Managing Ramcache Entries with Pycontrol

Jason Rahm, 2012-10-01

A DevCentral user posted a question in the forums asking for verification of an attribute on the RamCacheKey structure. The maximum_responses attribute should be a long integer. With his C# code, the maximum_responses returned from his iControl call is always double the setting. I fired up pycontrol to see if this was a bug, and in my pycontrol code, I received the expected responses. While he's taking a look at his code and the .Net library he's using, I took an interest in the ramcache methods as I have not messed with the Ramcache module much. In this article, I'll build out a pycontrol script that will enable users to query/evict ramcache entries.

The Interface

The iControl interface for querying/evicting Ramcache entries is RAMCacheInformation. The methods I'll handle in the script are:

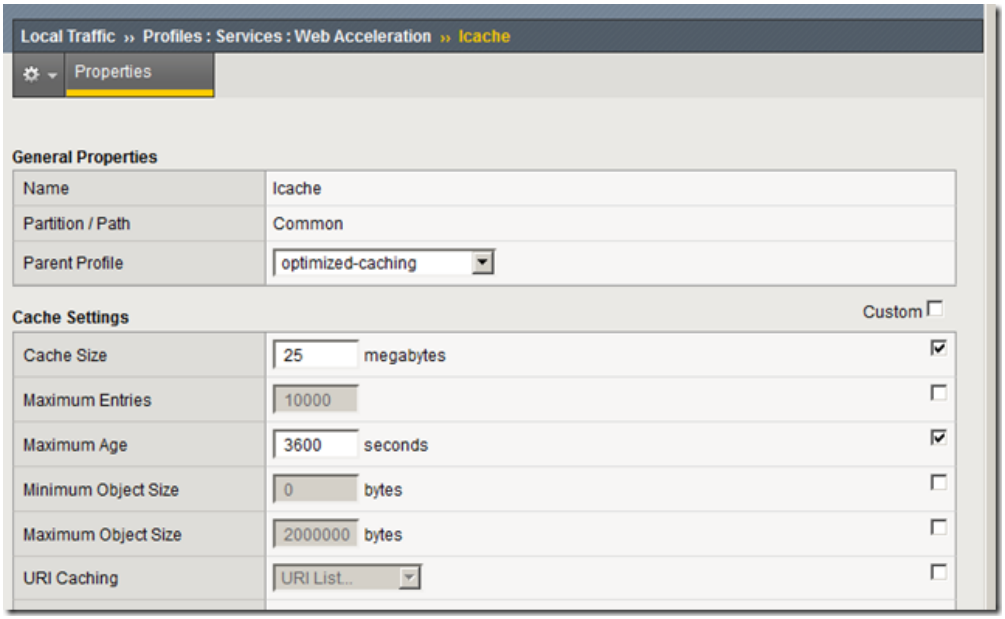
- get_ramcache_entry
- get_ramcache_entry_exact_match
- evict_ramcache_entry_v2
- evict_all_ramcache_entries

There are two more methods defined:

- get_version – All interfaces have this method, not important to Ramcache function
- get_ramcache_entry – Deprecated in favor of _v2 above.

Preparing the BIG-IP

Beginning in v11, the ramcache configuration was removed from the http profile and moved into its own under Local Traffic->Profiles->Services->Web Acceleration. I created a profile called lcache with a parent profile of optimized-caching, increasing the cache size to 25 MB and decreasing the max age to 3600 seconds for testing. I left all the other settings default.



Apply this profile to an http virtual and a pool member with some content and I'm ready to move to the iControl work.

Building the Script

Because there are multiple actions in this interface I want to address, I'll use command line options to control the script flow:

- Host – (-H) **Required**
- Username (-u) **Required**
- Get Entries (-g) **Profile Required**
- Get Exact Entry (-x) **Profile, Host and URI required** (as a white-space separated list in quotes. Ex. “lcache 10.10.20.60 /images/img1.png”
- Evict Entry (-E) **Profile, Host and partial URI required**
- Evict Exact Entry (-X) **Profile, Host and URI required**
- Evict All Entries (-A) **No arguments necessary, empties the ramcache**

With that plan in place, I can configure the parser options, shown in the following code snippet:

```
1:  from optparse import OptionParser
2:
3:  parser = OptionParser()
4:  parser.add_option("-H", "--host", action="store", type="string", dest="host")
5:  parser.add_option("-u", "--user", action="store", type="string", dest="uname")
6:  parser.add_option("-g", "--get", action="store", type="string", dest="get", help="Supply profile name")
7:  parser.add_option("-x", "--get_exact", action="store", type="string", dest="get_exact", help="Supply profile name, host, and
8:  parser.add_option("-E", "--evict", action="store", type="string", dest="evict_entry")
9:  parser.add_option("-X", "--evict_exact", action="store", type="string", dest="evict_exact")
10:  parser.add_option("-A", "--evict_all", action="store_false", default=False, dest="evict_all")
11:  (options, args) = parser.parse_args()
12:
13:  if (options.host is None) or (options.uname is None):
14:   print "\n\tHost (-H) and username (-u) must be supplied, exiting..."
15:   sys.exit()
```

Now that the options are in place, we need to act based on the options provided

```
if options.get is not None:
 rce = getRAMCacheEntries(rc, options.get)
 print rce

elif options.get_exact is not None:
 rce = getRAMCacheExactEntry(rc, options.get_exact)
 print rce

elif options.evict_entry is not None:
 rce = evictRAMCacheEntry(rc, options.evict_entry)
 print rce

elif options.evict_exact is not None:
 rce = evictRAMCacheExactEntry(rc, options.evict_exact)
 print rce

elif options.evict_all is not None:
 rce = evictRAMCacheAllEntries(rc)
 print rce

else:
```

```
22:   print "No options selected!"
```

```
23:   sys.exit()
```

In this code, I'm checking for the options and if none are set, I'm exiting. For each option, I'm calling a function to operate on that particular action. I'm passing the iControl object (see code below) and the argument list ("profile host uri") as necessary to each function.

```
1:  b = pc.BIGIP(
```

```
2:   hostname = options.host,
```

```
3:   username = options.uname,
```

```
4:   password = upass,
```

```
5:   fromurl = True,
```

```
6:   wsdls = ['LocalLB.RAMCacheInformation']
```

```
7:  )
```

```
8:
```

```
9:  rc = b.LocalLB.RAMCacheInformation
```

For all the functions except the one to handle evicting all ramcache entries, I need to create an object for the RamCacheKey structure. This is done in pycontrol with the typefactory. Once the structure is created, I just need to add the appropriate attributes as defined in the API: profile_name, host_name, uri, and maximum_responses. The first three are strings, the last is a long integer. So the code is very similar in each function:

```
1:  rc_key=obj.typefactory.create('LocalLB.RAMCacheInformation.RAMCacheKey')
```

```
2:
```

```
3:  key_attr = rc_info.split(' ')
```

```
4:  rc_key.profile_name = key_attr[0]
```

```
5:  rc_key.host_name = ''
```

```
6:  rc_key.uri = ''
```

```
7:  rc_key.maximum_responses = 100L
```

I split the arguments to get each attribute isolated, then assign as appropriate. For a general get, I only need the profile name so I set the host and uri to an empty string. I chose 100 as a max response just to pick something. Now that the structure is defined, I can make an iControl call, in this case, to get all ramcache entries, then I return those values from the function to the variable making the function call above.

```
1:  rc_entry = obj.get_ramcache_entry(keys=[rc_key])
```

```
2:  return rc_entry
```

So if I put all this together, I can issue this command at the command line, with the data below returned:

```
C:\Users\jrahm\PycharmProjects\pyControl_Scripts>python ramcache_manager.py -H 10.10.20.5 -u admin -g
Please enter the password for user admin
Password:
[[(LocalLB.RAMCacheInformation.RAMCacheEntry){
   profile_name = "lcache"
   host_name = "10.10.20.60"
   uri = "/images/gnome-64.png"
   vary_type = "RAM_CACHE_VARY_NONE"
   vary_count = 1
   hits = 0
   received = 1326237864
   last_sent = 1326237864
   expiration = 1326241060
   size = 4677
 }, (LocalLB.RAMCacheInformation.RAMCacheEntry){
   profile_name = "lcache"
   host_name = "10.10.20.60"
   uri = "/images/ad_remoteauth_ssh_1.png"
   vary_type = "RAM_CACHE_VARY_NONE"
   vary_count = 1
   hits = 0
   received = 1326237873
   last_sent = 1326237873
   expiration = 1326241069
   size = 16304
 }, (LocalLB.RAMCacheInformation.RAMCacheEntry){
   profile_name = "lcache"
   host_name = "10.10.20.60"
   uri = "/images/ad_descr.png"
   vary_type = "RAM_CACHE_VARY_NONE"
   vary_count = 1
   hits = 0
   received = 1326237818
   last_sent = 1326237818
   expiration = 1326241014
```

```
   size = 17942
 }]]
```

Now, if I want to evict one of those, I can change my options a little and try again, then do another query:

```
C:\Users\jrahm\PycharmProjects\pyControl_Scripts>python ramcache_manager.py -H 10.10.20.5 -u admin -E
Please enter the password for user admin
Password:
None

C:\Users\jrahm\PycharmProjects\pyControl_Scripts>python ramcache_manager.py -H 10.10.20.5 -u admin -g
Please enter the password for user admin
Password:
[[(LocalLB.RAMCacheInformation.RAMCacheEntry){
   profile_name = "lcache"
   host_name = "10.10.20.60"
   uri = "/images/gnome-64.png"
   vary_type = "RAM_CACHE_VARY_NONE"
   vary_count = 1
   hits = 0
   received = 1326237864
   last_sent = 1326237864
   expiration = 1326241060
   size = 4677
 }, (LocalLB.RAMCacheInformation.RAMCacheEntry){
   profile_name = "lcache"
   host_name = "10.10.20.60"
   uri = "/images/ad_remoteauth_ssh_1.png"
   vary_type = "RAM_CACHE_VARY_NONE"
   vary_count = 1
   hits = 0
   received = 1326237873
   last_sent = 1326237873
   expiration = 1326241069
   size = 16304
 }]]
```

You can see that the entry is now gone. And finally, If I evict all:

```
C:\Users\jrahm\PycharmProjects\pyControl_Scripts>python ramcache_manager.py -H 10.10.20.5 -u admin -A
Please enter the password for user admin
Password:
None

C:\Users\jrahm\PycharmProjects\pyControl_Scripts>python ramcache_manager.py -H 10.10.20.5 -u admin -g
Please enter the password for user admin
Password:
[[]]
```

Conclusion

There is access to great power via the iControl interface of your BIG-IP. With just a few lines of code, you can manage the ramcache from the CLI. For the full script, please check out the RAMCache Manager wiki page in the iControl codeshare. Happy Coding!

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