Demonstrating workload isolation in IBM PureApplication System

Hendrik van Run
Certified Consulting IT Specialist
IBM

Venkata (Vishy) Gadepalli (vgadepal@us.ibm.com)
Senior Management Consultant
IBM

Bobby Woolf
Certified Consulting IT Specialist
IBM

Learn how to create two or more application runtime environments in IBM® PureApplication™ System that are highly isolated from each other so that the workloads in one environment cannot interfere with those in another. This article shows how to create them and demonstrates that they are indeed isolated.

Introduction

In IBM PureApplication System W1500 V1.0 (hereafter called PureApplication System), applications run in the cloud as workloads that share the system’s resources, such as CPU, memory, and networking. However, some workloads need to be isolated from each other so that activity in one workload cannot interfere with that in another workload. As an administrator setting up a PureApplication System, how can you create two or more runtime environments that are highly isolated from each other so that the workloads in one environment cannot interfere with those in another? Furthermore, can we demonstrate that these environments and their workloads are indeed isolated from each other?

For example, an application development team may wish to divide their PureApplication System into two runtime environments:

- Production for applications to be used for running the business
- Test for applications being tested by development
How can PureApplication System isolate the production workloads from the test workloads so that production gets dedicated resources that cannot be consumed by the test workloads, and problems in the test workloads cannot interfere with the production workloads?

To understand how to create isolated application runtime environments on PureApplication System and demonstrate their isolation, we need to first understand what it means for two runtime environments to be isolated, the features PureApplication System provides for creating runtime environments, and how to use those features to create runtime environments with the isolation we desire. We can then demonstrate that this isolation actually works. This article draws from the concepts explained in Managing application runtime environments in IBM PureApplication System as well as Navigating the IBM cloud, Part 2: Understanding virtual system patterns in IBM Workload Deployer and IBM PureApplication Systems.

**Isolated workload environments**

To understand how to create isolated application runtime environments to isolate workloads on PureApplication System, let us first review what it means for runtime environments to run workloads in isolation.

The article Managing application runtime environments in IBM PureApplication System explains the relationship between resource isolation and resource sharing:

- **Isolation**: Workloads can be isolated from each other so that problems in one workload (such as a runaway process that consumes all CPU or memory) do not affect the others.
- **Sharing**: Workloads draw from common pools of resources so that some may use more resources when others do not need them.

The article goes on to explain that resources can be isolated in two main aspects:

- **Computational isolation**: Groups of CPU and memory capacity are separated from each other. When two workloads execute with computational isolation, one workload's consumption of those resources does not affect the other workload.
- **Network isolation**: Communication flows between computational resources via separate connections.

It also explains that the isolation can be physical or logical. Physical meaning that the resources are separate hardware, and logical meaning that hardware management software makes a single set of hardware work like multiple, separate sets of hardware.

In this article, we want the workloads to be completely isolated, so we do not want any resource sharing between them. Each workload should be assigned its own dedicated resources.

Whether you are using traditional computer hardware or PureApplication System, it is easier to set up the hardware to achieve dedicated computational resources than dedicated network resources. To isolate two applications, if you have one computer connected to your enterprise network, your only option is virtual computational isolation, such as by using virtual machines and a hypervisor, and logical network isolation, such as by using virtual local area networks (VLANs) defined in the
enterprise network. If you have two computers and can connect them to the enterprise network via separate sets of network equipment, then you can achieve physical computational isolation by putting each workload on a separate computer and physical network isolation by using each computer's separate network connection. Separate network connections are unusual, though, because it is simpler to connect the computers all as part of the single enterprise network, and so VLANs are a much more common means of network isolation. Therefore, to isolate two applications, typically the approach is separate computational hardware and VLANs to isolate the network traffic.

**PureApplication System features**

Before we show how to create two isolated workload runtime environments in PureApplication System and demonstrate that they are indeed isolated, let us first review the features in PureApplication System used to define runtime environments and to deploy applications as workloads.

**Runtime environments**

As explained in the article *Managing application runtime environments in IBM PureApplication System*, the computational hardware in a PureApplication System is organized into one or more cloud groups. A *cloud group* is a logical computer composed of one or more compute nodes and at least one IP group. A *compute node* is a compact but powerful computer, similar to a blade server. It contains CPU and RAM and runs a hypervisor that hosts virtual machines, as shown in Figure 1. An *IP group* is a set of IP addresses and is associated with a VLAN connected to the enterprise network. By combining compute nodes and IP groups, a cloud group is everything needed to run virtual machines - it has hardware for them to run in and IP addresses to assign to the virtual machines.

**Figure 1. Hypervisor stack in a compute node**

To better understand how a cloud group uses hardware resources, here are some details about cloud groups, the relationship with compute nodes, VLANs, and resources that need to be defined on the network.
There are a few constraints in the relationship between cloud groups and compute nodes. When a cloud group is created, it has no compute nodes assigned to it. For a cloud group to function, it needs to contain at least one compute node (and at least one IP group). While a cloud group can function properly with just one compute node, ideally each cloud group should contain at least two compute nodes so that if one fails, its virtual machines can failover to the other. A particular compute node can only belong to a single cloud group. To reassign a compute node from one cloud group to another, you must remove it from the first cloud group before adding it to the second one.

PureApplication System uses VLANs in two different ways, either as a management VLAN or an application VLAN. Each cloud group also requires a management VLAN ID. A cloud group contains an internal VLAN it uses to manage its virtual machines and other processes. These VLANs do not need any IP addresses assigned to them, but the network administrator needs to reserve the IDs so that the enterprise network does not contain any other VLANs with the same ID.

The PureApplication System administrator will need some information from the network administrator to configure the IP groups properly. A VLAN is needed for each set of applications whose network traffic should be separated from that of the other applications. Enough IP addresses need to be assigned to each IP Group such that each of the virtual machines in each of the applications can have a unique IP address. Some planning is needed to think about how many:

- Applications that will be deployed.
- Separate virtual machines that will be needed for them to run.
- Separate VLANs that will be needed for them to communicate.

Then the network administrator needs to actually provision the VLANs in the network with unique IDs and a sufficiently large set of IP addresses. Those VLANs also need to be physically connected to and configured on PureApplication System. Typically, a number of VLANs are configured during the initial setup of the system. Additional VLANs can also be added later.

**Virtual system patterns**

PureApplication System allows users to create patterns to define specific applications. Those patterns allow for a consistent and repeatable deployment process. A user deploys a pattern using an environment profile. An *environment profile* defines policies for how patterns are deployed to one or more cloud groups and how their instances run in the cloud group. To deploy a pattern, a user selects a profile for performing the deployment, which in turn specifies the cloud groups the deployer can deploy patterns to. Once deployed, you refer to each running pattern instance as a *workload*.

A common type of pattern in PureApplication System is a *virtual system pattern*. This type encapsulates standard middleware product installs (such as WebSphere® Application Server and DB2®) as deployable patterns, customized through the use of one or more script packages. Typically, deploying such a pattern lays down both your application and the middleware needed to run it. For more details about what virtual system patterns are and how they work, see the article *Navigating the IBM cloud, Part 2: Understanding virtual system patterns in IBM Workload Deployer and IBM PureApplication Systems*. 

Demonstrating workload isolation in IBM PureApplication System
Demonstrate isolated workload environments

To demonstrate two runtime environments that isolate their workloads from each other, we will perform these steps:

1. **Create the demonstration environment**: The demonstration environment will consist of two runtime environments that each runs its workloads isolated from the other.

2. **Deploy the demonstration workloads**: Each runtime environment will host a workload.

3. **Demonstrate the runtime isolation**: To demonstrate isolation between the two environments, we will run two tests:
   - We will show the network route between the two workloads, showing that they are isolated from each other on the system and only connected to each other by the network external to the system.
   - We will cause the workload in one runtime environment to consume a great deal of the environment's resources and show that the other environment and its workload are not affected.

Create the demonstration environments

The first step in demonstrating isolation between runtime environments is to create two separate runtime environments that will each run its workloads isolated from the other.

As explained earlier, a runtime environment in PureApplication System is a cloud group. Therefore, to create two separate runtime environments, we will create two cloud groups. A cloud group consists of one or more compute nodes and at least one IP group. Compute nodes are hardware resources that are part of PureApplication System, so we do not need to create them because they already exist inside the system. IP groups, on the other hand, just represent configuration data so we will have to create those. We will also create two environment profiles for deploying patterns.

Before you start, you need to ensure that the following hardware and network resources are available:

- **User**: There is one with full permissions on the PureApplication System.
- **Compute nodes**: You will need two compute nodes. They need to be unassigned, that is, not part of another cloud group, so you can assign them to your cloud groups. In this example, you will use two named "SN#23ZXL15" and "SN#23ZXL05".
- **Networks**: To demonstrate network isolation, you will need two application VLANs, one for each set of applications that is supposed to be isolated from the other. Two VLANs can be defined on a single network, which creates logical isolation. For this demonstration, you will define the two VLANs on two separate enterprise networks so that the network isolation will be not just logical but also physical. We assume that those networks are connected by their respective gateways. Greater network isolation can easily be achieved by reconfiguring the gateways or adding a firewall.

You will be working with two different VLANs each defined on a different network, as shown in Figure 2.
Figure 2. PureApplication System connectivity to enterprise networks

The details for the networks you are working with are listed in Table 1.

Table 1. Network properties and application VLANs

<table>
<thead>
<tr>
<th>Network Property</th>
<th>Network A</th>
<th>Network B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network</td>
<td>172.19.75.0</td>
<td>172.19.76.0</td>
</tr>
<tr>
<td>Subnet mask</td>
<td>255.255.255.0</td>
<td>255.255.255.0</td>
</tr>
<tr>
<td>Gateway</td>
<td>172.19.75.1</td>
<td>172.19.76.1</td>
</tr>
<tr>
<td>Primary DNS</td>
<td>172.19.64.2</td>
<td>172.19.64.2</td>
</tr>
<tr>
<td>Secondary DNS</td>
<td>172.19.64.4</td>
<td>172.19.64.4</td>
</tr>
<tr>
<td>VLAN</td>
<td>100</td>
<td>200</td>
</tr>
</tbody>
</table>

The demonstration environment you will create will consist of:

- **IP groups**: You will create two IP groups:
  - **IP Group A**: This one will use application VLAN 100.
  - **IP Group B**: This one will use application VLAN 200.

- **Cloud groups**: You will create two cloud groups:
  - **Cloud Group 1**: This one will contain compute node SN#23ZXL15 and IP Group A.
  - **Cloud Group 2**: This one will contain compute node SN#23ZXL05 and IP Group B.

- **Environment profiles**: You will create two environment profiles for deploying patterns to the two cloud groups:
  - **EnvProf-1**: This one will deploy patterns to Cloud Group 1.
  - **EnvProf-2**: This one will deploy patterns to Cloud Group 2.

The demonstration environment is shown in Figure 3.
Demonstrating workload isolation in IBM PureApplication System

Figure 3. Demonstration environment

Compute nodes

PureApplication System includes a set of compute nodes. The number of compute nodes in a system depends on the size of the system. For example, a small W1500 system contains six compute nodes. For this demonstration, you will only use two compute nodes to create the two cloud groups.

IP groups

You need to create two IP groups for the two networks that you are using for deployments. Each IP group will be configured with a set of predefined IP addresses. Table 2 shows the properties you will use for each of the IP groups. Note that these network settings are provided by your network administrator. In this example, you assign the full range of IP addresses to each IP Group.

Table 2. IP group properties

<table>
<thead>
<tr>
<th>IP Group Property</th>
<th>IP Group A</th>
<th>IP Group B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network address</td>
<td>172.19.75.0</td>
<td>172.19.76.0</td>
</tr>
<tr>
<td>Subnet/netmask</td>
<td>255.255.255.0</td>
<td>255.255.255.0</td>
</tr>
<tr>
<td>Gateway</td>
<td>172.19.75.1</td>
<td>172.19.76.1</td>
</tr>
<tr>
<td>Primary DNS</td>
<td>172.19.64.2</td>
<td>172.19.64.2</td>
</tr>
<tr>
<td>Secondary DNS</td>
<td>172.19.64.4</td>
<td>172.19.64.4</td>
</tr>
<tr>
<td>VLAN ID</td>
<td>100</td>
<td>200</td>
</tr>
<tr>
<td>IP range start</td>
<td>172.19.75.2</td>
<td>172.19.76.2</td>
</tr>
<tr>
<td>IP range end</td>
<td>172.19.75.254</td>
<td>172.19.76.254</td>
</tr>
</tbody>
</table>

To create an IP group, go to the IP groups page in the PureApplication System console. In the console, select the System Console tab and select Cloud > IP Groups. This page displays the list of existing IP Groups. To create a new one, press the green plus sign icon (+), which opens the Create IP group dialog. Populate the dialog with the values for IP Group A listed in Table 2, as shown in Figure 4, and press OK.
Now you need to add a set of IP addresses to each of them. You assign 253 addresses within the network to the IP group. Note that you do not need that many IP addresses for this demonstration.

To add IP addresses to an existing IP group, go to the IP groups page again (select System Console > Cloud > IP Groups) and select IP Group A. Enter the start and end addresses and press Add, as shown in Figure 5.

Create "IP Group B" by repeating the above steps.

**Cloud groups**

The cloud groups page in the administrative console shows the existing cloud groups. To view that page, go to the System Console tab and select Cloud > Cloud Groups.

A newly initialized PureApplication System contains a default cloud group named "Shared." From the cloud groups page, select an existing cloud group and examine its properties:

- **Compute nodes:** This is the list of compute nodes assigned to the cloud group.
• **IP groups**: This is the list of IP groups assigned to the cloud group.
• **Management VLAN ID**: Each cloud group needs a separate management VLAN. This is an internal VLAN which cannot be accessed from outside the system. However, the VLAN ID needs to be unique within PureApplication System.

Figure 6 shows these properties for the default cloud group "Shared".

**Figure 6. Cloud group attributes**

![Cloud group attributes diagram]

For the demonstration, you will need two cloud groups. Table 3 shows the properties for the two cloud groups.

**Table 3. Cloud group properties**

<table>
<thead>
<tr>
<th>Cloud Group Property</th>
<th>Cloud Group 1</th>
<th>Cloud Group 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Average</td>
<td>Average</td>
</tr>
<tr>
<td>Management VLAN ID</td>
<td>1000</td>
<td>2000</td>
</tr>
<tr>
<td>IP group</td>
<td>IP Group A</td>
<td>IP Group B</td>
</tr>
</tbody>
</table>

To create a cloud group, go to the cloud groups page (select **System Console > Cloud > IP Groups**) and press the green plus sign icon (+), which opens the Create cloud group dialog.
Populate the dialog with the values for Cloud Group 1 listed in Table 3, as shown in Figure 7, and press OK. Repeat these steps for the other cloud group, Cloud Group 2.

**Figure 7. Create cloud group dialog**

Describe the cloud that you want to create.

- **Name:** Cloud Group 1
- **Description:**
- **Type:** Average
- **Management VLAN ID:** 1000

Do not use the following VLAN IDs that are already in use:

1, 12, 23, 32, 123, 232, 323, 666, 777, 888, 999, 1111, 2222, 3333, 32, 01, 4031, 4093, 4094, 4095

With both cloud groups created, you need to assign each one a compute node and the appropriate IP group. Note that the compute nodes need to be unassigned. On the cloud groups page, select the first cloud group, **Cloud Group 1**. In the compute nodes section of the cloud group's properties, select the compute node to assign to this cloud group, **SN#23ZXL15**, as shown in Figure 8. In the IP groups section, select the IP group for the cloud group, **IP Group A**. Repeat these steps for the second cloud group, Cloud Group 2.
Environment profiles

You need to create two environment profiles, which you will use to deploy patterns to the cloud groups. Table 4 shows the properties for the two environment profiles. Note that each deploys to its own Cloud Group and IP Group.

Table 4. Environment profile properties

<table>
<thead>
<tr>
<th>Environment Profile Property</th>
<th>EnvProf-1</th>
<th>EnvProf-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cloud group</td>
<td>Cloud Group 1</td>
<td>Cloud Group 2</td>
</tr>
<tr>
<td>IP group</td>
<td>IP Group A</td>
<td>IP Group B</td>
</tr>
</tbody>
</table>

To create an environment profile, go to the environment profiles page (select Workload Console > Cloud > Environment Profiles) and press the green plus sign icon (+), which opens the "Create environment profile" dialog. Populate the dialog with the values for EnvProf-1 listed in Table 4, as shown in Figure 9, and press OK.
Figure 9. Create environment profile dialog

Describe the environment profile you want to create

- **Name:** EnvProf-1
- **Description:** Single cloud group, single compute node
- **Hypervisor type:** PureSystems_ESX
- **Environment:** All

Now add **Cloud Group 1** to the newly created environment profile **EnvProf-1**, as shown in Figure 10.

Figure 10. Add a cloud group to the environment profile

Once the cloud group has been added, you have to configure the environment profile to use the IP group associated with the cloud group. Tick the box as shown in Figure 11. Now the environment profile can be used for deployments. Repeat these steps for the other environment profile, **EnvProf-2**.
Deploy the demonstration workloads

The second step in demonstrating isolation between runtime environments is to deploy a workload to each of the runtime environments so that each environment can host its workload. The workloads do not need to be complete business applications, just simple applications sufficient to create load on the resources.

Create a virtual system pattern

First, you need a pattern that you can deploy as a workload. You will create a simple virtual system pattern with just a single part, the "IBM OS Image for Red Hat Linux Systems 2.0". Real workloads typically consist of more complex patterns that run in multiple virtual machines, however for the purpose of this demonstration, the base image is sufficient.

From the Workload Console, select Patterns > Virtual System Patterns to go to the virtual system patterns page and press the green plus sign icon (+) to create a new pattern. Provide a name, SingleCoreOS, and description in the dialog box and press OK, as shown in Figure 12.

Figure 12. Create virtual system pattern dialog

```
Describe the pattern you want to add.

* Name: SingleCoreOS

Description: Pattern with a single CoreOS VM

OK Cancel
```

Back in the virtual system patterns page, make sure your new pattern is selected and press the "edit" icon in the top right-hand corner to open the pattern editor. In the "Parts" list on the left side, look for Core OS, as shown in Figure 13, and drag it onto the canvas on the right.

Figure 13. Core OS pattern part

In the pattern editing canvas, select the Core OS part and press its "Properties" icon, as shown in Figure 14, to open the part's Properties dialog.
Fill in the dialog with the values in Table 5, as shown in Figure 15. Note that you are sizing the virtual machine to use just a single virtual CPU and 2048 MB of RAM. Set the passwords for root and virtuser here, so that you do not have to enter them every time you deploy the pattern.

Table 5. Core OS properties values

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual CPUs</td>
<td>16</td>
</tr>
<tr>
<td>Memory size (MB)</td>
<td>2048</td>
</tr>
<tr>
<td>Password (root)</td>
<td>password</td>
</tr>
<tr>
<td>Password (virtuser)</td>
<td>password</td>
</tr>
</tbody>
</table>

Once you have set the property values, press OK to close the properties dialog. In the pattern editing canvas, press Done Editing in the top right corner. This saves the changes you have made to the pattern and close the editor.

Deploy the virtual system instances

Second, you need to deploy the pattern to each of the cloud groups. You will deploy the pattern twice, once using each of the environment profiles. This deploys each pattern to its respective
cloud group, retrieving an IP address from that cloud group's IP group. As a reminder, the relationships between the deployment components are shown in Table 6.

Table 6. Deployment components for the virtual system pattern

<table>
<thead>
<tr>
<th>Virtual system instance</th>
<th>Environment profile</th>
<th>Cloud group</th>
<th>IP group</th>
</tr>
</thead>
<tbody>
<tr>
<td>SingleCoreOS-1</td>
<td>Env-Prof-1</td>
<td>Cloud Group 1</td>
<td>IP Group A</td>
</tr>
<tr>
<td>SingleCoreOS-2</td>
<td>Env-Prof-2</td>
<td>Cloud Group 2</td>
<td>IP Group B</td>
</tr>
</tbody>
</table>

On the virtual system patterns page, select your SingleCoreOS pattern and press Deploy to open the deployment dialog. In the section labeled Virtual system name, enter SingleCoreOS-1 for the name. Expand the section labeled Choose environment and select the Env-Prof-1 profile. The dialog looks like Figure 16. Then press OK to deploy the pattern. Once the pattern is deployed, repeat these steps to use Env-Prof-2 to deploy SingleCoreOS-2 to Cloud Group 2.

Figure 16. Virtual system pattern deployment dialog

Now the pattern instance SingleCoreOS-1 is running as a workload in cloud group Cloud Group 1 using an IP address from IP Group A, and SingleCoreOS-2 is running in Cloud Group 2 with an IP Group B address. You can verify this by going to the virtual system instances page (from the Workload Console select Instances > Virtual Systems) and selecting your pattern instance. The properties, as shown in Figure 17, show details, such as what pattern the instance was created from, its current status, and its virtual machines.
Verify that the status for both of your virtual system instances says, "The virtual system pattern has been deployed." You can also expand the virtual machines list to see how much CPU and memory each one is using.

**Demonstrate the workload isolation**

The third step in demonstrating isolation between runtime environments is to perform two tests:

- **Network isolation**: We will show the network route between the two workloads, showing that they are isolated from each other on the system and only connected to each other by the networks external to the system.
- **Computational isolation**: We will cause the workload in one of the runtime environments to consume a great deal of the environment's resources and show that the other environment and its workload are unaffected.

To start, you need to determine what hardware the workloads are running on so that you can perform the tests using that hardware.

**Hardware resources**

To demonstrate resource isolation, you first need to know specifically what hardware your pattern instances are running on. Typically, you do not know or care what hardware is being used by a particular workload. Part of the magic of cloud computing is that you know the workload is running somewhere in the cloud, but you do not need to know where. However, to demonstrate
resource isolation, you need to drill down into PureApplication System to see where it is running our demonstration workloads.

A workload is a running pattern instance that is composed of virtual machines, so you need to know what hardware resources those VMs are using. In this example, the settings are those shown in Table 7. The settings on your PureApplication System and your network will be different, but analogous.

### Table 7. Pattern instances' hardware resources

<table>
<thead>
<tr>
<th>Virtual system instance</th>
<th>VM name</th>
<th>Cloud group</th>
<th>Compute node</th>
<th>IP address (IPv4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SingleCoreOS-1</td>
<td>ipas-a-2-OS Node-SingleCoreOS-1-2405</td>
<td>Cloud Group 1</td>
<td>SN#23ZXL15</td>
<td>172.19.75.2</td>
</tr>
<tr>
<td>SingleCoreOS-2</td>
<td>ipas-b-2-OS Node-SingleCoreOS-2-2406</td>
<td>Cloud Group 2</td>
<td>SN#23ZXL05</td>
<td>172.19.76.2</td>
</tr>
</tbody>
</table>

How did we figure out the network settings shown in Table 7? Follow these steps:

1. Go to the virtual system instances page (select **Workload Console** > **Instances** > **Virtual Systems**) and select **SingleCoreOS-1**, as shown in Figure 17.
2. In the instance's properties list, expand its list of virtual machines. Add the virtual machine's name to the table.
3. Repeat these steps for SingleCoreOS-2.
4. Go to the virtual machines page (select **System Console** > **Cloud** > **Virtual Machines**) and select the SingleCoreOS-1’s VM, as shown in Figure 18.
1. In the virtual machine's properties list, find the cloud group, compute node, and IPv4 address, and add those to the table.
2. Repeat these steps for SingleCoreOS-2's VM.

This technique is also useful for diagnosing problems with how your organization's workloads are running on PureApplication System. When workloads do not seem to be running correctly, often the first step is to verify that they are running at all, and what hardware they are running on.

Network isolation

To demonstrate that the workloads are isolated from each other on the network, you will use ping and traceroute to show how they are connected and that this connection constitutes isolation. To show that the workloads are on the same network and therefore connected, you will log into the virtual machine of one of the pattern instances and ping the VM in the other pattern instance. This will work because both networks can reach each other through their respective gateways. However, there is no direct connection between the VMs.

To show that the pattern instances are indeed connected to the network via different networks, you will use traceroute to show the pathway of the connection between the separate pattern instances' VMs. Because the two pattern instances are connected to the network via different VLANs, traceroute will show that the connection goes through the network's default gateway.
(which is external from PureApplication System). To completely isolate the pattern instances, the network administrator can set up the gateway as a firewall that blocks any connection between the two networks.

To use ping and traceroute, you need to know the network settings of the virtual machines in our pattern instances. This was explained earlier in the Hardware resources section. Follow that procedure to complete a table for your system like that shown in Table 7.

The values shown in Table 7 are correct for the example on our system. Notice that the IP address for SingleCoreOS-1's VM is 172.19.75.2, which is clearly within the range of IP Group A, 172.19.75.2-254. Likewise, 172.17.76.2 for SingleCoreOS-2's VM is within the range of IP Group B. We know that within PureApplication System, these VMs are on different application VLANs. In turn, those VLANs are connected to different networks outside the system.

Now that you have the IP addresses for the virtual machines, you can test their network isolation:

1. Launch an SSH client, such as "Putty".
2. Provide the IP address for SingleCoreOS-1's VM, as shown in Figure 19. In our example, that VM can be reached using IP address 172.19.75.2.

Figure 19. Putty login

3. In the Putty session, use the command line to logon as root and provide the password you specified earlier (for example, "password").
   
   ```
   login as: root
   root@172.19.75.2's password:
   -bash-4.1#
   ```

4. Disable the VM's firewall by running `service iptables stop`. 
5. Use another Putty session to log on to SingleCoreOS-2’s VM (via 172.19.76.2) and disable its firewall as well.

6. With both firewalls disabled, back in the Putty session for SingleCoreOS-1’s VM, try to ping SingleCoreOS-2’s VM.

   -bash-4.1# ping 172.19.75.2
   PING 172.19.75.2 (172.19.75.2) 56(84) bytes of data.
   64 bytes from 172.19.75.2: icmp_seq=1 ttl=63 time=1.63 ms
   64 bytes from 172.19.75.2: icmp_seq=2 ttl=63 time=0.396 ms

7. The ping test succeeds, which proves that the VMs are able to connect to each other over the network. However, it is important to realize that they are able to connect only because the gateways have been configured to allow them to. To prove this, run the traceroute command from SingleCoreOS-1’s VM.

   -bash-4.1# traceroute 172.19.76.2
   traceroute to ipas-b-2.iic.hur.cdn (172.19.76.2), 30 hops max, 60 byte packets
   1  172.19.75.1 (172.19.75.1)  0.274 ms  0.265 ms  0.290 ms
   2  ipas-b-2.iic.hur.cdn (172.19.76.2)  0.692 ms  0.723 ms  0.694 ms

As you can see, the traceroute does not show just one hop to the target IP 172.19.76.2. Rather, it shows two hops, the first one is 172.19.75.1. This is the network’s default gateway, one of the settings you configured for IP Group A (which was used at deployment time to configure the network of the VM of SingleCoreOS-1). The default gateway is hosted outside PureApplication System, so that is where you can control the network isolation.

**Computational isolation**

To demonstrate that the workloads' computational resources (that is, CPU and memory) are isolated from each other, we will generate CPU load in one of our pattern instances and show that the load only uses resources in that workload’s cloud group, that the other cloud group remains unaffected. As you did earlier for testing network isolation, you need to create a table that describes the hardware resources your pattern instances are using, one like Table 7.

You will start by generating a system load on the SingleCoreOS-1. This translates into an actual system load on the Compute Node SN#23ZXL15 only.

1. Launch an SSH client such as Putty.
2. Provide logon details for SingleCoreOS-1’s VM and login as root. In our example, that VM can be reached using IP address 172.19.75.2.
3. Issue the following command. This launches a single-threaded process in the background to drive a single (virtual) CPU to 100% utilization.
   -bash-4.1# while true; do true; done &
4. Our Virtual Machine should have a total of 16 (virtual) CPUs. To confirm this, run the following command:
5. In order to drive all 16 CPUs to 100%, run the following command another 15 times:

```
-bash-4.1# while true; do true; done &
```

6. You should have a total of 16 jobs running in the background, as confirmed by the following command:

```
-bash-4.1# jobs
[1]    Running     while true; do True; Done &
... 
[16]+  Running     while true; do True; Done &
```

7. Now confirm that the CPU utilization reported by the OS on the VM is now at or close to 100%:

```
-bash-4.1# vmstat 5 5
procs ---------memory---------- ---swap-- -----io---- --system-- -----cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
1  0      0 1879008  14300  69764    0    0    26     4  564  298 54  0 46  0  0
1  0      0 1878992  14300  69764    0    0     0     0  916  31 100  0  0  0  0
1  0      0 1878992  14300  69764    0    0     0     0  917  34 100  0  0  0  0
1  0      0 1878992  14300  69764    0    0     0     0  917  31 100  0  0  0  0
```

The PureApplication System console shows the same virtual CPU utilization for that virtual machine.

8. In the management console, go to the virtual machines page (select **System Console > Cloud > Virtual Machines**).

9. Select the SingleCoreOS-1's VM: **ipas-d-221-OS Node-SingleCoreOS-1-2405**. To help find that VM in what can be a long list of VMs, filter the list. For example by entering **Single** as shown in Figure 20 (the pattern name is part of the VM name).
10. With **ipas-d-221-OS Node-SingleCoreOS-1-2405** selected, examine the properties panel on the right. The virtual CPU property shows that the utilization is at or near 100%, as shown in Figure 21. The actual value should match what was reported by the operating system tools earlier (vmstat).

**Note:** Sometimes the data presented in the console is not up-to-date. We found that it is typically refreshed once every 4 to 5 minutes.

**Figure 21. Properties for a virtual machine**

To demonstrate computational isolation between the cloud groups, we will now compare the CPU load in each cloud group’s compute node. This showed that the utilization is high in the compute node for SingleCoreOS-1 while remaining low in the compute node for SingleCoreOS-2.
With a high (virtual) CPU utilization on the SingleCoreOS-1 Virtual Machine, we expect to see corresponding (physical) CPU utilization on Compute Node SN#23ZXL15.

11. In the management console, go to the compute nodes page (select System Console > Hardware > Compute Nodes).

12. Select the SingleCoreOS-1's compute node, **SN#23ZXL15**.

13. With **SN#23ZXL15** selected, examine the properties panel on the right. You see that the CPU utilization of the physical cores is approximately 100%, as shown in Figure 22.

**Figure 22. A compute node with high CPU utilization**

![Compute Node with High CPU Utilization](image)

As we are driving the virtual machine with 16 virtual cores each driven to 100% utilization, the Compute Node (the underlying physical hardware) is driven to 100% of its 16 physical cores.

**Note:** Sometimes the data presented in the console is not up-to-date. In that case, just wait a few minutes until the console refreshes the data.

14. Now select the SingleCoreOS-2's compute node, **SN#23ZXL05**. You see that the CPU utilization of the physical cores is low, such as 1 to 2%, as shown in Figure 23.
We have demonstrated that a VM using all of its virtual cores drives the CPU utilization of the compute node where it is running. However, other compute nodes in the same PureApplication System remain unaffected. By design, a single compute node can be assigned to at most one cloud group. By deploying different workloads on different cloud groups, you can ensure that those workloads are never co-located on the same compute node. In other words, this is how you can achieve resource isolation.

Conclusion

This article demonstrated how workloads running in separate cloud groups are isolated from each other, both in terms of computational resources and networking. It discussed what it means for workloads to be isolated and explained the features in PureApplication System for creating isolated runtime environments. It also showed how to create the demonstration environments, deploy the test workloads, perform the tests, and discussed the test results. With this information, you have now seen the workload isolation provided by runtime environments in PureApplication System.

Acknowledgements

The authors would like to thank the following IBMers for their help with this article: Andy Bravery, Sara Mitchell, Kyle Brown, and Rohith Ashok.
Resources

- IBM PureApplication System
- IBM PureApplication System Version 1.0 Information Center
- Managing application runtime environments in IBM PureApplication System
- Navigating the IBM cloud, Part 2: Understanding virtual system patterns in IBM Workload Deployer and IBM PureApplication Systems
- IBM PureSystems Centre
- IBM PureSystems resources on developerWorks
About the authors

Hendrik van Run

Hendrik van Run is a Certified Consulting IT Specialist within the IBM Software Services for WebSphere (ISSW) consulting practice in Hursley, United Kingdom. He helps clients with the design, implementation, and analysis of their WebSphere Application Server infrastructure and specializes in IBM PureApplication System. He is co-author of a number of IBM Redbooks and a frequent conference speaker.

Venkata (Vishy) Gadepalli

Venkata Gadepalli (Vishy) is a member of the IBM for Software Services for WebSphere (ISSW) Pre-Sales / PoC team, which works closely with technical sales. He has more than 14 years of experience in the IT field and has been involved in customer engagements involving the WebSphere family of products. His main area of interest is working with first-of-a-kind engagements involving IBM PureApplication System. Vishy has authored numerous papers, which have been published both internally within IBM and outside of IBM. Vishy also co-authored the first and second editions of the WebSphere Portal Primer.

Bobby Woolf

Bobby Woolf is a Consultant for IBM Software Services for WebSphere (ISSW), focusing on IBM PureApplication System, service-oriented architecture, event-driven architecture, and application integration. He is the author of Exploring IBM SOA Technology and Practice, and a co-author of Enterprise Integration Patterns and The Design Patterns Smalltalk Companion.

© Copyright IBM Corporation 2013
Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)