Autonomic Management Framework for Cloud-based Virtual Networks

Taesang Choi, Nodir Kodirov, Tae-Ho Lee, Doyeon Kim, Jaegi Lee
Electronics and Telecommunications Research Institute (ETRI)
Daejeon, Republic of Korea
{choits, nodir, kthlee, kdy, jkilee}@etri.re.kr

Abstract—Cloud computing promises to reshape the way IT service is produced and consumed by virtualizing computing resources. As if cloud providers offer virtualized CPU and storage resources via machine virtualization and distributed storage technologies respectively, the providers wish to offer to their customers' virtual networks running on the cloud. With cloud-based virtual networks (CVNs), cloud customers (often, corporate customers) can easily build new sites, effectively expanding their enterprise networks into the cloud and thus leveraging the entire benefits of cloud computing (agility, manageability, low cost, etc.). In this paper, we identify core requirements and objectives for managing CVNs. We then define a high-level autonomic CVN-management framework by demarcating between the customer and provider with CVN Management Interface (CMI). Finally we propose an open-flow based autonomic cloud data center management solution based on CVN management framework.

Key words; Cloud Computing; Cloud-based Vrituanl Networks, Autonomic Management;

I. INTRODUCTION

Cloud computing promises to reshape the way IT service is produced and consumed by virtualizing computing resources (CPU, storage, and network). As if cloud providers offer virtualized CPU and storage resources via machine virtualization and distributed storage technologies respectively, the providers wish to offer to their customers' virtual networks running on the cloud. With cloud-based virtual networks (CVNs), cloud customers (often, corporate customers) can easily build new sites, effectively expanding their enterprise networks into the cloud and thus leveraging the entire benefits of cloud computing (agility, manageability, low cost, etc.).

Computing (i.e., CPU cycles) and storage resources in the cloud are already virtualized. Virtual machines (VMs) and blob storage, respectively, are the widely-adopted notions that abstract computing and storage resources. These abstractions offer a simple and flexible interface between cloud customers and providers. More and more customers move towards the cloud. Customers need more than just a simple set of computing or storage resources.

To meet such needs, we need a network interconnecting virtual computing resources. For this, we define new networks called, "Cloud-based Virtual Networks (CVN)". A CVN is a network interconnecting virtual computing resources (e.g., virtual machines) built in the cloud. A CVN customer can dynamically create, modify, and delete its CVN. Further, a customer can interconnect its CVN with its own existing on-premise networks, easily expanding its own existing network into the cloud. A cloud provider can host multiple CVNs of different customers on a shared physical infrastructure.

Public cloud offerings have just begun to or will soon offer this CVN service. In late 2009, Amazon Web Service has beta-released VPC (Virtual Private Cloud)[1] to a small set of customers. VPC allows customers to establish a secure protected connectivity between the VM instances created in Amazon cloud (EC2 instances) with customers own existing networks. In addition, customers can assign IP addresses of their own choice to their EC2 instances, essentially making the EC2 instances equivalent to the machines present in the customers' enterprise networks. This allows VPC customers to manage the EC2 instances using their existing network-management tools, without requiring any change to accommodate the cloud environment. Other cloud providers, such as Google and Microsoft, are preparing for a similar feature.

Efficient management of CVN becomes an essential enabler for such a convenient and flexible service. The current management solutions for cloud services, however, are limited in the scope of configuration and monitoring of cloud computing resources (e.g., VMs, VM CPU & memory, and storage,) only, not including management of network in the cloud data centers.

In this paper, we examine the related work. Given the gap analysis results, we identify core requirements and objectives for managing CVNs. We then define a high-level autonomic CVN-management framework by demarcating between the customer and provider with CVN Management Interface (CMI). From the perspective of CVN users, CVNs are highly malleable---it is permissible, and often encouraged, for a CVN customer to adapt the configuration of its own CVN frequently and dynamically to the varying conditions (demands, cost, etc.). Doing so, however, requires a decision-support tool for the CVN users---more specifically corporate IT professionals. On the other hand, the CVN providers need a new networking technology to host multiple virtual networks on the same shared network infrastructure, without being constrained to the scalability and agility limitations of the existing networking technologies. For the proof of concept, we propose an open-flow based autonomic cloud data center management solution based on CVN management framework. Finally, we conclude our effort with potential future work.

II. RELATED WORK

A few existing technologies which described below might look promising at a glimpse, but none is suitable for the requirements of CVNs. The following are the such work.

- MPLS (Multi-Protocol Label Switching) VPN: whether L2 MPLS VPN[2] or L3 MPLS VPN[3] is used for our purpose, it might have scalability limitations. In practice, a number of virtual routing and forwarding table instances which can be supported are limited in terms of memory and performance resources.
- Ethernet VLAN (Virtual LAN)[4]: Limited number of VLANs that can coexist at the same time. Thus, there is limited size of VLANs. It also can't cope with frequent VLAN trunk-configuration updates.
- TRILL (Transparent Interconnection of Lots of Links)[5]: TRILL is an IETF draft. It avoids broadcast storm in a bridged L2 network, but this doesn't necessarily enable a huge L2 network which can a core requirement of CVN.

A few recent proposals address some of the CVN requirements as follows:

- SEATTLE (Scalable Ethernet Architecture for Large Enterprise)[6]: SEATTLE is a novel network architecture that achieves the best of the two most-popular networking technologies for enterprise/datacenter networks---the scalability of IP combined with the simplicity of Ethernet. SEATTLE provides plug-and-play functionality via flat addressing, while ensuring scalability and efficiency through shortest-path routing and hash-based resolution of host information. In contrast to previous work on identity-based routing, SEATTLE ensures path predictability and stability, and simplifies network management.
- IETF ARMD (Address Resolution for Massive number of hosts in Data center)[7]: This is a very recent IETF Working Group. Modern data center networks face a number of scale challenges. One such challenge for so-called "massive" data center networks is address resolution, such as is provided by ARP(Address Resolution Protocol) and/or ND(Neighbor Discovery). The working documents in the group address the problem of address resolution in massive data centers.

The technologies described above address some partial solutions for CVN but improvements, appropriate integration, and introduction of new mechanisms are needed. Especially, the management aspects have to be considered together with CVN issues not separately. In this paper, we propose such a solution to fill up the gaps.

III. CVN CORE PROPERTIES AND FEATURES

A. What is CVN?

CVNs are different from VPNs (Virtual Private Networks), which are already widely available today. A VPN is a network that uses a public telecommunication infrastructure, such as the Internet, to provide remote sites, offices, or individual users with secure access to their enterprise network. In contrast to VPNs, CVNs have the following unique properties:

- A CVN is a network composed of a multitude of end points (virtual machines). A CVN is bigger in size and richer in topology than just a point-to-point interconnection established between two sites of a customer through a public network.
- A CVN can be purely logical. CVN users might not manage any of the network devices and links between the end points the CVN---CVN users might not even be aware of those components.
- CVN customers can offer various networking policies among the end points, but do not implement those policies.

B. Core features of CVNs

A CVN user can use its own CVN as if it is physically built in its own premise.

This means that a customer can do the following tasks:

- Assign its own IP addresses and names: Today cloud providers assign their own choice of IP addresses to customers' VM instances or VM endpoints (i.e., virtual NICs of customer VMs). Regardless of whether these provider IP addresses are private or public, the IP addresses can collide with the IP addresses given to the customers' existing machines residing in the customers' enterprise networks. When connecting the resources in the cloud and the resources in enterprise networks through VPNs, address collision results in critical connectivity failure.
- Continue to use existing network-management solutions;
- Apply its own networking policies;

IV. CVN MANAGEMENT REQUIREMENTS AND OBJECTIVES

A CVN technology needs to meet the following generic and management requirements. They are the complete list but core requirements.

- Reachability isolation: VMs in a CVN must see the traffic from other VMs of the same CVN only, irrespective of other CVNs' configuration (IP addresses, locations, communication patterns, etc.).
- Performance isolation: One CVN's networking performance should be immune to other CVNs' networking activities.

- Scalability: CVN should support O(10^4) tenants, O(10^3) physical network devices, O(10^5) virtual machines (hence end points), etc.
- Cope with highly-frequent network changes: There are O(10^4) network-change events per day for the large scale CVNs. Thus it should be able to manage such frequency of network changes.
- Support various policies: CVN should support various policies flexibly.
- Self-configuration and automated network bootstrapping and management capabilities should be supported
- Support for hierarchical service-offering model: A and B are customers of Cloud service P, and they offer services to other customers C, D, E … of P. This chain can go deep in any arbitrary length.
- Enable various networking functions, in addition to basic connectivity; various networking functions should be supported.
- Performance: CVN should support performance of CVN infrastructure to provide requested service quality.
- Various middle-box functions: CVN should support various middle-box functions for the virtual network clouds environment.

V. CVN Management Framework

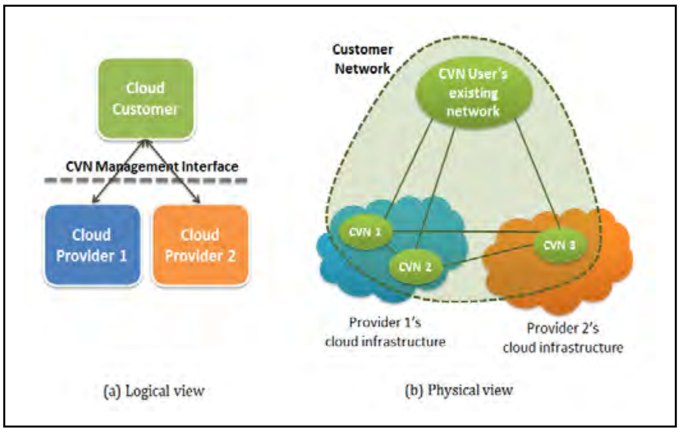

Fig.1. CVN Management Framework High-level View

Fig. 1 illustrates the CVN management framework at a high level. A CVN customer can interact with multiple CVN providers and build its own CVNs (CVN1, CVN2, and CVN3) in the diagram showing the physical view. The customer can connect the CVNs to its own existing network sites, expanding its enterprise network into the cloud.

A cloud customer interacts with multiple cloud-service providers to create and manage CVNs. The CVN Management Interface (CMI) offers well-defined interface between CVN customers and providers. With CVNs, a customer can easily and dynamically expand its network into the cloud, essentially building an enterprise network composed of multiple sites.

CVN Management Interface (CMI) defines the way CVN providers and customers interact with each other. That is, a CVN user can manage its own CVNs through the CMI. Through the CMI, a CVN provider receives users' requests and returns the results of the requests.

Specifically CMI exposes the following core functions to CVN customers.

- Life-cycle management: Create and deploy a new CVN. Modifying an existing CVN, and Delete an existing CVN.
- Operation management: Start or stop a CVN.
- Connect to existing network: Establish a protected connectivity (e.g., VPNs) between CVNs or between a CVN and a customer's own networks.
- Property update: Modify various properties of a CVN, such as size, addresses, performance specification, and policies, etc.
- Monitoring: Retrieve various kinds of operational information about a CVN such as CPU utilization, memory usage, storage usage, a number of active VM instances, VM network interface statistics, CVN flow numbers, etc.

While CMI offers "network-management" interface between CVN customers and providers, the set of network-management features CMI implements are rather different from the features that conventional network management framework offers owing to the unique properties of the cloud.

The following operations are important in the conventional network-management framework, but are not directly relevant to the CVN management framework.

- Device management
- Link management
- Traffic engineering
- Device-level failure management
- Configuration management
- Physical capacity management

Note that these management tasks are still very important for a CVN provider to manage its physical network infrastructure which CVN customers share. In the CVN management framework, however, a CVN provider is unlikely to expose individual network elements (devices, links, etc.) to CVN customers. Further, in the cloud, individual component failures are masked by running multiple identical components in a scaled-out fashion. This eliminates the need for component-level failure management as well.

On the other hand, the following management tasks are relevant to the CVN framework.

- Service-level failure management
- Performance management
- Virtual capacity management
- Security management
- Usage and cost management (Accounting)
- Traffic monitoring (passive and active)

Fig. 2 shows the high-level task flow for CVN management. To create a CVN, a customer first identifies the needs and constraints for the CVN, such as the CVN's size (# of member VMs, IP-address prefixes to use for the VMs, connectivity to the customer's existing enterprise network or other CVNs, routing information, host names, etc.). Then the customer translates the needs and constraints into a specific CVN configuration instance and supplies it to the CVN provider, issuing a "CVN-creation" request. Upon receiving the CVN configuration instance, the CVN provider first validates the configuration. When the configuration passes the validation step, the provider runs a resource-allocation algorithm and chooses a right set of resources that can accommodate the "CVN-creation" request. Finally, the provider deploys and provisions all the needed resources---instantiates VMs, configures the virtual network interfaces of the VMs, assigns IP addresses, sets up the correct performance & reachability policies, etc.).

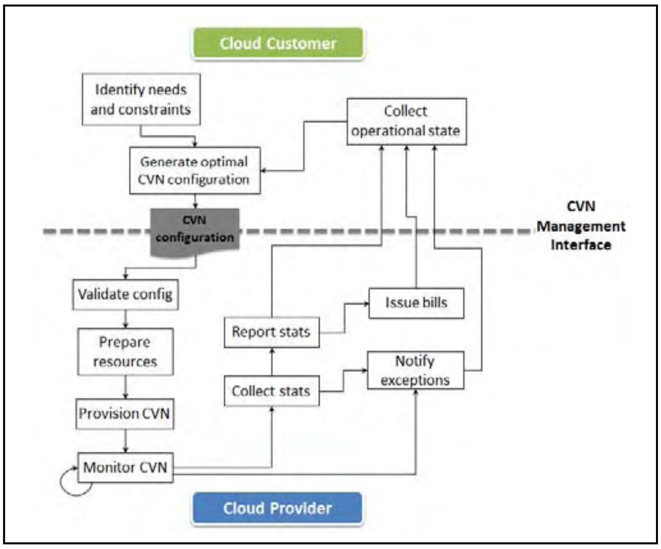

Fig. 2. High-level task flow for CVN management.

Once a CVN is provisioned, the provider continuously monitors the CVN's state and collects appropriate operational logs needed for charging and reporting. The CMI offers well-defined channels through which the provider periodically reports the CVN's operational state to the customer. Upon an occasional exception, the provider immediately notifies the state to the customer. Additionally, the CVN customer also polls the CVN's state on a demand basis.

The type of information a provider offers to a customer include the followings but not limited to:

- CVN's operational state: Up-time, etc.
- Usage: Traffic volume (per class), storage footprint, etc.
- Performance: Latency, loss rate, etc.
- Failure notification
- Exceptions: Policy violation incidents, customer-defined alarms, etc.

In order to meet various (cost, performance, availability, etc.) objectives, the customer can use this information supplied by the provider and dynamically adjust the CVN's configuration. In this case, the customer represents autonomic cloud service management application which interacts with CVN management system.

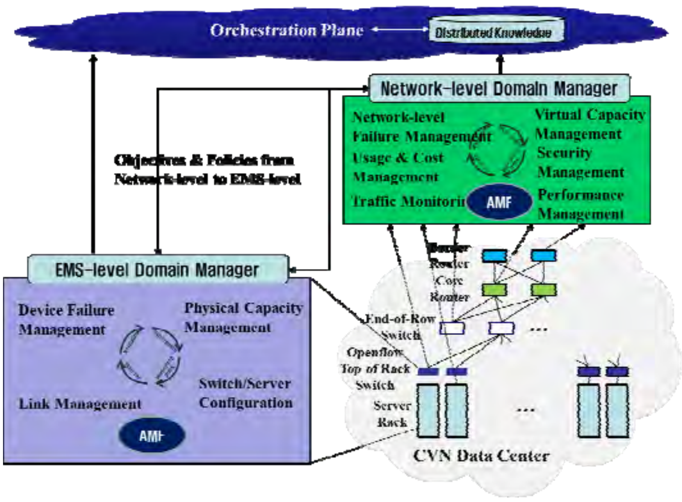

Fig. 3. Autonomic CVN management framework

Fig. 3 shows how autonomicity can be enabled for the CVN management. It illustrates a CVN data center autonomic management scenario. EMS(Element Management System)-level domain manager has an autonomic management function (AMF) which implements a control loop for autonomic element-level management capabilities such as switch/server configuration, link management, device failure management, and physical capacity management. Network-level domain manager also has its own AMF which implements a control loop for autonomic network-level management capabilities such as traffic monitoring, performance management, security management, network-level failure management, usage & cost management, and virtual capacity management. Two domain mangers cooperate to enable autonomic management of a CVN data center. Network-level domain manager decides autonomic management policies and push them to EMS-level manager for the enforcement. EMS-level domain manager also provides its own management information to the network-level manager for further analysis and decision making. Both managers send or retrieve rules that requires cross level coordination via Orchestration plane. It contains common knowledge repository in the form of ontology and associated cognition and inference engine.

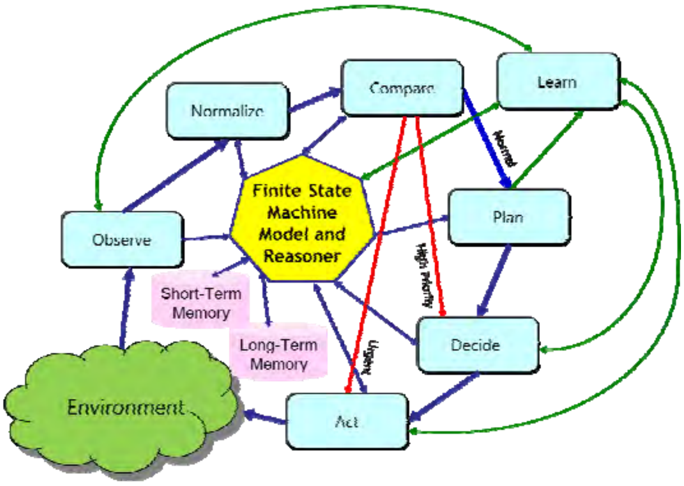

Fig. 4. Cognition Model for Autonomic Management

Autonomic management capability is supported by the cognition model illustrated in the fig. 4. This model is extended by the FOCALE initial cognition model[8]. It supports Minsky's three cognition processes[9]: reflective, deliberate, and reactive. Since all processes use the finite state machine and reasoning, the model can realize when an event or set of events has been previously encountered. Such results are stored in short-term memory. This reactive mechanism enables much of the computationally intensive portions of the control loop to be bypassed, producing the two "shortcuts" labeled "high priority" and "urgent". The deliberative process is embodied in the set of bold arrows, which takes the Observe-Normalize-Compare-Plan-Decide-Act path. This uses long-term memory to store how goals are met on a context-specific basis. The reflective process examines the different conclusions made by the set of deliberative processes being used, and tries to predict the best set of actions that will maximize the goals being addressed by the system. This process uses semantic analysis to understand why a particular context was entered and why a context change occurred to help predict how to more easily and efficiently. This model is used to realize an autonomic management control loop of a domain manager.

VI. PROTOTYPE IMPLEMENTATION OF CVN MANAGEMENT FRAMEWORK

To verify the feasibility of autonomic CVN management framework, we have implemented a proof of concept system. Fig. 5 shows our PoC system architecture, namely OpenFlow-based autonomic CVN management architecture. In this design, we introduced additional OpenFlow-capable switches. The CVN Manager communicates with the OpenFlow switches and configures them to manage CVNs. As long as the OpenFlow switches generate Ethernet or IP packets, the other network devices do not require any special functions. Any traffic originating from the VMs are always forwarded to an OpenFlow switch. Upon receiving the first packet of a flow, the OpenFlow switch communicates with the CVN manager on demand and obtains the appropriate control-plane information needed to be virtualized networking (encapsulation/ decapsulation, ID-to-locator mappings). CMI can be provided to the customer through the web interface.

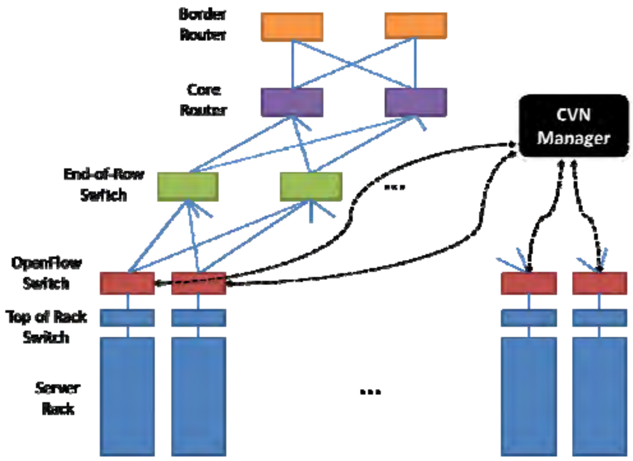

Fig. 5. Proof of Concept System for Autonomic CVN Management

For the management, a CVN customer supplies the configuration for its CVN through the CMI, which includes the following information.

- Size: This specifies the size of a CVN in terms of the number of member end points (virtual machines).
- IP addresses: This specifies the IP addresses given to the member end points. Depending on provider's approach, customer can specify the addresses either for each individual end point of a group of end point sharing the same attributes.
- Names: This specifies the machine names (FQDNs) associated to the member end points. Some providers might not allow customers to specify names for individual end points. Instead, they can allow customers to specify their domain zones (postfixes for the FQDNs).
- Location: This specifies the location of the CVN members. To ensure high availability and short latency to customers, a CVN provider often runs multiple data centers at geo-distributed locations. Some provider might allow customers to choose data centers (locations) in which the customers' CVNs are created.
- Credentials: This specifies the credentials needed to establish trusted connectivity with other members in the CVN or any other machines in the Internet.
- Networking policies (access-control policies, QoS, security, routing, etc.): This specifies various networking policies that a customer wants to apply to the CVN.

Determining each of these configuration settings requires clear understanding of the customer's needs, as well as the current operational state of the CVN---this can be a highly challenging task. Hence, customers are likely to leverage a decision-support tool for this CVN management task. Meanwhile, perhaps the most interesting and appealing aspect of CVN is flexibility and agility; a customer can frequently stop and restart its CVNs, change the configuration of the CVNs, including the size and locations. This unique feature of CVN allows customers to run efficiently their computing and networking resources to minimize cost, ensure performance, and enforce networking policies (see Table I).

TABLE I. CVN OBJECTIVES

Objective	Minimize cost
Constraints	Meet performance guideline Enforce access-control policies Conform regulatory guidelines

Also we have developed virtual programmable switch which include Openflow capability based on TilePro64 multi-core network processor. The design schematic diagram of forwarding plane is illustrated in Fig. 6. The TilPro64 is a PCI express board with multi-core (supports 64 cores) chip. It is installed in the PC server and host operating system is Linux. Board consists of CEA(Control Element Agent), MEA(Management Element Agent), virtual switch and multiple forwarding planes as shown in the diagram. This board provides forwarding plane capability only and control plane exists separately in the host PC OS. CEA and MEA takes 2 cores, virtual switch takes another 2 cores, remaining 60 cores can be used for forwarding purposes. In our current implementation, it supports upto 54 forwarding elements. Each represents a virtual router forwarding plane. Openflow is used for control plane capability. Thus, CEA and MEA implements Openflow protocol of router/switch agent part and Openflow server and controller is implemented in the host PC OS. Each core supports 900MHz and can handle upto 1Gbps in best case.

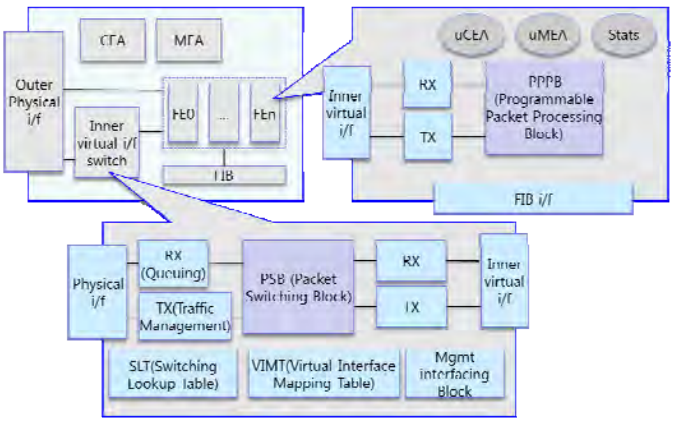

Fig. 6 Openflow switch forwarding plane design schema

In a nutshell, our prototype system enables customer to create CVN, specify/make topologies, allocate IP addresses, names, and define various networking policies between end-points. Eucalyptus - Open Source cloud environment [10] provides basic VM management features through plug-in running on web browser. CVN interface is provided in the similar way, where customer is able to identify need and constraints. Based on the provided requirements, CVN module generates optimal CVN configuration and presents it to the user for the verification. This is an optional step, where experienced customer can check/modify topology and connections between VMs. Topology Management Tool is used to visualize CVN configuration and provide user convenient web-based GUI interface to modify connections and customize CVN topology. Final CVN configuration is validated and passed to prepare resources. Once CVN is provisioned it comes to monitor loop, which feeds data to collect stats, report stats, notify exceptions and issue bills. Lastly, CVN operational state is collected and used to generate evolved optimal CVN configuration. These steps such as collection, normalization, analysis, making decision, and action of CVN information are conducted autonomically based on the cognition model described in the section V and Fig. 4. Through above autonomic CVN configuration and feedback steps CVN management framework provides to customer service-level failure management, performance management, security management, accounting and traffic monitoring.

VII. CONCLUSION AND FUTURE WORK

In this paper, we identified core requirements and objectives for managing CVNs. We then defined a high-level autonomic CVN-management framework by demarcating between the customer and provider with CVN Management Interface (CMI). Finally we proposed an open-flow based autonomic cloud data center management solution based on CVN management framework.

The work proposed here is still in the early stage that we are currently verifying the functionalities we designed and implemented in our PoC system for its feasibility. We are also conducting various performance evaluations for system's scalability and reliability. We will incorporate these analysis results in our future version of paper for the completeness.

ACKNOWLEDGMENT

This research was supported by the KCC(Korea Communicatio ns Commission), Korea, under the "Novel Study on Highly Manageable Network and Service Architecture for New Generation" support program supervised by the KCA(Korea C ommunications Agency)"(KCA-2011-10921-05003)

REFERENCES

[1] Amazon VPS, http://aws.amazon.com/ec2/
[2] L2 MPLS VPN, http://tools.ietf.org/wg/l2vpn/
[3] L3 MPLS VPN, http://tools.ietf.org/wg/l3vpn/
[4] IEEE 802.1q
[5] TRILL, http://tools.ietf.org/wg/trill/charters
[6] C. Kim, M. Caesar, and J. Rexford, "SEATTLE: A Scalable Ethernet Architecture for Large Enterprises," ACM SIGCOMM, August 2008.
[7] ARMD, http://www.spinics.net/lists/ietf-ann/ msg59787.html
[8] Focale J. Strassner, N. Agoulmine, E. Lehtihet, "FOCALE – A Novel Autonomic Networking

Architecture", ITSSA Journal, Vol. 3, No. 1, May 2007, pages 64-79, ISSN 1751-1461

[9] M. Minsky, "The Society of Mind", Simon and Schuster, New York, 1988, ISBM 0671657135

[10] Eucalyptus, http://www.eucalyptus.com/