Using Container-specific Sysnames

Andrew Deason

June 2019

OpenAFS Workshop 2019

The Problem

- Say `/afs/cell/bin/gcc` → `/afs/cell/@sys/bin/gcc`
- RHEL6 running docker RHEL7, SLES12
- `--volume /afs:/afs`
- Containers get `amd64_rh6`, not `amd64_sles12`

Solutions

- Run separate clients
  - FUSE?
  - Needs work
  - Duplicate caching
- Separate @sys overlay
- Separate @sys lists per container

Multiple Sysname Lists

1. Get lookup request for `foo.@sys`

2. Pick sysname list for current pid

3. Do normal lookup using that list

What is a container?

- No “container” object in the Linux kernel
- mount namespace, pid namespace, etc
- We use the root object (dentry, vfsmount)
- Actually, a per-chroot sysname list
  - Adaptable to other platforms (zones)

Usage

```
$ fs sysname amd64_rh7 -pid 1234

$ fs sysname -global

$ fs sysname -pid 1234 -delete

$ fs sysname -debug-pid-sysnames

$ pid=$(docker inspect $container_id \
        | jq -r .[0].State.Pid)
$ fs sysname amd64_rh7 -pid $pid
```

Setting sysnames still requires root (`CAP_SYS_ADMIN`)

Examples

```
$ fs sysname new_sysname -pid 1234
fs: new sysname list set for pid 1234.

$ fs sysname -pid 1234
Current sysname for pid 1234 is 'new_sysname'

$ fs sysname -pid 1234 -delete
fs: sysname list deleted for pid 1234.

$ fs sysname new_sysname -global
fs: new global sysname set.

$ fs sysname -global
Current global sysname is 'new_sysname'
$
```

Examples

```
$ fs sysname -debug-pid-sysnames
   orig pid: 13560, last pid: 13662, key: { dentry: ffff96b05cda2e40,
mnt: ffff96b048d59860 }
     sysnames list: 'i386_linux26'
   orig pid: 13627, last pid: 13707, key: { dentry: ffff96b05858f6c0,
mnt: ffff96b046ddf3a0 }
     sysnames list: 'amd64_linux26'
$
```

Future

- Testing at a couple of sites
- OpenAFS release
- Port to Solaris?
- Docker plugin?

Code

Top Commit

`https://gerrit.openafs.org/13439`

All Commits

`https://gerrit.openafs.org/#/q/topic:chroot-sysname`

Slides

`http://dson.org/talks`

?