A Node Plug-in Architecture for Evolving Network Virtualization Nodes

Yasusi Kanada
Hitachi, Ltd.
Outline

► What is VNode?
■ VNode is a physical node with network virtualization function.

► Evolution of VNode
■ VNode allows independent evolution of components of VNode.

► Proposal
■ An architecture and a method for evolving the components in two steps using plug-ins are proposed.

► Prototyping and evaluation
What is VNode?

We are developing *VNode* and *VNode Platform* in a collaborative project directed by Prof. Aki Nakao.

VNode Platform is a network-virtualization platform.
- On this platform, multiple developers can create and use slices (virtual networks) concurrently and independently.

VNode is physical node with network-virtualization functions.
Components of VNode

- VNode consists of three components:
  - **Programmer**: a *deeply-programmable* computational component (software & hardware).
    - Deeply-programmable: Data-plane is programmable.
  - **Redirector**: a networking component (software & hardware).
  - **VNode manager**: a control component.
Programmer and redirector may evolve by using *new node hardware or software, or new functions*. 
- **New hardware/software**: New network processors, GPGPUs, New types of VMs, etc. 
- **New functions**: A new protocol stack, etc.

Programmer and redirector can evolve independently because they are *modular*. 
- The interfaces can be clearly defined and work efficiently.

However, no method for this evolution has been available.
Proposed Method: Two-step Evolution of VNode

► Step 1: To develop new subcomponents as plug-ins and to connect them to the original components.
■ The subcomponents (software & hardware) can be dynamically evolving without interference to existing slices.

► Step 2: To merge the plug-ins into the components.
■ The merged components are static and stable.
Plug-in Architecture for Step 1

Data plug-in and Control plug-in
- A new data-plane function is implemented by a data plug-in.
- A data plug-in requires new control functions, which are implemented by a control plug-in.

Plug-ins may be placed at logically and physically distant location from VNode.
Plug-in Interfaces for Step 1

- **Open VNode plug-in interfaces (OVPIs)** are defined.
- Two types of OVPIs: C-plane and D-plane interfaces.
Design of OVPI for Control Plug-ins

► Components of Control OVPI

■ Plug-in host name or address
  ● It specifies the host that contains the plug-in.
  ● It is usually a domain name or an IP address.

■ Plug-in identifier
  ● It specifies a plug-in in the host.

■ Parameters
  ● They specify control information including a slice identifiers.

► Examples

■ CLI of the plug-in host:
  add_link vlan=id esmac=p1 edmac=p2 ismac=p3.

■ XML-RPC (connected to the host):
  <methodCall>
    <methodName>create-slivers</methodName>
    <params>
      <param>
        <linkSliver type="link" name="virtual-link-1">
          <vports>
            <vport name="vport0">
              <params>
                <param key="controller" value="plug-in-0-addr" />
                <param key="port" value="data-plug-in-0-port"/>
              </params>
            </vport>
            …
          </vports>
        </linkSliver>
      </param>
      <params>
        <param key="ExtensionName" value="vlan_link"/>
      </params>
    </params>
  </methodCall>...
Components of Data OVPI

- **Plug-in channel tag**
  - A host and a plug-in are not specified separately.
  - A tag (such as a VLAN identifier) can specify a collection of plug-ins.

- **Parameters**
  - They are protocol parameters such as addresses.
  - Some of them can be used for identifying the slice.

Examples

- A VLAN can be used for the data-plane protocol.
  - Plug-in channel tag: A VLAN identifier.
  - Parameters: Source and destination MAC addresses.

- GRE/IP can be used for the data-plane protocol.
  - Plug-in tag: A key in the GRE header
  - Parameters: Addresses in the IP header.
Prototyping & Evaluation: Architecture

► Data plug-in and OVPI
- VLAN is used for data OVPI.
- Data plug-in is a network processor (NP) board with Cavium Octeon with a program (software & hardware).

► Control plug-in and OVPI
- CLI is used for control OVPI.
- Control plug-in is a Linux PC with a C program.

[Diagram of VNode with VLAN, Control-plane component, Data-plane component, CLI, Plug-ins on a PC, Network processor board with 750-MHz Cavium Octeon (Data plug-in)]
Prototyping & Evaluation: Two Sets of Plug-ins

► 1: Network accommodation
   ■ It connects a slice to an external network through a VLAN.
   ■ This is a re-implementation of a function supported by the platform.

► 2: New type of virtual link
   ■ VNode platform only have a type of GRE-based virtual links.
   ■ A type of VLAN-based virtual links is implemented by using plug-ins.

(to be continued …)
Prototype & Evaluation: VLAN-based Virtual Link

► Prototype development
- Two evolving VNodes with plug-ins were developed.
- A new type of virtual links was implemented on the VNodes.
  - Controllers in VNodes “tunnels” control information of data plug-ins.

► Evaluation
- A slice with two virtual nodes and a VLAN-based virtual link between the nodes is created.
- The virtual link worked correctly and the throughput was 9 Gbps.
Summary and Future Work

► A method for evolving VNodes by two steps was proposed.
  ■ Step 1: Data & control plug-ins, connected to a VNode by OVPIs, are used.
  ■ Step 2: Plug-ins are merged into VNode.

► Programmer and redirector can be independently evolved using this method.
  ■ Only redirector plug-ins has been tested.

► Prototypes of OVPIs and plug-ins were developed and evaluated.
  ■ The evolving redirector can support new types of network accommodation functions and new types of virtual links.
  ■ The throughput is close to a wire rate of 10 Gbps.

► Future work
  ■ To implement programmer plug-ins.
  ■ To implement more new types of virtual links (redirector plug-ins).
  ■ To apply this method to VNodes in Japan-wide testbed, JGN-X.