Server Woes and ZFS Performance

What happened, why we think it happened, and what we did about it once it had happened

Thanks / Credits

- Thanks to
  - Allan Jude
  - Michael W. Lucas
  - Matt Ahrens
  - …and other random twitter users
  - Jonathan Stewart, and Les.net

How this all started

- The server has 12 x 4TB HDDs arranged in a ZFS RAIDZ2 array, and
- 2 x 256GB SSDs running both Linux md(8) RAID for boot, and two ZFS partitions each for both write cache (SLOG, aka ZIL) and read cache (L2ARC).
- The server was connected to the ISP (Les.net) using 2 x 10GE interfaces, and
- 1 (out of 2) 1Gbps ethernet ports to a local hardware firewall protecting the out-of-band mgmt. ports

How this all started, continued

- …and the auto-updates had been broken for a while, without anyone noticing or fixing them.
  - (oops)
- AND the server hadn't been rebooted in quite a while.
- So when I finally ran updates, we jumped multiple kernel (minor) versions...
  - straight into the post-regression-failure era for the bnx2x driver. ☹
  - and also erased the old, working, kernel, only keeping the two most recent. &^%$#@!#$%$#@!!!!

Mucking with kernels

- I wanted to avoid rolling my own kernel, as Debian applies a significant number of patches to upstream, none of whose importance I can guess.
  - But I'll betcha at least one of them is important!
- Luckily, Debian Backports provides the working kernel versions, so 5.3 and 5.4 were both available.
- Much updating of toolchain and supporting packages was required to get 5.3 running, 5.4 was impossible.

Mucking with kernels, continued

- Once everything had leveled up to 5.3, it was then possible to advance to 5.4.
- 5.3 did not fix the hard-lockup bug in bnx2x
- Neither did 5.4. ☹
- Newer firmware is available, but can’t figure out how to load it – the firmware version string is hardcoded into the module!

Kernel status

- Muug.ca now runs 5.4.
- ZFS modules are not 100% supported on 5.4 yet, but close enough that they work.
  - Only one error is logged at module-load time, and it seems to be harmless.
- I guess this is still an improvement, overall… *sigh*

ZFS problems, 1

- While we were looking at the kernel problems, we noticed that average network throughput on our 20Gbps link was only ~200Mbps.
- What happened to the other 99.9% of theoretically-available performance?
- I’m personally connected (at work) to Muug.ca via Les.net and MBIX at 10Gbps, so watching my VMs update at modem speeds *really* ticks me off!

ZFS problems, 2

- Local testing verified that the filesystem was the problem, not the network stack.
  - mlocate’s updatedb(8) could take >24hrs to run!
- Hundreds upon hundreds of processes in IOWAIT caused load averages > 1000. You read that right. The 1-minute loadavg was regularly in excess of 800.
  - This in turn caused Sendmail to pause itself, in order to “save” the server. Except sendmail was running from SSDs, which were not IO-bound, so all it accomplished was to randomly stop delivering mail.

Apache & mod_php

- Load average is calculated by counting the # of processes waiting on the kernel’s run queue. This includes processes in IOWAIT.
- Apache is a preforking server, one process per connection.
- Lots and lots of connections = lots and lots of proceses = silly load average numbers.
- Mod_php is no longer best practice, hasn’t been for years both for performance and security reasons. (Nor are SSI documents, let’s just ignore that for now, though.)

Nginx and PHP-FPM

- To reduce the load average by leaving the prefork model means switching to php-fpm anyway.
- We've tried a few times to ditch Apache for the much more efficient Nginx, maybe we can succeed this time?
- Success. And implementation of the QoS module in nginx, too, to mitigate people hammering the ^%$# out of the server with hundreds of concurrent processes.
- Some SSI converted to PHP. Many PHP uplifts (some of this code was written for PHP3) were required.

Another network oops

- To keep muug.ca online at <u>all</u>, traffic was re-routed through the hardware firewall protecting the out-of-band interface.
  - A pfSense/Netgate SG-2220
- This worked, but for one thing…
- The Management network at Les.net is 100Mbps, not 1Gbps.
- Oops. No wonder it's so slow, now…
- But at least it's online?

Network solution, for now

- Les.net made available to us (at no charge) a 1Gbps ethernet port
- Muug.ca now runs off that 1Gb ethernet port, until the 10GE ports become usable again.
- The management router is no longer pushing 99.9Mbps 24x7, which makes Les.net's operation staff a bit happier.
  - Turns out **anything** running 24x7 at 99.9% triggers various alarms, even if it's relatively harmless.

ZFS problems, 3

- Around the same time, one of the SSDs started alerting on a SMART pre-failure condition.
- The other SSD was showing some concerning SMART stats, even though it hadn't declared a pre-failure condition yet.
- One SSD was replaced, and a replacement for the other one was acquired (in advance, this time).
- These were already the 2nd or 3rd set of SSDs we'd gone through!

ZFS problems, 4

- The SLOG was mirrored across both SSDs, so both got the same write load applied to them.
  - SLOG is the correct term for a ZIL when the ZIL is on a separate device. Most people still call it a “dedicated ZIL” or something like that.
- The Intel SSD indicated that we’d written some unbelievable number of terabytes to a 256GB drive.
- SLOG is a very, very busy partition, it turns out.
- And it’s not needed *at all* for datasets where you can afford to lose ~5sec of data.
  - The array now runs in the ZFS equivalent of “async” mode, so it’s now only about as reliable as ext4 in the face of power outages or crashes.

ZFS problems, 5

- Remember we had both SLOG and L2ARC on the same set of SSDs?
  - Turns out that was dumb.
    - It seemed like a good idea at the time…

- Oh, and L2ARCs are deliberately throttled after a reboot, in order to not overwhelm them when they are “really fast” 15K RPM SAS or SCSI drives fronting an array of “really slow” SATA or SCSI 5400rpm drives.
  - But these are SSDs, they can take a lot more (random) writes in the same period of time…
    - There’s a tunable knob for that!

ZFS Overkill, 1

- In order to save our SSDs, we eliminated both the L2ARC and the SLOG from them.
  - This made ZFS reallllllllllllly slow. There's some not-too-hard math that explains exactly why, but essentially high-spindle-count is good for capacity, not speed.
    - We probably should've done RAID 60 instead of RAID 6. Oh, well… too late to change our minds about it now.
- But we need that L2ARC cache, even if "sync=disabled" eliminated the need for the SLOG

ZFS Overkill, 2

- So the server has no free drive bays whatsoever. There isn't enough room *or* ports to float another SSD inside the case. (It's a 1U case.)
- But wait, there's a single PCIe slot with riser card, surely we can do something with that?
- We can – NVMe!
- But the board doesn't support NVMe…

ZFS Overkill, 3

- Turns out NVMe (in the M.2 form factor) only needs motherboard support to <u>boot</u>. We aren't booting from it.
- Obtained an M.2 to PCIe x8 adapter online, obtained a ~1TB NVMe drive online, combine the two, insert into slot… and nothing.
- Well, shoot.

ZFS Overkill, 4

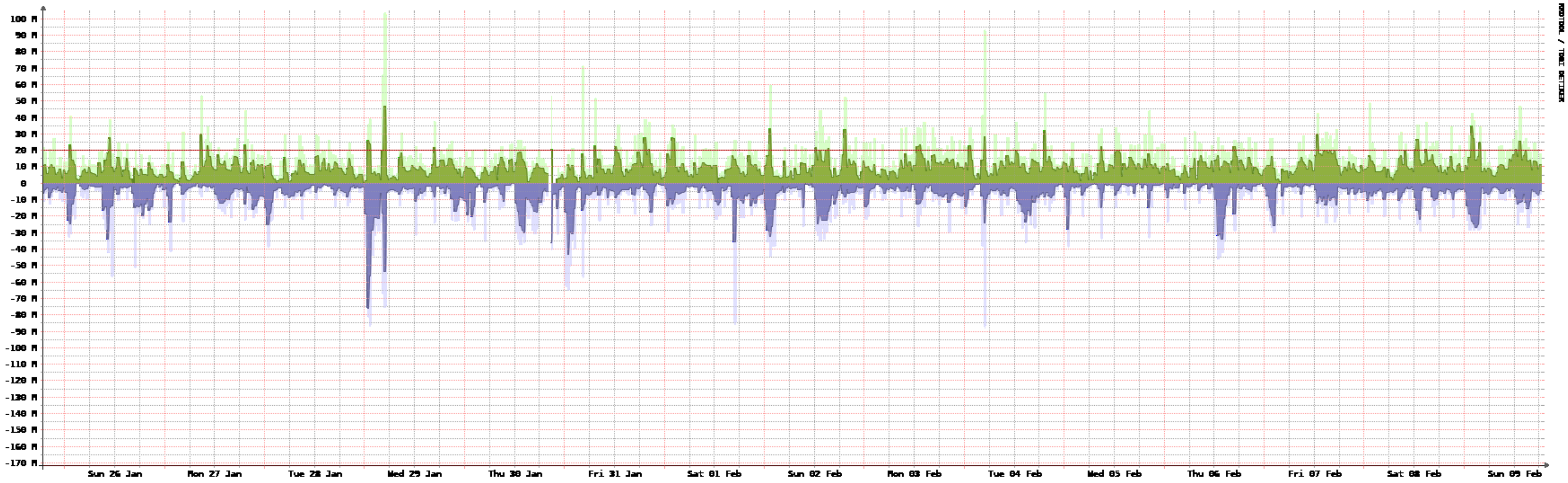
- After discovering there’s an NVMe kernel module that needs to be loaded, it got loaded.
- After discovering that NVMe devices have ***weird*** device naming conventions, I found the device.
- …and partitioned it.
- …and turned it into L2ARC for the main ZFS array
- ZZZZOOOOOOOOOMMMMMM!

ZFS Overkill, 5

- Well, almost. Many ZFS tunables had to be tuned:
  - l2arc_headroom=100
  - l2arc_noprefetch=0
  - l2arc_write_boost=1073741824
  - l2arc_write_max=1073741824
  - zfetch_array_rd_sz=1073741824
  - zfetch_max_streams=24
  - zfs_arc_grow_retry=1
  - zfs_arc_max=25769803776
  - zfs_arc_meta_limit=25769803776
  - zfs_arc_meta_prune=1
  - zfs_dedup_prefetch=1
  - …35 different settings, in total!

ZFS Results, 1

- NVMe Disk I/O in bps:

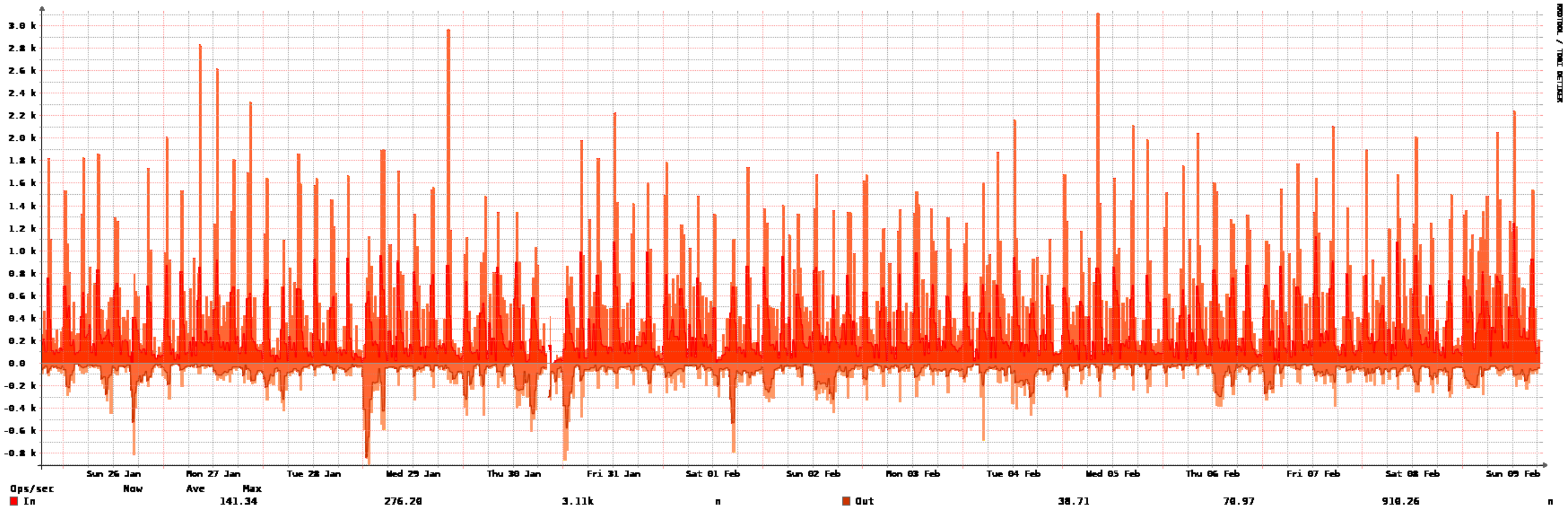


```
bps      Now      Ave      Max      95th %
🟩 In   11.42M   10.04M   103.67M  20.22Mn  🟦 Out    4.73M    6.86M    88.20M    21.45Mn   Total   21.87TB   (In   12.99TB   Out   8.88TB)
```

- Not too impressive, really…

ZFS Results, 2

- NVMe Disk I/O in IOPS:

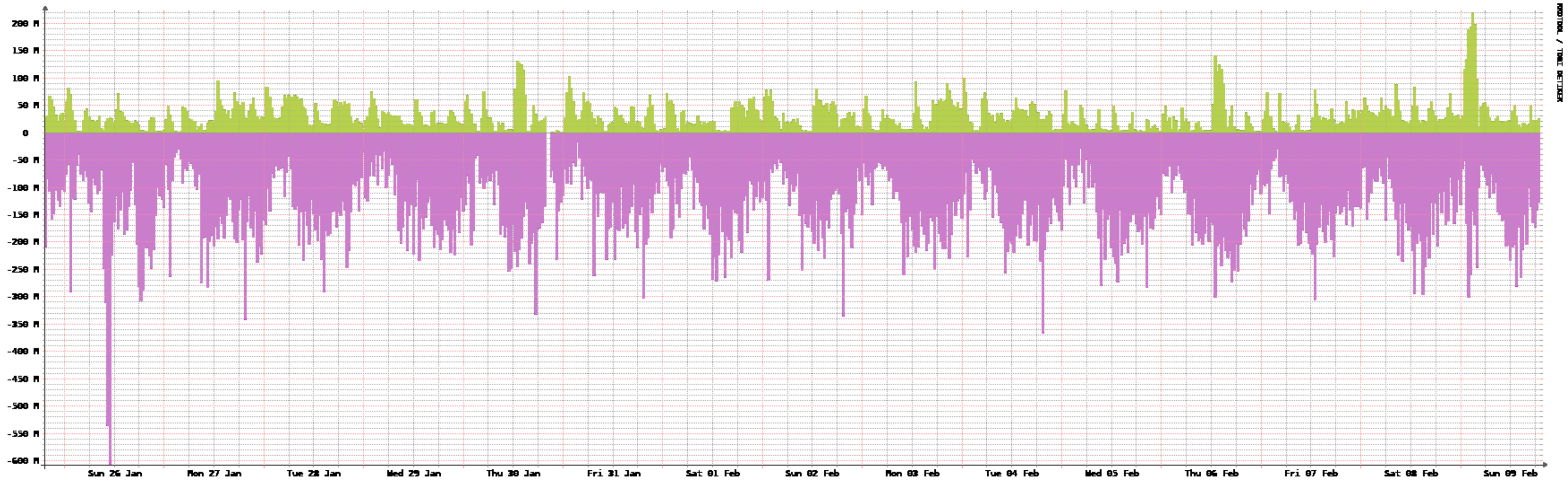


```
Ops/sec       Now      Ave      Max
🟥 In        141.34   276.20   3.11k   n   🟥 Out   38.71   70.97   910.26   n
```

- What was that about unimpressive, again?

Overall throughput

- All ports and all traffic, combined:



- Not too bad, not awesome either.
- Not really sure why it isn't better…

LibreNMS

- Pause to demonstrate LibreNMS

Results

- Server still can’t use the 2x10GE ports, using 1x1Gbps for now. Not a major bottleneck.
- Server monitoring now exists in far greater breadth & depth than ever before.
- Holy cow did ZFS ever get a lot faster!
- We’re no longer murdering our SSDs on a semi-regular basis now.
- We spent a little bit of money.

Image credit: Charlie Cottrell, https://xeyeti.com/, https://charliecottrell.com/