# Arvados - Bug #5425

## [Crunch] SLURM Node failure

03/09/2015 03:04 PM - Peter Amstutz

| | | | |
|---|---|---|---|
| **Status:** | Resolved | **Start date:** | 03/23/2015 |
| **Priority:** | Normal | **Due date:** | |
| **Assigned To:** | Peter Amstutz | **% Done:** | 100% |
| **Category:** | Crunch | **Estimated time:** | 0.00 hour |
| **Target version:** | 2015-04-01 sprint | | |

**Description**

I have seen repeated SLURM node failures when re-running Sally's killer job of doom. We should find out why.

https://workbench.qr1hi.arvadosapi.com/jobs/qr1hi-8i9sb-7jc0nde0tqv3u6y
https://workbench.qr1hi.arvadosapi.com/jobs/qr1hi-8i9sb-exy86qos9p66mbb
https://workbench.qr1hi.arvadosapi.com/jobs/qr1hi-8i9sb-5v3jd4d1va26a0p

Example log:

```
2015-03-09_14:30:23 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 29 stderr crunchstat: mem 3624960 cache 25346048 swap 5014 pgmajfault 214380544 rss
2015-03-09_14:30:23 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 29 stderr crunchstat: cpu 442.5000 user 11.4500 sys 8 cpus -- interval 10.1232 seconds 4.79
00 user 0.0400 sys
2015-03-09_14:30:23 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 29 stderr crunchstat: blkio:202:32 0 write 37167104 read -- interval 10.1231 seconds 0 writ
e 446464 read
2015-03-09_14:30:23 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 29 stderr crunchstat: blkio:202:16 0 write 21766144 read -- interval 10.1231 seconds 0 writ
e 0 read
2015-03-09_14:30:23 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 29 stderr crunchstat: net:eth0 201676172 tx 1899475 rx -- interval 10.1232 seconds 0 tx 0 r
x
2015-03-09_14:30:23 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 26 stderr crunchstat: mem 3215360 cache 64135168 swap 9609 pgmajfault 655527936 rss
2015-03-09_14:30:23 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 26 stderr crunchstat: cpu 1021.2800 user 34.5800 sys 8 cpus -- interval 10.0002 seconds 3.3
500 user 1.2500 sys
2015-03-09_14:30:23 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 26 stderr crunchstat: blkio:202:32 0 write 71942144 read -- interval 10.0002 seconds 0 writ
e 4063232 read
2015-03-09_14:30:23 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 26 stderr crunchstat: blkio:202:16 0 write 31727616 read -- interval 10.0002 seconds 0 writ
e 0 read
2015-03-09_14:30:23 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 26 stderr crunchstat: net:eth0 1291918100 tx 4381065 rx -- interval 10.0001 seconds 2838390
02 tx 648240 rx
2015-03-09_14:30:23 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 24 stderr crunchstat: mem 3481600 cache 68702208 swap 31962 pgmajfault 525033472 rss
2015-03-09_14:30:23 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 24 stderr crunchstat: cpu 1408.4100 user 49.9600 sys 8 cpus -- interval 9.9997 seconds 6.16
00 user 0.1400 sys
2015-03-09_14:30:23 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 24 stderr crunchstat: blkio:202:16 0 write 4444160 read -- interval 9.9996 seconds 0 write
77824 read
2015-03-09_14:30:23 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 24 stderr crunchstat: blkio:202:32 0 write 288575488 read -- interval 9.9996 seconds 0 writ
e 176128 read
2015-03-09_14:30:23 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 24 stderr crunchstat: net:eth0 1612714910 tx 5267991 rx -- interval 9.9996 seconds 0 tx 0 r
x
2015-03-09_14:30:24 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 30 stderr crunchstat: mem 3399680 cache 25018368 swap 5972 pgmajfault 275226624 rss
```

```
2015-03-09_14:30:24 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 30 stderr crunchstat: cpu 517.9000 user 18.2500 sys 8 cpus -- interval 10.4004 seconds 5.33
00 user 0.0800 sys
2015-03-09_14:30:24 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 30 stderr crunchstat: blkio:202:32 0 write 42160128 read -- interval 10.4003 seconds 0 writ
e 0 read
2015-03-09_14:30:24 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 30 stderr crunchstat: blkio:202:16 0 write 770048 read -- interval 10.4003 seconds 0 write
0 read
2015-03-09_14:30:24 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 30 stderr crunchstat: net:eth0 403174439 tx 2484773 rx -- interval 10.4003 seconds 0 tx 0 r
x
2015-03-09_14:30:25 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 31 stderr crunchstat: mem 3248128 cache 34291712 swap 1019 pgmajfault 243732480 rss
2015-03-09_14:30:25 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 31 stderr crunchstat: cpu 382.0000 user 13.2300 sys 8 cpus -- interval 10.0001 seconds 4.07
00 user 0.0500 sys
2015-03-09_14:30:25 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 31 stderr crunchstat: blkio:202:32 0 write 7876608 read -- interval 10.0001 seconds 0 write
 212992 read
2015-03-09_14:34:59 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 33 stderr srun: error: Node failure on compute19
2015-03-09_14:34:59 qr1hi-8i9sb-7jc0nde0tqv3u6y 18871  backing off node compute19 for 60 seconds
2015-03-09_14:34:59 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 32 stderr srun: error: Node failure on compute19
2015-03-09_14:34:59 qr1hi-8i9sb-7jc0nde0tqv3u6y 18871  backing off node compute19 for 60 seconds
2015-03-09_14:34:59 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 26 stderr srun: error: Node failure on compute19
2015-03-09_14:34:59 qr1hi-8i9sb-7jc0nde0tqv3u6y 18871  backing off node compute19 for 60 seconds
2015-03-09_14:34:59 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 34 stderr srun: error: Node failure on compute19
2015-03-09_14:34:59 qr1hi-8i9sb-7jc0nde0tqv3u6y 18871  backing off node compute19 for 60 seconds
2015-03-09_14:34:59 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 30 stderr srun: error: Node failure on compute19
2015-03-09_14:34:59 qr1hi-8i9sb-7jc0nde0tqv3u6y 18871  backing off node compute19 for 60 seconds
2015-03-09_14:34:59 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 24 stderr srun: error: Node failure on compute19
2015-03-09_14:34:59 qr1hi-8i9sb-7jc0nde0tqv3u6y 18871  backing off node compute19 for 60 seconds
2015-03-09_14:34:59 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 31 stderr srun: error: Node failure on compute19
2015-03-09_14:34:59 qr1hi-8i9sb-7jc0nde0tqv3u6y 18871  backing off node compute19 for 60 seconds
2015-03-09_14:34:59 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 29 stderr srun: error: Node failure on compute19
2015-03-09_14:34:59 qr1hi-8i9sb-7jc0nde0tqv3u6y 18871  backing off node compute19 for 60 seconds
2015-03-09_14:34:59 qr1hi-8i9sb-7jc0nde0tqv3u6y 18871 24 child 1486 on compute19.2 exit 0 success=
2015-03-09_14:34:59 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871 24 failure (#1, temporary ) after 3006 seconds
2015-03-09_14:34:59 qr1hi-8i9sb-7jc0nde0tqv3u6y 18871 24 task output (0 bytes):
2015-03-09_14:34:59 qr1hi-8i9sb-7jc0nde0tqv3u6y
 18871  Every node has failed -- giving up on this round
```

**Subtasks:**

| | |
|---|---|
| Task # 5509: Review 5425-set-docker-memory-limits | **Resolved** |

**Related issues:**

| | | |
|---|---|---|
| Related to Arvados - Feature #5540: [Crunch] Contain all job/task containers ... | **Closed** | **03/23/2015** |

### Associated revisions

**Revision 1fea1b7a - 03/23/2015 05:47 PM - Peter Amstutz**

Merge branch '5425-set-docker-memory-limits' refs #5425

### History

**#1 - 03/09/2015 03:05 PM - Peter Amstutz**

*- Subject changed from [Crunch] Node failure to [Crunch] SLURM Node failure*

#### #2 - 03/09/2015 03:08 PM - Peter Amstutz

*- Description updated*

#### #3 - 03/09/2015 03:12 PM - Peter Amstutz

*- Description updated*

#### #4 - 03/11/2015 08:27 PM - Peter Amstutz

*- Target version changed from Bug Triage to 2015-04-01 sprint*

#### #5 - 03/11/2015 08:37 PM - Peter Amstutz

Set memory limit per docker container.  (total memory / number of tasks)

#### #6 - 03/18/2015 06:51 PM - Peter Amstutz

*- Category set to Crunch*

*- Assigned To set to Peter Amstutz*

#### #7 - 03/19/2015 02:49 PM - Peter Amstutz

*- Status changed from New to In Progress*

#### #8 - 03/23/2015 03:11 PM - Peter Amstutz

Some notes relating to 5425-set-docker-memory-limits

This uses docker run --memory=xxx.  Based on some experimentation, it turns out this limits the resident set size of the container, but not the total virtual memory available to the application.  This has a  couple of implications:

- If the container exceeds the limit set by --memory, it won't automatically fail, it will just spill over into swap.
- Since the goal is leave some breathing room for system process (the SLURM daemon, arv-mount, sshd, etc), instead of oversubscribing the task containers, it makes more sense to allocate (RAM / tasks) * 95% to reserve 5% of RAM for stuff outside the container.
- If we wanted to let the tasks compete for memory but still cap the memory usage to leave room for the host processes, we would have to put all the task containers in one big parent container with a memory limit set.  I'm not sure of the best way to do that.

#### #9 - 03/23/2015 04:42 PM - Tom Clegg

Conservative memory limits will cause jobs to run unnecessarily slowly:

- when a worker runs 4 tasks, 3 of them finish and 1 keeps running → 3/4 RAM is free while the 1 running task is swapping.
- when concurrent tasks actively use a lot of RAM, but not all at the same time → some RAM is idle while other tasks swap.

This is suboptimal, but it is a step toward using a single worker node to run multiple [users'] jobs at the same time, which is good.

Combined with better reporting/graphing of swap activity, this should also help users determine the optimal memory/concurrency profile for their jobs (without going as far as killing them off when they hit some hard vsize limit).

#### #10 - 03/23/2015 05:00 PM - Tom Clegg

The code in 5425-set-docker-memory-limits lgtm @ c96d48a.

However, I've tried some tests locally and setting --memory seems to kill my processes when their swapped-out memory size reaches {somewhere between 1x and 2x RSS}, instead of just letting them swap. Is this expected?

#### #11 - 03/23/2015 05:33 PM - Tom Clegg

I see. Docker 1.2 defaults to limiting swap to 2x given --memory, but doesn't yet have the --memory-swap=-1 option mentioned in the docs that could override that.

I'm OK with trying the default swap limit. If users actually need more swap, I guess we'll just have to [bump docker dependencies and] add a --memory-swap=-1 there.

#### #12 - 03/23/2015 05:37 PM - Peter Amstutz

Tom Clegg wrote:

> This is suboptimal, but it is a step toward using a single worker node to run multiple [users'] jobs at the same time, which is good.

I agree and that's why I made a note about some kind of container-inside-container resource allocation might eventually be a better solution. However the specific problem I'm trying to solve here is that the entire node can fall over due to resource exhaustion, in which case it is probably better for the job to run slowly than to mysteriously blow up with a SLURM node failure.  There is definitely more that we will want to do in to improve

reporting for resource usage (like "I noticed your job has lots of major page faults, maybe you should lower the number of concurrent tasks").

> The code in 5425-set-docker-memory-limits lgtm @ c96d48a.
>
> However, I've tried some tests locally and setting --memory seems to kill my processes when their swapped-out memory size reaches {somewhere between 1x and 2x RSS}, instead of just letting them swap. Is this expected?

According to https://docs.docker.com/reference/run/#runtime-constraints-on-cpu-and-memory it suggests that doesn't limit swap usage unless you explicitly say so, but this seems to be the Docker 1.5 documentation and we're using Docker 1.3 which appears to not have an option to limit swap usage, so I'm not clear on what the Docker 1.3 behavior is.

#### #13 - 03/31/2015 03:17 PM - Peter Amstutz

*- Status changed from In Progress to Resolved*

Re-ran the original job, which completed successfully.  Marking this as resolved.