iControl 101 - #23 - Module Resource Provisioning

Joe Pruitt, 2010-24-11

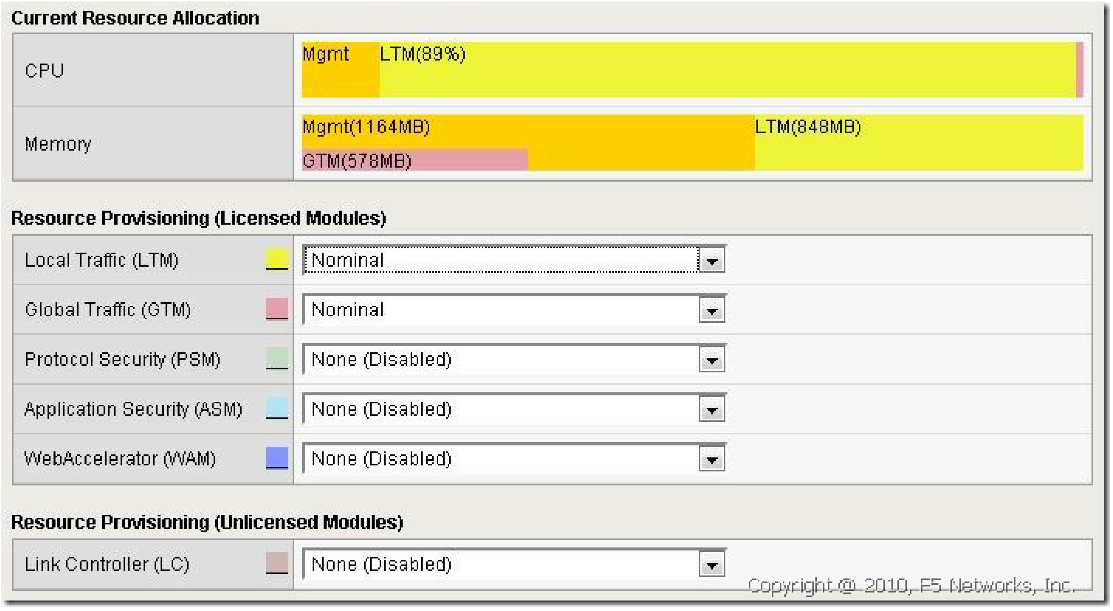
A somewhat unknown feature on the BIG-IP is the ability to override the default resource allocations for core modules on the BIG-IP. This is available in the product GUI under the System's "Resource Provisioning menu". In this article, I'm going to walk you through the iControl Management.Provision interface and show you how to automate the configuration of the resources.

The Provision interface manages the partitioning of system memory, disk space, and CPU usage among the modules (e.g., LTM, GTM, WOM) licensed to run on the system. The modules and the "host" system (everything not dedicated to network traffic processing) are first assigned a minimal set of resources, based on pre-determined values and the available system memory. The remaining resources are distributed among the modules, according to pre-determined rules, which can be modified by values which can be set via this iControl interface. The pre-defined module resource allocation handles a wide variety of circumstances, so this allocation should only need to be changed for unusual circumstances, such as turning modules on or off and large network traffic or configurations.

Read that last sentence: "**allocation should only need to be changed for unusual circumstances**". So, unless you have a need for doing so, I'd recommend keeping the default configuration. With that said, on with the article.

Querying the Provisioning settings

The default settings in the GUI can be found by looking on the System "Resource Provisioning" menu. A form similar to the following will be presented. You'll notice that I have 6 modules on my system: LTM, GTM, PSM, ASM, WAM, and LC.   Your configuration may be different depending on the version of your BIG-IP.



Current Resource Allocation

CPU	Mgmt LTM(89%)

Memory	Mgmt(1164MB) LTM(848MB)
	GTM(578MB)

Resource Provisioning (Licensed Modules)

Local Traffic (LTM)	🟨	Nominal
Global Traffic (GTM)	🟥	Nominal
Protocol Security (PSM)	🟩	None (Disabled)
Application Security (ASM)	🟦	None (Disabled)
WebAccelerator (WAM)	🟪	None (Disabled)

Resource Provisioning (Unlicensed Modules)

Link Controller (LC)	🟫	None (Disabled)

Copyright @ 2010, F5 Networks, Inc.

In my configuration, only LTM and GTM are enabled and they have the default allocation of "Nominal". The iControl Management.Provision interface contains the following methods to query the data from this form:

- get_list() - Query the list of LTM Modules
- get_provisioned_list() - Query the list of modules that have resources to run on the system (ie. not set to None)
- get_level(), set_level() - get or set the allocation level (none, minimal, nominal, dedicated, custom)
- get_custom_cpu_ratio(), set_custom_cpu_ratio() - get or set the custom CPU ratio (only for custom level)
- get_custom_disk_ratio(), set_custom_disk_ratio() - get or set the custom disk ratio (only for custom level)
- get_custom_memory_ratio(), set_custom_memory_ratio() - get or set the custom memory level (only for custom level)

Here's a PowerShell function that will use the iControl PowerShell Cmdlets to pull out the list of modules and then query the custom ratios for them.

```
function Get-Provisioning()
{
  Write-Host "Retrieving the Provisioning List...";

  $module_list = (Get-F5.iControl).ManagementProvision.get_list();
  $levels = (Get-F5.iControl).ManagementProvision.get_level($module_list);
  $cpu_ratios = (Get-F5.iControl).ManagementProvision.get_custom_cpu_ratio($module_list);
  $disk_ratios = (Get-F5.iControl).ManagementProvision.get_custom_disk_ratio($module_list);
  $memory_ratios = (Get-F5.iControl).ManagementProvision.get_custom_memory_ratio($module_list);

  Write-Entry "Module Name" "Level" "CPU Ratio" "Disk Ratio" "Memory Ratio"
  Write-Entry
  for($i=0; $i -lt $module_list.Length; $i++)
  {
    $level = $levels[$i].ToString().Replace("PROVISION_LEVEL_", "");
    Write-Entry $module_list[$i] $level $cpu_ratios[$i] $disk_ratios[$i] $memory_ratios[$i]
  }
}
```

Running this function on my system results in this output showing the 6 modules and that only LTM and GTM are provisioned.

```
Retrieving the Provisioning List...
     Module Name      Level     CPU Ratio    Disk Ratio Memory Ratio
---------------- --------- ------------ ------------ ------------
 TMOS_MODULE_LTM   NOMINAL            0            0            0
 TMOS_MODULE_GTM   NOMINAL            0            0            0
  TMOS_MODULE_LC      NONE            0            0            0
 TMOS_MODULE_PSM      NONE            0            0            0
 TMOS_MODULE_ASM      NONE            0            0            0
 TMOS_MODULE_WAM      NONE            0            0            0
```

Changing The Resource Allocation Setting

Now, let’s assume we have a need to tweak the configuration for some reason. In my example I’m going to illustrate how to set the allocation setting to Custom and tweak the various ratio settings for cpu, disk, and memory.

```
function Set-Custom()
{
  Write-Host "Setting Custom Provisioning levels";

  $modules = ("TMOS_MODULE_LTM", "TMOS_MODULE_GTM");
  $levels = ("PROVISION_LEVEL_CUSTOM", "PROVISION_LEVEL_CUSTOM");
  (Get-F5.iControl).ManagementProvision.set_level($modules, $levels);
  (Get-F5.iControl).ManagementProvision.set_custom_cpu_ratio($modules, (60, 30));
  (Get-F5.iControl).ManagementProvision.set_custom_disk_ratio($modules, (100, 100));
  (Get-F5.iControl).ManagementProvision.set_custom_memory_ratio($modules, (75, 50));
}
```

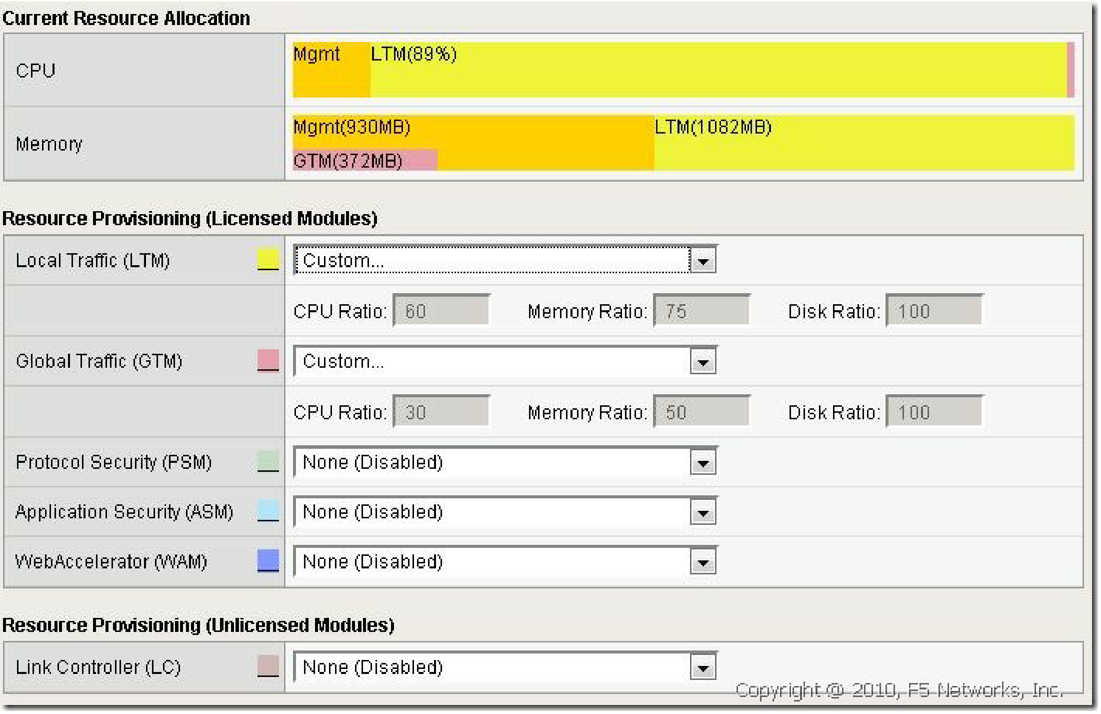
One thing that you’ll need to know for setting custom levels is that it’s an all or nothing thing. To set one module to a custom setting, you have to set ALL other modules that have allocated resources to custom as well.   You’ll notice in my script above that I set both LTM and GTM to the Custom level in the same method call. I then set the CPU ratios to 60 and 30, the Disk ratios to 100 and 100, and the Memory ratios to 75 and 50 respectively.

There is no rhyme nor reason to these numbers, I just picked some different values. The Ratio itself is a number from 0 to 255 with 0 being equivalent to the “minimal” allocation level.

After running this script the results can be seen by calling the above Get-Provisioning function in Powershell.

```
Retrieving the Provisioning List...
     Module Name      Level     CPU Ratio    Disk Ratio Memory Ratio
---------------- --------- ------------ ------------ ------------
 TMOS_MODULE_LTM    CUSTOM           60          100           75
 TMOS_MODULE_GTM    CUSTOM           30          100           50
  TMOS_MODULE_LC      NONE            0            0            0
 TMOS_MODULE_PSM      NONE            0            0            0
 TMOS_MODULE_ASM      NONE            0            0            0
 TMOS_MODULE_WAM      NONE            0            0            0
```

And the management GUI will look like this.



Current Resource Allocation

CPU	Mgmt LTM(89%)

Memory	Mgmt(930MB) LTM(1082MB)
	GTM(372MB)

Resource Provisioning (Licensed Modules)

Local Traffic (LTM) 🟨	Custom...			
		CPU Ratio: 60	Memory Ratio: 75	Disk Ratio: 100
Global Traffic (GTM) 🟥	Custom...			
		CPU Ratio: 30	Memory Ratio: 50	Disk Ratio: 100
Protocol Security (PSM) 🟩	None (Disabled)			
Application Security (ASM) 🟦	None (Disabled)			
WebAccelerator (WAM) 🟦	None (Disabled)			

Resource Provisioning (Unlicensed Modules)

Link Controller (LC) 🟫	None (Disabled)

Copyright @ 2010, F5 Networks, Inc.

Resetting The Resource Allocations To The Defaults

The default values for the allocation levels are the “Nominal” settings. To reset the values, you will have to switch all modules away from the “custom” settings at one shot (remember, it’s an all-or-nothing thing).

```
function Set-Nominal()
{
  Write-Host "Resetting Provisioning Levels to NOMINAL...";

  $modules = ("TMOS_MODULE_LTM", "TMOS_MODULE_GTM");
  $levels = ("PROVISION_LEVEL_NOMINAL", "PROVISION_LEVEL_NOMINAL");
  $ratios = (0, 0);
  (Get-F5.iControl).ManagementProvision.set_level($modules, $levels);
  (Get-F5.iControl).ManagementProvision.set_custom_cpu_ratio($modules, $ratios);
  (Get-F5.iControl).ManagementProvision.set_custom_disk_ratio($modules, $ratios);
  (Get-F5.iControl).ManagementProvision.set_custom_memory_ratio($modules, $ratios);
}
```

I went ahead and set all the ratios to zero but those values are ignored for any allocation levels other than “custom”.

Download The Source

The source for this article can be found in the iControl CodeShare under PowerShellManagementProvision.

F5 Networks, Inc. | 401 Elliot Avenue West, Seattle, WA 98119 | 888-882-4447 | f5.com

F5 Networks, Inc.
Corporate Headquarters
info@f5.com

F5 Networks
Asia-Pacific
apacinfo@f5.com

F5 Networks Ltd.
Europe/Middle-East/Africa
emeainfo@f5.com

F5 Networks
Japan K.K.
f5j-info@f5.com