iC2I: Dumping a Pool

Don MacVittie, 2008-15-05

The Pool class is the glue that binds Virtual Servers to Nodes. Central to what happens in LTM, the Pool holds information about what nodes it represents, how load balancing is handled between them, and what happens when not enough nodes are available.

Much like the dump routine we developed for VirtualServer, this article simply makes the API calls to get the information we wish to include in the dump and writes it to standard out. As with most i2CI articles, there is nothing special about what we're doing, but we'll point out the areas where things aren't necessarily straight-forward.

Once again, if you are using Eclipse with the iControl JAR file installed then simply posting the dump routine into the source file will generate the correct import statements for you. If you are not an Eclipse user, you will want to add the following import statements to the top of the BigIpPool class:

```
import iControl.LocalLBObjectStatus;
import iControl.LocalLBLBMethod;
import iControl.CommonEnabledState;
import iControl.CommonHAAction;
import iControl.LocalLBServiceDownAction;
import iControl.LocalLBPoolPoolStatistics;
import iControl.LocalLBPoolPoolStatisticEntry;
import iControl.CommonStatistic;
import iControl.CommonStatisticType;
```

WIth those imports in place, we can enter the pool dump routine. Since you can look through the source and observe the output we won't dissect it for you. We will call out one or two items for you after the source listing.

```
public void dump(String poolName) throws java.lang.Exception {
        // Make poolName feel like an array...
        String poolNames[] = {poolName};

        System.out.println("Dumping Pool: " + poolName);

        // Get the current status
        LocalLBObjectStatus status[] = stub.get_object_status(poolNames);
        System.out.println("Current Status: " + status[0].getStatus_description());

        // Get the Load Balancing Method
        LocalLBLBMethod lbMethod[] = stub.get_lb_method(poolNames);
        System.out.println("Load Balancing Method: " + lbMethod[0].toString());

        // Get the active member count
        long members[] = stub.get_active_member_count(poolNames);
        System.out.println("Total members active: " + members[0]);

        // Check if the minimum up member count is enabled
        CommonEnabledState upMemberTracking[] = stub.get_minimum_up_member_enabled_state(poolNames);
        if(upMemberTracking[0].equals(CommonEnabledState._STATE_DISABLED))
            System.out.println("There is no minimum number of active members enabled.");
        else {
            // Get the minimum overall member count
            members = stub.get_minimum_active_member(poolNames);
            System.out.println("Minimum active members: " + members[0]);

            // Get the minimum up member count
            members = stub.get_minimum_up_member(poolNames);
            System.out.println("Minimum members up: " + members[0]);

            // Get the action when less than minimum up member count is crossed
            CommonHAAction action[] = stub.get_minimum_up_member_action(poolNames);
            System.out.println("When not enough members up, pool will: " + action[0].toString());
        }

        // Get the list of nodes
        CommonIPPortDefinition nodeList[][] = stub.get_member(poolNames);
        CommonIPPortDefinition poolNodes[] = nodeList[0];
        if(poolNodes.length == 0)
            System.out.println("There are no Nodes assigned to this pool.");
        else {
            System.out.println("Nodes assigned to this pool:");
            for(int i = 0; i < poolNodes.length; i++)
                System.out.println("\t" + poolNodes[i].getAddress() + "\tOn port: " + poolNodes[i].ge
        }

        // Get the action on service down
        LocalLBServiceDownAction failAction[] = stub.get_action_on_service_down(poolNames);
        System.out.println("When the service goes down, it will: " + failAction[0].toString());

        // Get the timeout value
        long timeouts[] = stub.get_simple_timeout(poolNames);
        System.out.println("This pool's timeout value is: " + timeouts[0]);

        // Get the ramp up time value
        long rampupTimes[] = stub.get_slow_ramp_time(poolNames);
        System.out.println("Ramp up time for this pool: " + rampupTimes[0]);

        // Get Statistics
        LocalLBPoolPoolStatistics poolStats = stub.get_statistics(poolNames);
        System.out.print("Statistics for this pool as of ");
        System.out.print(poolStats.getTime_stamp().getHour() + ":" + poolStats.getTime_stamp().getMin
        System.out.println(" on " + poolStats.getTime_stamp().getMonth() + "/" + poolStats.getTime_st

        LocalLBPoolPoolStatisticEntry ourPoolStats[] = poolStats.getStatistics();

        // LocalLBPoolPoolStatisticEntry is one element per pool passed in. Since we only passed one
        CommonStatistic statsList[] = ourPoolStats[0].getStatistics();
        for(int j = 0; j < statsList.length; j++) {
            CommonStatisticType type = statsList[j].getType();
            UsefulU64 value = new UsefulU64(statsList[j].getValue());
            System.out.println("\t" + type.toString() + "\t" + value.toString());
        }
    }
```

While most of the data accessed in this routine is well documented on the Wiki, it is worth noting that poolStats.getTime_stamp().toString() returns a description of the class as if Class.toString() was called, not the timestamp you would expect.

The other interesting bit is that get_minimum_up_member_enabled_state() is used as a flag to tell the system whether the values in get_minimum_active_member(), get_minimum_up_member(), and get_minimum_up_member_action() are to be used or ignored.

Finally, statistics are much like Virtual Server statistics, with the array held in the LocalLBPoolPoolStatisticEntry object being one element per Pool passed in to the stub.getStatistics() method.

That's it for dumping a Pool. We are going to leave this topic of sending information about an object to output for a while, and present you with some cool little tools in the next few articles.

Get the Flash Player to see this player.