gce-xfstests

Testing kernels using Google Compute Engine

tytso@google.com

Slides: http://thunk.org/gce-xfstests-lcna2016

Disclaimer

- The opinions expressed in this talk are my own, and do not necessarily reflect those of my employer.
- The xfstests-bld project, kvm-xfstests, and gce-xfstests are not a Google-supported project
  - Google supports me as an engineer; they don’t officially bless or sanction this project
  - Although I have gotten help from many other engineers both in and outside of Google
    - Thanks, all!

What is xfstests?

- A file system regression test suite written by SGI to test XFS
- Like XFS, originally was targeted for Irix
  - Later ported to Linux --- and now, only supported on Linux
  - (But it shouldn't be that hard to port xfstests to FreeBSD / OpenBSD / NetBSD)
- Now used to test all of the major file systems on Linux
  - xfs, ext2/3/4, cifs, btrfs, f2fs, reiserfs, gfs2, jfs, udf, nfs, tmpfs
  - (tmpfs support contributed by Google's Node Storage team)
- As ext4 maintainer I require all major changes be tested using xfstests before they are sent to me
  - Before pushing the ext4 tree to Linus, I run a full set of xfstests with multiple file system configurations --- took 22-23 hours using KVM on Goobuntu; now takes 7-8 hours using GCE using PD/SSD.

What is xfstests-bld?

- External, open-source project created by Yours Truly
- Originally designed as a hermetic build system for xfstests
  - xfstests required libraries that were packaged for Fedora, but not Debian
- Then extended to automate testing using KVM/qemu
- More recently, extended to do the testing in the cloud using GCE
- Also recently extended to run xfstests on Android

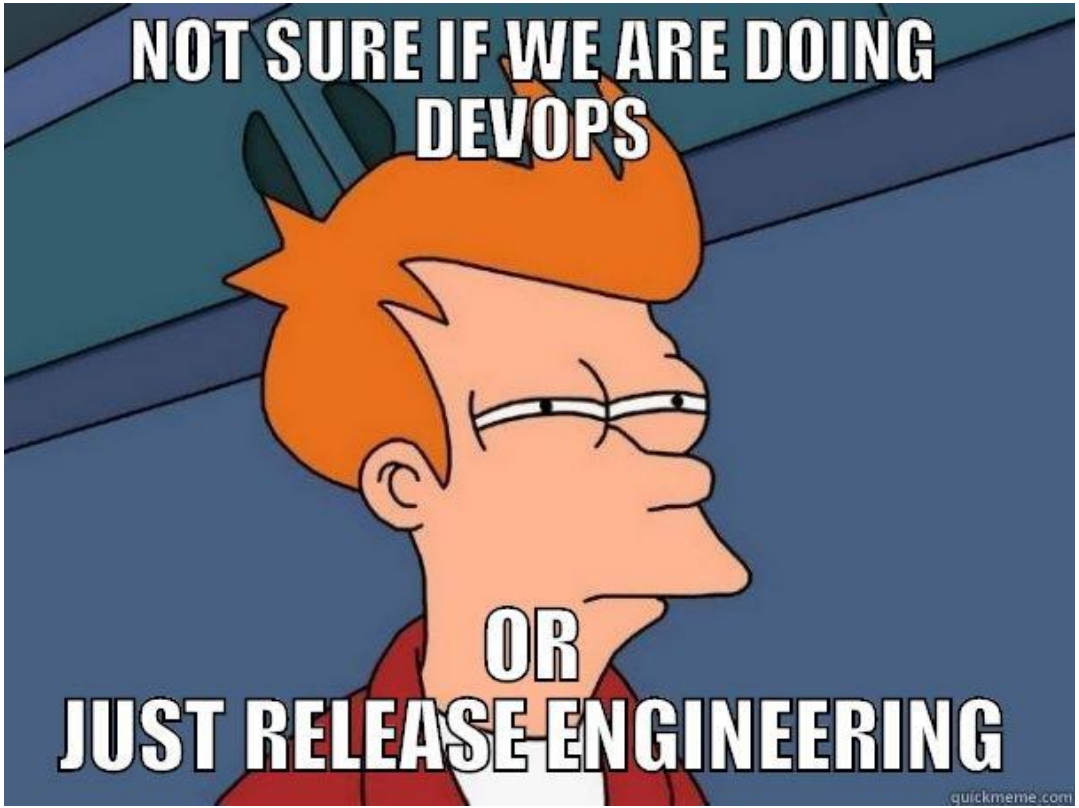
Why gce-xfstests?

- Faster than running the tests on your laptop
- You can run multiple tests in parallel on different VM's
  - Good for testing many stable kernels: 3.14.y, 3.18.y, 4.1.y, 4.4.y, etc.
  - Good for speeding up bisections
- Fire and forget
  - Launch a test using "gce-xfstests smoke" or "gce-xfstests full"
  - Suspend your laptop and go to lunch
  - Test report will be e-mailed to you when the test is completed
- Great way to learn more about using cloud VM's :-)

Interesting technology bits inside gce-xfstests (or, “I’m not a file system developer, why should I care?”)

- Automated, repeatable building of a GCE image
  - Hermetic build of the GCE image using: the Official Debian Jessie image, the xfstests binary tarball, and the gce-xfstests control scripts to create a test appliance VM image
- Sendgrid integration in order to send e-mail from a GCE VM
- Sample configs for building custom kernels for GCE
- Launching/updating kernels using kexec
- Example uses of various GCE offerings
  - Google Cloud SDK (the gcloud CLI)
  - Google Cloud Storage
  - Google Cloud Logging
  - Serial console integration
  - Local SSD

Building GCE images for xfstests



Building GCE images for xfstests

- Why use release automation? Important for many reasons
  - Traditional reasons for doing release engineering
    - Identifiability
      - Knowing all of the components used to make up release
    - Reproducibility
      - Repeatable integration of the components → operational stability
    - Consistency
      - Eliminating the “human factor” --- shouldn’t matter who builds the release
    - Agility
      - Enables continuous integration, “push on green”, etc.
  - GPL compliance
    - Using the latest hotness (Docker images, VM appliances) doesn’t eliminate the need to comply with the GPL --- you are still shipping binaries!

Building GCE images for xfstests

- Why use release automation? Important for many reasons
- Leveraging the Debian VM image
  - We could build an image from scratch, but it's faster not to
  - Debian does a great job creating a reliable, released VM image
    - Uses bootstrap-vz
    - Manifests used to control the build process are checked into git
    - https://wiki.debian.org/Cloud/GoogleComputeEngineImage
  - Debian keeps an archive of all of its binary and source packages
    - Technique used by kvm-xfstests
    - Use debootstrap using an a repository from snapshot.debian.org
      - E.g., http://snapshot.debian.org/archive/debian/20160808T105701Z/
      - Given a debian package version (use dpkg -l), you can find the sources corresponding to that binary package

Building GCE images xfstests

- Why use release automation? Important for many reasons
- Leveraging the Debian VM image
- Creating the payload
  - The xfstests-bld build system checks out a number of git trees
    - xfstests, xfsprogs, fio, quota
    - Generally because we want a newer version than what is in the distribution
  - The build is done in a hermetic build chroot created using debootstrap
  - The git commits used are stored in file which is included in xfstests.tar.gz
  - The payload is installed in the VM under /root/xfstests

```
...
quota           81aca5c (Tue, 12 Jul 2016 16:15:45 +0200)
xfsprogs        v4.5.0 (Tue, 15 Mar 2016 15:25:56 +1100)
xfstests-bld    7452b79 (Mon, 8 Aug 2016 10:44:22 -0400)
xfstests        linux-v3.8-1149-g4e58a5b (Mon, 8 Aug 2016 10:50:34 -0400)
```

Building GCE images for xfstests

- Why use release automation? Important for many reasons
- Leveraging the Debian VM image
- Creating the payload
- Building the GCE image
  - Like everything else, controlled by an automated script: “gce-xfstests create-image”
    - Uploads the xfstests.tar.gz file, some control scripts, and a custom startup script into Google Cloud Storage bucket
    - Create a build VM using the debian-8 image with a custom startup script
    - The custom startup script installs packages, downloads and unpacks xfstests.tar.gz and files.tar.gz, and edits some system config files. Then it shuts down the build VM.
    - The GCE image is then created from the root disk of the build VM using the Cloud SDK: “gcloud compute images create xfstests-201608132226 --source-disk xfstests-bld”

Building GCE images for xfstests

- Why use release automation? Important for many reasons
- Leveraging the Debian VM image
- Creating the payload
- Building the GCE image
- Releasing the GCE image
  - After testing the image: “gce-xfstests smoke”
  - Export the image to a GCS bucket: “gce-xfstests export-image”
  - This creates a tar.gz file which can be copied into a different GCE project and then instantiated as an image for that project
  - This is how I publish GCE images using the **xfstests-cloud** GCE project
    - Access controlled via the **gce-xfstests** Google Group because IAM currently doesn’t allow a world readable GCE project (considered too dangerous)

Using gce-xfstests as a (kernel / fs) developer

- Set up a config file in ~/.config/gce-xfstests
  - Shell script variables: GS_BUCKET, GCE_PROJECT, GCE_ZONE, GCE_KERNEL
  - Optionally: GCE_SG_API and GS_REPORT_EMAIL
- Build a x86_64 kernel using the sample kernel configs in xfstests-bld/kernel-configs
- Run “gce-xfstests smoke” or “gce-xfstests full”
  - The full test takes about 7-8 hours, and costs less than $1.50 at full retail prices
  - The smoke test takes about 10 minutes, and costs pennies
- The results will be e-mailed to the developer when the test is complete
- To get more information for a particular test run:
  - gce-xfstests get-results [--unpack] <datecode>
- To run manual tests: “gce-xfstests shell”

Sample test report

```
CMDLINE: full
FSTESTIMG: gce-xfstests/xfstests-201608132226
FSTESTVER: e2fsprogs    v1.43.1-22-g25c4a20 (Wed, 8 Jun 2016 18:11:27 -0400)
FSTESTVER: fio          fio-2.6-8-ge6989e1 (Thu, 4 Feb 2016 12:09:48 -0700)
FSTESTVER: quota                81aca5c (Tue, 12 Jul 2016 16:15:45 +0200)
FSTESTVER: xfsprogs     v4.5.0 (Tue, 15 Mar 2016 15:25:56 +1100)
FSTESTVER: xfstests-bld 75f1eb0 (Sat, 13 Aug 2016 22:18:57 -0400)
FSTESTVER: xfstests     linux-v3.8-1149-g4e58a5b (Mon, 8 Aug 2016 10:50:34 -0400)
FSTESTVER: kernel       3.18.36-00001-g1c216bb #15 SMP Tue Jul 5 15:57:29 EDT 2016 x86_64
FSTESTCFG: "all"
FSTESTSET: "-g auto"
FSTESTEXC: ""
FSTESTOPT: "aex"
MNTOPTS: ""
CPUS: "2"
MEM: "7496.82"
MEM: 7680 MB (Max capacity)
BEGIN TEST 4k: Ext4 4k block Sat Aug 13 23:53:02 EDT 2016
Passed all 226 tests
BEGIN TEST 1k: Ext4 1k block Sun Aug 14 00:43:40 EDT 2016
Failures: generic/018 generic/273
...
```

Debugging / examining a currently running test VM

- List currently running test VMs: “gce-xfstests ls [-l]”
- Sshing into a test VM: “gce-xfstests ssh <instance-name>”
- Get the serial console output: “gce-xfstests console <instance-name>”
- Get the GCE VM metadata: “gcloud compute instances describe <instance>”
  - n.b. This is why we have “gce-xfstests ls -l”
- Query the test VM using a web browser
  - In addition to the top-level page, also supports varz, statusz, healthz requests

GCE xfstests

xfstests-201509141356

Status

Date/Time: Mon, 14 Sep 2015 14:06:48 EDT
Status: 14:06 4k generic/074
gce-xfstests command line: full

Test Log

```
Enabling auto exclude
e2fsck 1.43-WIP (18-May-2015)
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
/dev/mapper/xt-vdb: 11/327680 files (0.0% non-contiguous), 58462/1310720 blocks
FSTESTVER: e2fsprogs    v1.43-WIP-2015-05-18-64-g2334bd3 (Sat, 5 Sep 2015 22:21:35 -0400)
FSTESTVER: fio          fio-2.1.3 (Tue, 24 Sep 2013 08:42:24 -0600)
FSTESTVER: quota                022aabf (Wed, 26 Nov 2014 10:22:08 +0100)
FSTESTVER: xfsprogs     v4.2.0 (Mon, 7 Sep 2015 10:14:31 +1000)
FSTESTVER: xfstests-bld cd35820-dirty (Wed, 9 Sep 2015 19:38:57 -0400)
FSTESTVER: xfstests     linux-v3.8-723-g8c3016b-dirty (Tue, 4 Aug 2015 16:34:42 +1000)
FSTESTVER: kernel       4.2.0-ext4-10829-gd44a809-dirty #57 SMP Mon Sep 14 11:47:40 EDT 2015 x86_64
FSTESTCFG: "4k 1k ext3 nojournal ext3conv dioread_nolock data_journal inline bigalloc bigalloc_1k"
FSTESTSET: "-g auto"
FSTESTEXC: ""
FSTESTOPT: "aex"
MNTOPTS: ""
CPUS: "2"
MEM: "7477.53"
MEM: 7680 MB (Max capacity)
GCE ID: "12071580802646383625"
DATECODE: 201509141356
             total       used       free     shared    buffers     cached
Mem:          7477        428       7049          9         14         79
-/+ buffers/cache:        334       7143
Swap:            0          0          0
BEGIN TEST 4k: Ext4 4k block Mon Sep 14 13:57:41 EDT 2015
Device: /dev/mapper/xt-vdb
mk2fs options: -q
mount options: -o block_validity
FSTYP         -- ext4
PLATFORM      -- Linux/x86_64 xfstests-201509141356 4.2.0-ext4-10829-gd44a809-dirty
MKFS_OPTIONS  -- -q /dev/mapper/xt-vdc
MOUNT_OPTIONS -- -o acl,user_xattr -o block_validity /dev/mapper/xt-vdc /xt-vdc

ext4/001 1s ... [13:57:44] [13:57:44] 0s
ext4/002 0s ... [13:57:45] [13:57:45] 0s
ext4/003 3s ... [13:57:45] [13:57:48] 3s
ext4/004 3s ... [13:57:48] [13:57:51] 3s
ext4/005                [13:57:51] [13:57:52] - no qualified output
ext4/271 0s ... [13:57:52] [13:57:53] 1s
ext4/305 181s ...       [13:57:53] [14:00:54] 181s
ext4/306 4s ... [14:00:54] [14:00:58] 4s
```

Links

Support for other file systems

- Originally gce-xfstests and kvm-xfstests was ext4-specific
- All of the ext4-specific assumptions have been removed
  - The -c option can now take a file system type:
    - gce-xfstests -c ext4/all -g auto
    - gce-xfstests -c btrfs/4k,ext4/1k -g auto
  - PRIMARY_FSTYPE configuration variable specifies the default file system type
- Proof of concept support added for: btrfs, xfs, ext2, tmpfs
  - What is missing is the file system configurations that should be used for testing the fs
    - Mount options and mkfs options
  - If you are a file system developer and you would like to try using kvm-xfstests or gce-xfstests --- let's talk

To Learn More

- Sources: git://git.kernel.org/pub/scm/fs/ext2/xfstests-bld.git
  - https://github.com/tytso/xfstests-bld/blob/master/Documentation/gce-xfstests.md
  - “gce-xfstests help”
- Information about kvm-xfstests
  - https://github.com/tytso/xfstests-bld/blob/master/Documentation/kvm-quickstart.md
  - https://github.com/tytso/xfstests-bld/blob/master/Documentation/kvm-xfstests.md
- Interested in contributing to this project? Some potential ideas:
  - Control scripts for automated testing on Android
  - App-engine support for gce-xfstests
    - Browsing currently running tests
      - Watchdog to auto-recover if a kernel under test wedges
    - Browsing completed test results
      - Flaky test detection
    - Sharding fs configs to be tested on separate VM’s

Questions?