Interface of Resource Allocator in V-NetLab

Mingwei Zhang
11/20/2010

This document contains the instructions of how to compile resource allocator daemon and protocol format of the daemon with group-manager. Specifically, it contains 3 parts:

1) How to use resource allocator daemon;
2) Protocol from group-manager to resource allocator daemon
   (Hereinafter referred as daemon).
3) Protocol on the other direction.

1 How to Use/Test Daemon

1.1 How to Compile Daemon

Change directory to allocator and use commands:

make OR
make clean && make OR
make daemon /*This will compile daemon separately*/

1.2 How to Start Daemon

./daemon /*run daemon at frontend*/

1.3 How to Test Daemon

There are two methods to do the testing. One is to use the client (another executable file at allocator directory); another one is to use the script "testDaemon.sh" which invokes client continuously with different parameters.

The following is the command line format of client:

client servAddr <networkfile> <hostfile> <targetLoad> <algorithm> [Additional Info]

For the script:

./testDaemon.sh

2 Protocol From Group-Manager To Daemon

2.1 Description

group-manager provides the following 5 pieces of info to the allocator daemon:

1 network file path (string)
2 host information path (string)
3 current target load (float)
4 algorithm chosen (int)
5 additional info (string)

Notice that the 1st and 2nd parameters are the absolute paths rather than the content of the files. The above info will be separated by 0x00 as a delimiter.

2.1.1 Network File Format

network file is file name of a network topology in chaco format. if using simulator these files could be found at /tmp. The following is a sample of a network file:

```
#- 1 4 49 432 48 /*Format: nodeId nodeName memory TotalCPU CPUperSecond */
#- 2 5 52 350 50
#- 3 6 53 689 53
#- 4 8 56 275 55
#- 5 9 56 280 56
#- 6 10 58 61 61
#- 7 12 62 310 62
7 6 111          /*number_of_nodes number_of_edges flags*/
1 48 3 102       /*nodeId nodeWeight Connected_nodeId edgeWeight*/
2 50 6 37 7 12
3 53 1 102
4 55
5 56
6 61 2 37
7 62 2 12
```

Notice that the above is consist of three parts including:

1) Lines starting with "#- ". It represents the node information
2) A line with exactly 3 numbers represents: number of nodes, number of edges and a flag
3) All the lines behind 2) to represents the edge/graph information

2.1.2 Host File Format

Host file information has EXACTLY the same format as the host info for group-manager. Please check the interface file of group-manager for more details

2.1.3 Other Parameters

target load is simply a float that is maintained by group-manager. It represents
target load after the most recent allocation

algorithm is an integer representing the algorithms that you may choose to do the allcation. The available algorithms are:

1)First Fit

2)Last Fit

3)Round Next Fit

4)Random Choose

Additional Info:

Additional Info is optional. For the round Next fit algorithm, additional information is need to tell daemon that where the "current" position is since last allocation.

2.2 Protocol Format

FORMAT: networkFile 0x00 hostFile 0x00 targetLoad 0x00 Algorithm 0x00 Additional_Info 0x00

Notice: 0x00 is the delimiter

Protocol:

networkFilePath	hostFilePath	targetLoad	Algorithm	Additional_Info

3 From Daemon to Group-manager

3.1 Description

After receiving the request from group-manager, the daemon will read both two files and give out the allocation result as well as teh updated target load.

In general, daemon's feedback contains the following

1 header information

2 a set of vm allocation results

2.1 vm name (string)

2.2 vm cpu allocation (int)

2.3 vm memory allocation (int)

2.4 vm network allocation (int)

2.5 vm completion rate (float)

2.6 vm hostName (string)

3.2 Protocol Format

FORMAT: header_info vm_allocation_formats

header_info:

result (int) /*1 success; 0 failure*/
targetLoad (float) /*updated targetLoad*/
vm_number (int) /*number of vm_allocation_format*/
additional_info (string) /*for some allocation algorithm only*/

vm_allocation_formats: vm_allocation_format vm_allocation_formats

vm_allocation_format:

name (string)
cpu (float)
memory (int)
nw (float)
completRate (float)
hostname (string)

Notice: there is no delimiter

* When allocation fails, there is no need to parse info behind.

Protocol:

Header Information	vm_allocation_formats

Header Information:

result	targetLoad	vm_number	additional_info

vm_allocation_formats:

vm_allocation_format	vm_allocation_formats

vm_allocation_format:

name	cpu	memory	nw	completRate	hostname