QuickStart: rsfcdb

Objective

This guide provides an overview of how to create, update and distribute an RSF-1 cluster configuration using the rsfcdb command line utility [1]. The rsfcdb utility is located at /opt/HAC/RSF-1/bin/rsfcdb.

The rsfcdb utility manipulates the RSF-1 configuration database, which in itself is initialized when RSF-1 is first installed via a post installation script. Note that subsequent updates of RSF-1 (i.e. installation of a newer RSF-1 package) does not overwrite an existing database and its properties (i.e. once set they are persistent unless explicitly changed).

Getting started

When RSF-1 is first installed the configuration database will contain no configuration data; this can be checked using the **config_preview** sub-command thus:

```
root@rsf1# rsfcdb config_preview
#
# Created Thu 2014-02-06 23:04:59 GMT by RSF-1 config_db
# This file is automatically generated by rsfcdb - DO NOT EDIT BY HAND

# Global Section

# Machine Section

# Services Section
```

Lines beginning with a # are comments. In the above preview each section of the configuration file is commented, but as yet contains no data. Lets start by naming the cluster:

```
# rsfcdb ga_name Quick_Start_Cluster
```

If we run the config_preview command again we now get:

```
root@demo1# rsfcdb config_preview
#
# Created Thu 2014-02-06 23:18:05 GMT by RSF-1 config_db
# This file is automatically generated by rsfcdb - DO NOT EDIT BY HAND

# Global Section
CLUSTER_NAME Quick_Start_Cluster

# Machine Section

# Services Section
```

[1] RSF-1 also provides a C++ API to allow programatic configuration file creation. It is located in `/opt/HAC/RSF-1/lib` as `librsfcdb.a`

Using rsfcdb we can also interrogate any single property:

```
root@demo1# rsfcdb gg_name
Quick_Start_Cluster
root@demo1#
```

All sub-commands follow a specific syntactical pattern. The first character refers to the section the sub-command applies to. There are three distinct sections in the configuration file, the global section where sub-commands are suffixed by a “g”; the heartbeats section where sub-commands are suffixed by an “h” and the service section where sub-commands are suffixed by an “s”.

The next character is either a “a”, a “r” or a “g”, where “a” is for add, “r” is for remove and “g” is for get. Next the underscore character “_” separates the section and request type from the actual property to be acted on. Some sub-commands have a third character, the letter “p” after the section/action pair which signifies that a pair of values are required for this sub-command (more on this later).

Breaking down the ga_name sub-command we can read this as Global Add Name, and similarly gg_name can be broken down as Global Get Name.

Finally lets remove the cluster name then re-add it:

```
root@demo1# rsfcdb gg_name
Quick_Start_Cluster
root@demo1# rsfcdb gr_name
root@demo1# rsfcdb gg_name
root@demo1# rsfcdb ga_name Quick_Start_Cluster
root@demo1# rsfcdb gg_name
Quick_Start_Cluster
root@demo1#
```

At any point entering rsfcdb on its own will output a help summary documenting all of the available sub-commands.

Creating a more complete globals section

Next, lets fill in the remainder of the globals section with some commonly used values:

```
root@demo1# rsfcdb ga_bo 20,3,3600
root@demo1# rsfcdb ga_pt 1
root@demo1# rsfcdb ga_rt 1
root@demo1# rsfcdb ga_en "/opt/HAC/RSF-1/bin/event_notifier"
root@demo1# rsfcdb ga_ipd 3,2
```

The above set of commands added, in order, are: the disk back off, the poll time value, the real time value, an event notifier and finally the ip device monitor enabler. All of these options are documented in the administrators guide available from the high-availability web site.

Previewing the configuration file now gives us:

```
root@demo1# rsfcdb config_preview
#
```

```
# Created Tue 2014-02-11 23:01:19 GMT by RSF-1 config_db
# This file is automatically generated by rsfcdb - DO NOT EDIT BY HAND

# Global Section
CLUSTER_NAME Quick_Start_Cluster
DISC_HB_BACKOFF 20,3,3600
POLL_TIME 1
REALTIME 1
EVENT_NOTIFY "/opt/HAC/RSF-1/bin/event_notifier"
IPDEVICE_MONITOR 3,2

# Machine Section

# Services Section
```

Next lets populate the machines section.

Creating the machine section

To declare machines in the machine section we simply add a set of heartbeats between machines in the cluster - if those machines do not exist in the configuration, rsfcdb simply adds them for you:

```
root@demo1# rsfcdb ha_net demo1#demo2 demo2#demo1
```

Breaking this command down we are declaring one network heartbeat from demo1 to demo2 and a second one from demo2 to demo1:

```
root@rsf1# rsfcdb config_preview
#
# Created Tue 2014-02-11 23:32:37 GMT by RSF-1 config_db
# This file is automatically generated by rsfcdb - DO NOT EDIT BY HAND

# Global Section
CLUSTER_NAME Quick_Start_Cluster
DISC_HB_BACKOFF 20,3,3600
POLL_TIME 1
REALTIME 1
EVENT_NOTIFY "/opt/HAC/RSF-1/bin/event_notifier"
IPDEVICE_MONITOR 3,2

# Machine Section
MACHINE demo1
 NET demo2
MACHINE demo2
 NET demo1
```

To add a further heartbeat over a private network we can run the command:

```
root@demo1# rsfcdb ha_net demo1#demo2-priv demo2#demo1-priv
root@demo1# rsfcdb config_preview
#
# Created Tue 2014-02-11 23:48:48 GMT by RSF-1 config_db
# This file is automatically generated by rsfcdb - DO NOT EDIT BY HAND
```

```
# Global Section
CLUSTER_NAME Quick_Start_Cluster
DISC_HB_BACKOFF 20,3,3600
POLL_TIME 1
REALTIME 1
EVENT_NOTIFY "/opt/HAC/RSF-1/bin/event_notifier"
IPDEVICE_MONITOR 3,2

# Machine Section
MACHINE demo2
 NET demo1
 NET demo1-priv
MACHINE demo1
 NET demo2
 NET demo2-priv

# Services Section
```

Finally, be aware that rsfcdb has some shortcut commands mentioned earlier, they have a third “p” character. As an example these two commands produce identical entries:

```
root@demo1# rsfcdb ha_net demo1#demo2 demo2#demo1
```

And

```
root@demo1# rsfcdb hap_net demo1#demo2
```

The “p” means mirrored pair, so hap_net creates heartbeats from demo1 to demo2 and then the mirror of these demo2 to demo1.

Next lets add some disk heartbeats (this is usually done at service creation time as disk heartbeats tend to be associated with services, however for the purpose of this tutorial we’ll just add some):

```
root@demo1# rsfcdb ha_disc\
  demo1#demo2#/dev/rdsk/c3t5000C5000EAE27B7d0s0:512:518#demopool1\
  demo2#demo1#/dev/rdsk/c3t5000C5000EAE27B7d0s0:518:512#demopool1

root@demo1# rsfcdb config_preview
#
# Created Thu 2014-02-13 00:06:15 GMT by RSF-1 config_db
# This file is automatically generated by rsfcdb - DO NOT EDIT BY HAND

# Global Section
CLUSTER_NAME Quick_Start_Cluster
DISC_HB_BACKOFF 20,3,3600
POLL_TIME 1
REALTIME 1
EVENT_NOTIFY "/opt/HAC/RSF-1/bin/event_notifier"
IPDEVICE_MONITOR 3,2

# Machine Section
MACHINE demo2
 NET demo1
 NET demo1-priv
 DISC demo1 /dev/rdsk/c3t5000C5000EAE27B7d0s0:518:512 TAG demopool1
MACHINE demo1
 NET demo2
```

```
 NET demo2-priv
 DISC demo2 /dev/rdsk/c3t5000C5000EAE27B7d0s0:512:518 TAG demopool1

# Services Section
```

This is probably the most complicated option we've used so far; but breaking it down we can view the heartbeat path as thus:

[<source>#<destination>#<path>#<tag>]

As in the previous heartbeat command, the source and destination are specified along with the path; here the path points to a raw disk device and also includes read and write offsets into the drive. Finally a tag is appended to link the heartbeat to a service, which we add in the following section.

Creating a service section and completing the configuration

To complete the configuration we'll now add a service to the cluster and preview it:

```
root@demo1# rsfcdb sa_desc demopool1#vip01 "Demopool1 service"
root@demo1# rsfcdb sa_opt  demopool1#vip01 sdir=appliance#ip_up_after=1
root@demo1# rsfcdb sa_it   demopool1#vip01 20
root@demo1# rsfcdb sa_rt   demopool1#vip01 8
root@demo1# rsfcdb sa_mp   demopool1#vip01 /demopool1
root@demo1# rsfcdb sa_ipd  demopool1#vip01 demo1#bge0
root@demo1# rsfcdb sa_ipd  demopool1#vip01 demo2#e1000g
root@demo1# rsfcdb config_preview
#
# Created Thu 2014-02-13 00:40:12 GMT by RSF-1 config_db
# This file is automatically generated by rsfcdb - DO NOT EDIT BY HAND

# Global Section
CLUSTER_NAME Quick_Start_Cluster
DISC_HB_BACKOFF 20,3,3600
POLL_TIME 1
REALTIME 1
EVENT_NOTIFY "/opt/HAC/RSF-1/bin/event_notifier"
IPDEVICE_MONITOR 3,2

# Machine Section
MACHINE demo2
 NET demo1
 NET demo1-priv
 DISC demo1 /dev/rdsk/c3t5000C5000EAE27B7d0s0:518:512 TAG demopool1
MACHINE demo1
 NET demo2
 NET demo2-priv
 DISC demo2 /dev/rdsk/c3t5000C5000EAE27B7d0s0:512:518 TAG demopool1

# Services Section
SERVICE demopool1 vip01 "Demopool1 service"
 OPTION "sdir=appliance ip_up_after=1"
 INITIMEOUT 20
 RUNTIMEOUT 8
 MOUNT_POINT "/demopool1"
 SERVER demo2
  IPDEVICE "e1000g"
 SERVER demo1
```

IPDEVICE "bge0"

If you are happy with the contents of the config_preview command, you can generate an RSF config file from the database contents by running:

root@demo1# rsfcdb config_make <path>

For example, running rsfcdb config_make myconfig will create an RSF config file called 'myconfig' in the current directory. If no path is given, the config file will be generated to a file called 'config' in the current directory.

Finally, you can use rsfcdb to distribute the new config file by running:

root@demo1# rsfcdb dist <config> <nodes...>

Here 'config' is the configuration file to be distributed and ‘nodes’ is a space-separated list of nodes that should be sent the configuration file.

The dist command runs the RSF config_dist command as a sub-command - it doesn't do anything that config_dist doesn't, it is simply a convenience function. This command can also be provided with a -h flag to enable a 'hot restart' of RSF as opposed to the default 'cold restart' method. The command format with the hot restart flag is as follows:

root@demo1# rsfcdb -h dist <config> <nodes...>

This completes the RSFCDB quick start guide - running rsfcdb on it’s own with no arguments will produce a complete list of all the configuration commands available and should be your next point of call for coverage of all the commands available.