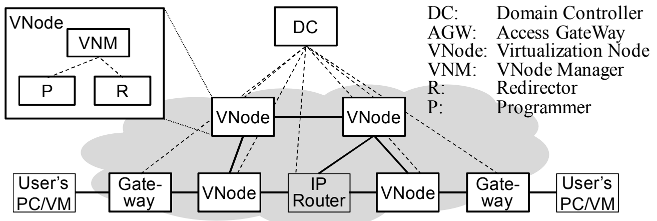

Figure 1. Physical structure of virtualization platform

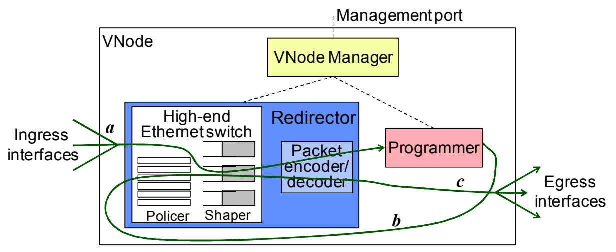

Figure 2. Policing and shaping in a VNode

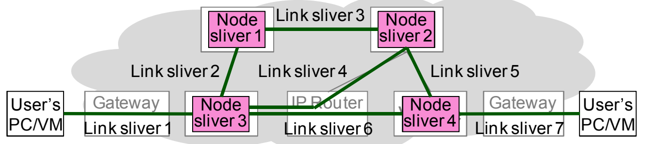

Figure 3. Example of slice design

II. Resource Isolation and Specification

The user of a slice should be able to use the slice without being affected by the behavior of other slices created on the same platform. For this purpose, methods for resource isolation must be developed. The virtualization platform has three types of resources, so three types of resource isolation can thus be classified: *network-resource isolation* (*NRI*), *computational-resource isolation* (*CRI*), and *storage-*

resource isolation (*SRI*).

Redirectors play the most important role in NRI because they measure and control traffic between VNodes. NRI can be implemented by using QoS functions of the VNodes, especially shaping and policing. It is necessary to specify bandwidth (and the burst size) in defining a link sliver with NRI. In the following example, the bandwidth is specified as 30 Mbps, and the burst size is specified as 10 kB.

```
<linkSliver type="link" subtype="GRE"
      name="LinkSliver1">
  <vports><vport name="port0" /><vport name="port1" />
  </vports>
  <resources><resource key="bandwidth" value="30M" />
             <resource key="burstSize" value="10k" />
  </resources></linkSliver>
```

III. METHODS OF NETWORK-RESOURCE ISOLATION

Two major traffic control functions can be used for NRI. *Traffic shaping* queues packets and controls and limits the egress traffic. If the egress queue is filled, the packet output is delayed, and packets may be dropped. *Traffic policing* measures network traffic without accumulating packets and that drops packets when the bandwidth or the burst size exceeds a predefined limit. Traffic policing can be used for guaranteeing bandwidth of link slivers that shares a queue. If the total bandwidth is guaranteed by traffic shaping or any other method, the bandwidth can be divided to the link-slivers and each bandwidth can be guaranteed by using policing. The cost of policing is much lower than shaping because policing requires much less queues.

Two methods for implementing NRI are proposed in this study: per-slice shaping and per-link policing. In *per-slice shaping*, traffic in a slice, including both ingress and egress traffic, is aggregated and shaped. Although traffic shaping can be used to avoid interference in both bandwidth and in delay, it is relatively expensive, because a queue must be allocated to each link sliver to guarantee QoS and this allocation causes high packet-scheduling overhead. As a result, the scalability may be sacrificed.

We believe *per-slice* queue allocation suffices our objective because the purpose of resource isolation is to avoid interference between slices, but not to guarantee QoS thoroughly. It is not necessarily required to avoid interference between flows within a slice. The bandwidth used for each slice is guaranteed, so interference between slices is avoided. The number of link slivers connected to a node sliver is considered to be typically 3 to 5; therefore, the number of queues can be reduced by 80 to 90%.

In *per-link policing*, the second method, an ingress network interface with a policing function is used. A rule for packet P, which implements the link-sliver definition described above, is shown.

```
if (P.slice = S and
    (S.bandwidth>=30Mbps or S.burstSize>=10kB)) drop P;
```

Tens or hundreds of slices may share an egress queue in per-link policing. The queue may be a bandwidth-based (e.g., WFQ-based) or a priority-based queue. If the former is used, the bandwidth of the queue must be (at least) the sum of the bandwidths of the slices sharing the same queue.

Per-link policing is more scalable than per-link shaping; that is, there may be thousands or more slices even when a filter is created for each link sliver. Policing can avoid bandwidth interference, although policing cannot avoid interference or correlation concerning delay completely.

Per-slice shaping and per-link policing may be combined to improve the characteristics of resource isolation.

IV. IMPLEMENTATION AND EVALUATION

Three methods for NRI, i.e., per-slice shaping with/without policing and per-link policing, have been implemented in our VNode prototype. We show the evaluation result of the round-trip delay, jitter, and packet-drop ratio of four types of link slivers, i.e., per-slice shaping with/without policing, per-link policing, and best-effort using the prototype. Three slices are used: one for foreground traffic to be measured and two for background cross traffic.

Table 1 lists the measurement results using slow-path (Linux VM) node slivers. It is clear that both drop and delay interferences are mostly suppressed; there are only small differences between the cases except the no-isolation case. The delay is slightly larger in the per-link policing case (1.6) than in the other two methods and the congestion-less case (1.3). This result shows that the per-slice shaping is effective for delay-sensitive services while per-link policing may be sufficiently used for the other types of services.

We have also measured a foreground slice with higher-bandwidth (4.55 Gbps) link slivers and a fast-path node sliver (using a network processor) with two background slices. To keep the traffic acceptable to the node sliver, the total traffic is shaped to 4.0 Gbps by the shaper just before the SMC. The result shows there is no significant increase in packet drop ($< 4\times10^{-5}$) both in per-slice shaping and per-link policing when the bandwidth of the foreground traffic is 4.0 Gbps or less and the packet size is 1350 B.

Table 1. NRI performance of 100Mbps link-sliver (90Mbps)

Isolation type	Delay (mS)		Jitter (mS)		Drop ratio	
	Average	Std dev	Average	Std dev	Average	Std dev
Per-link policing	1.60	0.12	0.10	0.01	0	0
Per-slice shaping w/o policing	1.30	0.08	0.11	0.02	0	0
Per-slice shaping with policing	1.33	0.10	0.10	0.01	0	0
No isolation	12.08	4.28	0.12	0.01	0.41	0.05
(Congestion-less)	1.31	0.15	0.12	0.02	0	0

V. CONCLUSION

Two methods for NRI for virtualization networks are proposed in this paper: per-slice shaping and per-link policing. The former enables NRI with 80–90% less queues compared to the per-link shaping. The latter enables less strict isolation between tens or hundreds of slices using only one queue. Evaluations of these methods show that the former performs slightly better in terms of delay and packet-drop ratio. Accordingly, the former with/without policing is effective for delay-sensitive services while the latter may be sufficiently used for the other types of services.

ACKNOWLEDGMENT

Part of research results described in this paper is an outcome of the Advanced Network Virtualization Platform (Project A) funded by National Institute of Information and Communications Technology.

REFERENCES

[GEN 09] The GENI Project, "Lifecycle of a GENI Experiment", GENI-SE-SY-TS-UC-LC-01.2, April 2009, http://groups.geni.-net/geni/attachment/wiki/ExperimentLifecycleDocument/-ExperimentLifeCycle-v01.2.pdf?format=raw .

[Nak 10a] Nakao, A., "Network Virtualization as Foundation for Enabling New Network Architectures and Applications, *IEICE Trans. Commun.*, Vol. E93-B, No. 3, pp. 454–457, March 2010.

[Nak 10b] Nakao, A., "Virtual Node Project — Virtualization Technology for Building New-Generation Networks", *NICT News*, No. 393, pp. 1–6, Jun 2010.

[Tur 07] Turner, J., Crowley, P., Dehart, J., Freestone, A., Heller, B., Kuhms, F., Kumar, S., Lockwood, J., Lu, J.,Wilson, M., Wiseman, C., and Zar, D., "Supercharging PlanetLab — High Performance, Multi-Application, Overlay Network Platform", *ACM SIGCOMM Computer Communication Review*, Vol. 37, No. 4, pp. 85–96, October 2007.