Sunday, March 5. 2017

Empty homes

Recently I had a performance tuning gig at a customer reporting that despite having the same number of vCPUs configured into the Logical Domain, the performance of both systems was different. A further observation was that a significant number of CPUs weren't used on that system.
The explanation for this is quite easy. Solaris is using a concept using locality groups. To vastly simplify it: Solaris knows about the topology of the system and thus tries to dispatch processes accordingly. For example to prevent to run a process on a core that is on the diametral side of the system than the memory the process is using. Locality group are an important part of the technology allowing Solaris to run loads on very large systems.

I have to explain that the system in question was virtually through dozens or hundreds of iterations in regard of the Logical Domain configuration. Memory added, CPU added, CPUs removed, CPU added, CPU removed, Memory added, half of that removed, another domain memory added, memory removed, CPUs added. Let's say it that way: The running configuration had a long and changeful life before it went into production without a further reboot. 300 days of uptime without a problem. Because of a performance problem the admins added some CPUs to the LDOM. Memory looked fine … no shortage. So no memory was added.

Okay, with this knowledge I started to check the configuration. On multiprocessor systems I always check the output of lgrpinfo when I see not evenly distributed load (when the uneven load wasn't enforced by processor sets).

The lgrpinfo command yielded the following output:

lgroup 7 (leaf):
Children: none, Parent: 6
CPUs: xxx-yyy zzz-aaa
Memory: installed 512M, allocated 512M, free 0K
Lgroup resources: 7 (CPU); 7 (memory)
Load: 0
Latency: 20
A lot of vCPU in the locality group. That what perfectly normal. But ... ouch ... just a soupçon of memory and nothing of it not in use.

After checking the LDOM configuration everything was obvious. The customer had actually configured a logical domain where it used a number of CPUs but the configuration didn't used any memory from the same CPUs.

How does this affect the system performance. Well, when a process is started a so called home locality group is assigned to it. All threads of the process inherits this home locality group later on. The dispatcher will try to run the the process as often as possible in this home locality group.

What has this to do with unevenly loaded cores: Well … Solaris ignores locality groups that have no memory left when choosing a home locality group. Makes a lot of sense. Processes would run there and have to go always to memory with higher latencies as the memory is for example on a different board behind a different CPU. And the locality group as totally out of memory as indicated by the lgrpinfo output as well as by the output of kstat -p:
kstat -p | grep "lgrp" | grep "pages free" | grep "\t0"
lgrp:7:lgrp7:pages free 0.
So the dispatcher almost never put a process in locality group as only a few unimportant processes had their home locality group set to lgroup 7. And for other processes the dispatcher had a preference to run them in different lgroups. And thus the locality group was never used a lot because the system simply ignored it right from the start. Perfectly correct and reasonable because of the situation.

You can check the home locality group of a process with prstat -H for example or with plgrp -h pid/lwp> to check it for a single process.

The conclusion out of this at least for me: Consider memory and CPU as a combination, not something that you can add one without thinking about the other. Oracle VM for SPARC knows this and tries to assign memory and CPUs in a optimal manner. But sometimes even this logic can't do wonders when all is fragmented. If time has messed up your

configuration of a system running virtually forever: Absolutely easiest way. halt all domains, unbind them, rebind them, let Oracle VM for SPARC take care of it. There are other ways, but those involve more typing. And think about the fact that memory and CPUs don't exist independently in modern systems when you are doing your configuration or fast changes on the fly.

Posted by Joerg Moellenkamp in English, Solaris, SPARC at 20:42

Great post! As always.

What about this situation: a leaf with only memory resource, without a CPU

lgroup 2 (leaf):
Children: none, Parent: 0
Memory: installed 5.5G, allocated 5.5G, free 0K
Lgroup resources: 2 (memory)

How should I understand this?
Anonymous on Mar 23 2017, 16:00