Qualcomm's Journey with SUSE: The Value of Enterprise Linux

CAS1057

Mike Marion – IT Engineer, Sr Staff, Qualcomm Incorporated

Gokhan Cetinkaya – Sales Engineer, SUSE

Introductions

Gokhan Cetinkaya

- Gokhan Cetinkaya is a sales engineer at SUSE, based in San Diego, California.
- He supports sales initiatives through client-specific solution design and technical consultations.
- Over the past 15 years, he has held roles in training, systems administration and pre-sales engineering.
- He started working for SUSE in 2013 in Istanbul, Turkey. He was offered a position in the United States two years later.
- Passionate about gaming since the Commodore 64 and the Amiga.
- You can reach him on LinkedIn: www.linkedin.com/in/gcetinkaya

Introductions

Mike Marion

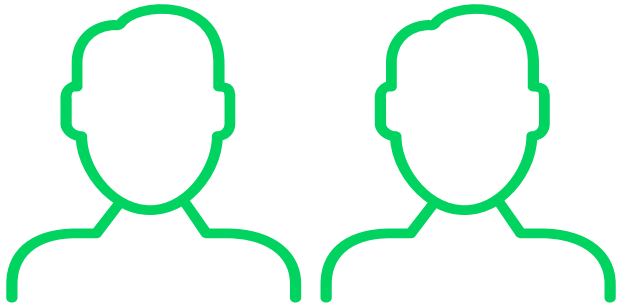
- UNIX SysAdmin since 1997
- Started on a nearly all Sun Solaris compute farm
- Ran Linux at home and did a lot of experimenting there at the same time
- In the early 2000s, Linux usage started in EDA compute
- Shortly after Novell purchased Suse (2003), we started using SLE in our compute farm
- Early growing pains both with EDA ISVs not supporting Linux or only supporting one specific distro

Introductions

Why are we here to talk?

Our journey with SUSE began with SUSE 9 and has continued through various adventures to today.

We are here to discuss our personal tales and successes that the relationship between our two companies have provided.



Introductions

Before SUSE…

We were a ~99% Solaris (on SPARC) shop

Some EDA ISVs started saying to run their tools on x86

We created our own Linux imaged based off a well-known distro

- There was no “Enterprise” distribution at the time
- Multiple teams worked on various “images” at the company
- Early installs were started via CD/DVD, eventually USB… PXE years later
- We were on our own!

When x86_64 came out, the first real Enterprise offerings followed

- The initial cost for one was very high

SUSE had just been purchased by Novell

- SUSE/Novell were interested in getting a foot in the door with customers doing EDA work
- Were open to licensing the OS separately from the support
- SUSE, along with us and other vendors, were able to get the ISVs to support SLE

Introductions

Partnership grows

From the start, our biggest issue was related to bugs in autofs

- Details to follow
- The largest source of open cases with SUSE by far

We dig into issues a lot before opening a case with SUSE

- We've learned what questions SUSE support will ask, so we include a lot of data when we open a case
- Supportconfigs, strace/core dumps, detailed examples
- A way to re-create the bug on demand whenever possible
- Our due diligence often saves SUSE a lot of time in digging into the issue
- We work on bugs as a team, often taking their engineering questions/suggestions and testing internally while they work in their labs. It's a partnership, versus us simply handing off.

Foundation for Success: Building on Our Relationship

Foundation for Success

Early autofs growing pains

autofs

Foundation for Success

Early autofs growing pains

Before autofs4 (and Ian Kent taking over as maintainer)

- No direct map support
  - Engineers were used to paths on Solaris, which supported direct maps
  - We needed to recreate the same paths on Linux to support existing flows
- Had to use shell-based maps
  - This resulted in a daemon spawned for every direct path traversed that was not a mount point
  - OOMs or crashes would kill various such daemons, paths would become inaccessible
  - Could not ls to see available paths (ghost) and must simply know the paths
- ARG_MAX issue
  - Solaris was 2M, Linux was hard-coded to 128k
  - “Solved” by getting engineering to shorten variables from using longer /usr/local/nfs paths to /nfs link/path

```
kvmautofsprog(sles11sp4) [tmp]$ ps -eo pid,ppid,comm | awk '($NF=="automount"){print $2}'  |
sort | uniq -c
    110 1


/usr/sbin/automount --submount /nfs program /etc/auto.nfs.sh proto=tcp port=2049 -
Dpref="baz" -Dpref="bar" -Dpref="foo"
/usr/sbin/automount --submount /nfs/foo program /etc/auto.nfs.sh proto=tcp port=2049 -
Dpref="baz" -Dpref="bar" -Dpref="foo"
/usr/sbin/automount --submount /nfs/foo/bar program /etc/auto.nfs.sh proto=tcp port=2049 -
Dpref="baz" -Dpref="bar" -Dpref="foo"
/usr/sbin/automount --submount /nfs/foo/bar/baz program /etc/auto.nfs.sh proto=tcp port=2049
-Dpref="baz" -Dpref="bar" -Dpref="foo"
/usr/sbin/automount --submount /nfs/foo/bar/baz/sandiego program /etc/auto.nfs.sh proto=tcp
port=2049 -Dpref="sandiego" -Dpref="baz" -Dpref="bar" -Dpref="foo“
```

```
root     23543  0.0  0.0  16740   920 ?        S    12:23   0:00 /usr/sbin/automount --
submount /nfs/foo/bar program /etc/auto.nfs.sh proto=tcp port=2049 -Dpref="bar" -Dpref="foo"

kvmautofsprog(sles11sp4) [tmp]$ sudo kill -9 23543

kvmautofsprog(sles11sp4) [tmp]$ ls -al /nfs/foo/bar/test
ls: cannot access /nfs/foo/bar/test: No such file or directory

/usr/sbin/automount /nfs program /etc/auto.projects.sh proto=tcp,port=2049
/usr/sbin/automount --submount /nfs/foo program /etc/auto.projects.sh proto=tcp port=2049 -
Dpref="foo"

automount(pid21675) /nfs autofs
rw,relatime,fd=4,gid=1,pgrp=21675,timeout=300,minproto=2,maxproto=4,indirect 0 0
automount(pid23535) /nfs/foo autofs
rw,relatime,fd=4,gid=1,pgrp=21675,timeout=300,minproto=2,maxproto=4,indirect 0 0
automount(pid23543) /nfs/foo/bar autofs rw,relatime,fd=-
1,gid=1,pgrp=21675,timeout=300,minproto=2,maxproto=4,indirect 0 0
automount(pid24962) /nfs/foo/sysadmin autofs rw,relatime,fd=-
1,gid=1,pgrp=21675,timeout=300,minproto=2,maxproto=4,indirect 0 0
```

Foundation for Success

Autofs4

Early attempts at direct maps mounted /usr/local/nfs/ base over /usr

- The logic saw /usr as the least common denominator
- Mounting over /usr is bad

Migrated to LDAP due to issues with large file maps and autofs4 (interesting bugs otherwise)

- We have stuck with LDAP since then

We saw an extremely rare bug with a broken mount

- For example, maps have /prj/foo/bar and /prj/foo/baz
- Somehow, /prj/foo/baz file mount point is mounted on /prj/foo
- Extremely rare (~a couple dozen cases found)
- Never could re-create to give case to SUSE to troubleshoot

Foundation for Success

Things got better with autofs5

Before autofs5, the automounter was often the cause of issues

- Users would often blame things on it that had nothing to do with it

Direct map support

- Also supports large file-based maps

Examples of bugs we encountered and that SUSE fixed

- Direct mounts would stop expiring, but new mounts would work and indirects would expire
- Rarely: a mount would get umounted, but automount was not aware
- Every re-read from every client is a full dump of the maps configured in auto.master

Foundation for Success

Things got better with autofs5

Huge maps can have issues

- Performance on wake and check for busy/umount
  - ¼ $DEFAULT_TIMEOUT value: wake, look for busy paths, umount those possible, perhaps re-read maps
  - We actually bumped from default of 10 minutes to 4 hours
  - Larger maps (and actual NFS mounts) means more CPU used per wake
  - Jobs might mount a path and not use it again for some time, default 10 minutes would sometimes umount out from under a job still running
- Every re-read from every client is a full dump of the maps configured in auto.master

Foundation for Success

Audits to watch for issues we had over time

Older autofs had issues with too many open fds

- Once limit hit, a full restart was required (usually just rebooted)
- Not a problem for some time now (SUSE fixed)

Nested mounts

- Sadly, even autofs5 can’t handle (though it’s PEBCAK)
- We’ve put seatbelts in place to avoid them getting into maps, some sysadmins still manage
- Cleanup once they hit host

Foundation for Success

Audits to watch for issues we had over time

Look for direct mounts not expiring

- SLE11.4 sometimes stops expiring direct mounts
- Audit looks for 100s of NFS mounts, does a SIGUSR1 to automount to trigger an expiration run
  - Daemon usually seems to “wake up” and expire on it’s own again
- Problem not seen on SLE12 to date

Find/fix vanished mounts

- Look in syslog for a path last logged by automount as NFS mounted but not in mount table
- Fix is to umount the autofs cache ‘mount’ and kick a re-read to pull back in
- Problem not seen on SLE12 to date

Foundation for Success

Audits (continued)

Cwd -> (deleted)

- Not sure if autofs or NFS itself
- Can’t recreate on demand, so difficult for SUSE engineers to debug
- Processes stuck on loop: getpwd() = ENOENT
- On host, doing sudo ls –l /proc/$pid/cwd shows (deleted)
- If another process does an ls of the dir, something kicks the VFS layer into “seeing” the path again and the process finally gets a valid response to getpwd() and continues
- Difficult to be sure that it’s a problem in all cases because users sometimes do it themselves

Not an autofs problem, but look for option/filer mismatches

- Sometimes paths are held busy long enough or haven’t expired when maps are updated
- Audit finds those and either kicks path (if possible) to remount with updated values or closes hosts to compute jobs and reboots when possible

Foundation for Success

The introduction of systemd with SUSE 12

systemd startup/shutdown times (until suse patched)

- The change of /etc/mtab -> /proc/self/mounts coincided with systemd
- This change has bad impacts for larger sets of direct maps
- Initial testing on SLE12 had autofs time out
- The following shows startup times for automount from various sizes of direct map sources
  - `time systemctl start autofs`

# of Direct mounts in map	Time to start up Stock autofs		
100	0m0.095s		
500	0m0.572s		
1000	0m2.163s		
1500	0m5.887s		
2000	0m12.887s		
2500	0m28.863s		
5000	3m36.023s		
7500	13m5332s		
10000	27m10.645s		
15000	88m12.085s		
20000	208m20.439s		
30000	724m58.740s		
40000	2223m59.755s		
50000	4954m57.003s		
60000	10004m17.733s		

# of Direct mounts in map	Time to start up Stock autofs	Time to start up Stock autofs (human)	
100	0m0.095s	0.095 Sec	
500	0m0.572s	0.572 Sec	
1000	0m2.163s	2.163 Sec	
1500	0m5.887s	5.887 Sec	
2000	0m12.887s	12.887 Sec	
2500	0m28.863s	28.863 Sec	
5000	3m36.023s	3 Min 36.023 Sec	
7500	13m5332s	13 Min 5.332 Sec	
10000	27m10.645s	27 Min 10.645 Sec	
15000	88m12.085s	1 Hours 28 Min 12.085 Sec	
20000	208m20.439s	3 Hours 28 Min 20.439 Sec	
30000	724m58.740s	12 Hours 4 Min 58.740 Sec	
40000	2223m59.755s	1 Days 13 Hours 3 Min 59.755 Sec	
50000	4954m57.003s	3 Days 10 Hours 34 Min 57.003 Sec	
60000	10004m17.733s	6 Days 22 Hours 44 Min 17.733 Sec	

# of Direct mounts in map	Time to start up Stock autofs	Time to start up Stock autofs (human)	Time to start up patched
100	0m0.095s	0.095 Sec	0.054s
500	0m0.572s	0.572 Sec	0.180s
1000	0m2.163s	2.163 Sec	0.279s
1500	0m5.887s	5.887 Sec	0.379s
2000	0m12.887s	12.887 Sec	0.555s
2500	0m28.863s	28.863 Sec	0.650s
5000	3m36.023s	3 Min 36.023 Sec	1.189s
7500	13m5332s	13 Min 5.332 Sec	2.811s
10000	27m10.645s	27 Min 10.645 Sec	4.957s
15000	88m12.085s	1 Hours 28 Min 12.085 Sec	8.594s
20000	208m20.439s	3 Hours 28 Min 20.439 Sec	13.113s
30000	724m58.740s	12 Hours 4 Min 58.740 Sec	22,155s
40000	2223m59.755s	1 Days 13 Hours 3 Min 59.755 Sec	36.241s
50000	4954m57.003s	3 Days 10 Hours 34 Min 57.003 Sec	55.446s
60000	10004m17.733s	6 Days 22 Hours 44 Min 17.733 Sec	1m3.511s

Last Startup Datapoints

Mount rates

```
count=1055 change=359 359/s
count=5002 change=5 5.0000/s
count=10009 change=11 11/s
count=20034 change=50 .8333/s
count=30001 change=12 .2000/s
count=40003 change=5 .0833/s
count=50001 change=2 .0333/s
count=59998 change=4 .0333/s
```

Foundation for Success

A few notes on autofs to share

Autofs on SLE12.3+ now very stable

Logging suggestion

- /etc/sysconfig/autofs
- # DEFAULT_LOGGING - set default log level "none" "verbose" or "debug"
- None (default) setting is essentially nothing but automount start/stop
- verbose logs helpful information, showing when direct mounts are added/removed and all NFS mount/expire operations – comes in very handy
- debug is extremely noisy and should only be used when tracking down a known issue
  - Log output grows by about 10x

Two other options

- TIMEOUT
- LOCAL_OPTIONS="proto=tcp,port=2049"

Foundation for Success

Sampling of issues other than autofs

Foundation for Success

Other support issues/bugs with Suse

Stale NFS Handle triggered when reusing deleted paths

- While it seems like a bad idea, engineers would often rm –rf a work directory and then remake it with new jobs shortly after
- Resulting jobs would get back an ESTALE and crash
- SUSE found the bug was due to a backport and fixed within a few days

sssd ignores group nesting level directive

- Co-worker found it while working on ARM development
- It was a big performance hit for group names that weren't listed in /etc/groups and pulled from AD

Foundation for Success

Other support issues/bugs with Suse

Latest xorg-x11 lib updates cause blank screen (SLE11.4)

- Updates broke some X11 programs
- Matlab would open and remain blank
- Suse fixed in < 2 weeks (most of delay with getting a test copy of matlab that showed the bug)

Kernel backport request related to GPFS testing

- Co-worker doing GPFS testing ran into performance issues and found some info on google
- Suse engineers actually found that the extra sync calls he had in his test scripts was triggering an issue in the kernel but wasn't something that effected normal use cases

OpenSSL bug triggering perl Net::SSLeay issue

- When the older SSL modes were mostly deprecated we ran into issues with the perl module
- The real "fix" was in the openssl1 updates from Suse, but those didn't link into the perl module
- Suse engineers came up with a patch to perl that we could apply to fix the issue

Foundation for Success

Other support issues/bugs with Suse

perl scripts executed using -x much slower

- Our newly developed (internal) imaging tool had an issue when parsing json for some steps
- Suse engineering helped pinpoint that the module being loaded in the installation ramdisk was a point rev behind what comes on SLE11.4 and our dev group fixed.
- Installs went from over an hour to ~20min

NFS clients experiencing TCP port starvation

- On some compute hosts, enough NFS mounts over time would cause port starvation
- Suse engineers pinned it down to the automounter doing an rpc call to ask server which port to use for mount calls
- Since we don’t change off the default port 2049, we simply set that in the sysconfig file and the problem went away

Foundation for Success

Other support issues/bugs with Suse

OpenAFS issue on ext3 and SLES10 SP3

- Suse doesn’t technically support OpenAFS
- Triggered with specific versions of OpenAFS when doing cache on ext3 FS
- Suse engineers still dug into the kdumps we gave them
- Turned out to be a security issue where users could create a DoS on any host quite easily
- Sine Nomine worked with us and Suse to fix the bug and rushed out a security fix before anyone found it in the wild

Foundation for Success

Other support issues/bugs with SUSE… systemd related

Server shutdown takes a very long time, due to large number of mounts

- On shutdown/reboot, the systemd wants to umount everything in /proc/mounts
- I came up with a tiny patch to systemd source and /usr/lib/dracut/modules.d/99shutdown/shutdown.sh
- SUSE came up with a much better patch to ignore autofs “mounts” on shutdown

dracut very slow when dealing with a large number of autofs mounts

- When doing patching or any rpm install, if Dracut is called it could take 20+ minutes
- SUSE did a patch similar to what was done with autofs and we had a PTF in < 2 days

Be Excellent!
and
Thank You!

Gokhan Cetinkaya
Sales Engineer, SUSE
gokhan.cetinkaya@suse.com
LinkedIn: gcetinkaya

Mike Marion
IT Engineer, Sr Staff, Qualcomm Incorporated
mmarion@qualcomm.com
LinkedIn: mmarion


SUSE
We adapt. You succeed.