CRIU mounts migration: problems and solutions

Pavel Tikhomirov
Software developer at Virtuozzo

Aug 2020
Problems we face with mounts in CRIU and solutions

- Mount propagation struggle
- Overmounting “/” and chroot problem
- Opening overmounted files
- Procsfs from nested pidns
- Mounts lack info about tagged namespaces
Problems we face with mounts in CRIU
Mount propagation struggle #1
Mount propagation struggle
Mount propagation struggle

- We need to restore shared group topology
- shared_id + master_id
- ids can be only inherited
Mount propagation struggle

- This small example can create from 3 to >7 mounts depending on parent sharing:

```bash
mkdir a b
mount --bind a b
mount --bind a b
mount --bind a b
mount --bind a b
```
Mount propagation struggle: mount reparenting

```bash
cat /proc/self/mountinfo | grep testdir
# 644 639 0:59 / /tmp/testdir rw,relatime shared:19 - tmpfs tmpfs rw
# 645 644 0:59 /a /tmp/testdir/b rw,relatime shared:19 - tmpfs tmpfs rw
# 646 645 0:59 /a /tmp/testdir/b rw,relatime shared:19 - tmpfs tmpfs rw
# 647 644 0:59 /a /tmp/testdir/a rw,relatime shared:19 - tmpfs tmpfs rw

mount --bind a b
cat /proc/self/mountinfo | grep testdir
# 644 639 0:59 / /tmp/testdir rw,relatime shared:19 - tmpfs tmpfs rw
# 645 644 0:59 /a /tmp/testdir/b rw,relatime shared:19 - tmpfs tmpfs rw
# 646 645 0:59 /a /tmp/testdir/b rw,relatime shared:19 - tmpfs tmpfs rw
# 647 644 0:59 /a /tmp/testdir/a rw,relatime shared:19 - tmpfs tmpfs rw
# 648 646 0:59 /a /tmp/testdir/b rw,relatime shared:19 - tmpfs tmpfs rw
# 750 644 0:59 /a /tmp/testdir/a rw,relatime shared:19 - tmpfs tmpfs rw
# 749 645 0:59 /a /tmp/testdir/b rw,relatime shared:19 - tmpfs tmpfs rw
# 748 647 0:59 /a /tmp/testdir/a rw,relatime shared:19 - tmpfs tmpfs rw
```

Mount propagation struggle: mount reparenting
Mount propagation struggle: “Mount trap”

- you can’t access mount just after mounting it

```bash
# Prepare shared parent
mkdir /tmp/testdir
mount -t tmpfs tmpfs /tmp/testdir
mount --make-private /tmp/testdir
mount --make-shared /tmp/testdir
cd /tmp/testdir

# Actual commands
mkdir a/b -p
mount --bind a a/b
mount -t tmpfs tmpfs a/b
umount a/b

# umount: /tmp/testdir/a/b: no mount point specified.
```

Mount propagation struggle: "Mount trap"
Mount propagation struggle: “Non-uniform” propagation

- Mount order - shared mount created after propagation
- Umount one of propagation mounts when others are locked with children mounts
Mount propagation struggle: and some more

- “Cross-namespace” sharing groups
  - Order shared group creation and namespace creation
- We don’t know history we see only final snapshot
- Mounts from propagation group can have complex sharing, see [4]

- All those problems/inconveniences make restoring sharing in mount tree almost impossible…
Mount propagation struggle: Solution

- Assume kernel supports mount flag MS_SET_GROUP to copy sharing [1]
- Assume all mounts mounted right but **without** sharing options
- Assume we have an opened fd on each mount’s root dentry

Preparation:
- Group all mounts with same (shared_id, master_id) pair - sharing groups
- Put sharing groups in a trees where child->master_id == parent->shared_id
  - If two groups have same master_id mark them siblings (even if no parent)
  - If group has master_id but no parent - “external slavery”
Mount propagation struggle: Solution

Actual setup:

- Walk sharing group trees (parents before children)
  - Setup first (any) mount in a group
    - Is slave
      - Find any mount from parent sg or find external mount source
      - Copy sharing from it with MS_SET_GROUP
      - Make slave
    - Is shared - make it also shared
  - Setup other mounts - copy sharing from the first one
Problems we face with mounts in CRIU
Overmounting “/” and chroot problem #2
Overmounting “/” and chroot problem

• Container user overmounts “/”, e.g.:
  ```bash
  mount -t tmpfs tmpfs /
  ```

• There is currently no way to list files in this mount (without living ns)
  ○ Solution: O_MNT might help [5]

• Setns to such a mount namespace becomes chrooted
  ○ CRIU can chroot to proper root though not always
  ○ Solution: Maybe setns should not be chrooted?

• /proc/self/mountinfo is mangled for it

• Multiple overmount “/”

• Detached namespace with nsfs bindmount below chroot can become inaccessible
Problems we face with mounts in CRIU
Opening overmounted files #3
Opening overmounted files

- Fds should be reopened on the same mount + path which can be overmounted
- Also need to configure sharings with MS_SET_GROUP for overmounted mounts
Opening overmounted files: Solution

- All mounts of all mount namespaces are pre-mounted “plain” in service mntns
- For each mount namespace
  - Unshare new mntns copying all plain mounts
  - For each mount (tree order) in mntns
    - Open mp_fd to mountpoint before moving
    - Move mount to final mount tree (MS_MOVE)
    - Open mnt_fd to mountpoint after moving
      - Not always possible due to #2
    - Pivot_root to remove all extra mounts
- This way via mp_fd and mnt_fd all files can be openat
Problems we face with mounts in CRIU

Procfs from nested pidns #4
Procfs from nested pidns

• Procfs-es are per pid namespace
• Use /proc/1/ns/pid to detect
• Non root proc bindmount can be undetectable
  ○ Maybe need some ioctl?

• procfs: tasks -> pidns -> mount
• COW mappings: mounts -> tasks
Procfs from nested pidns: Solution

• Extra preparations:
  - Remove all procfs (with nested pidns) and their ancestors from the tree
  - Add “internal yard” tmpfs mount to the tree
  - Add host proc mount to internal yard
  - For each removed non-proc add helper mount

• Actual mounting: new stage after forking all processes
  - For each mntns: mount required procfs (entering proper pidns)
  - Reconstruct ancestors by bind-mounting from helpers
  - Use mnt_fd and mp_fd to resolve all paths and open them for new mounts
Problems we face with mounts in CRIU
Mounts lack info about tagged namespaces
#5
Mounts lack info about tagged namespaces

- Proc's `/proc/<pid>/ns/pid` is not always reliable
- Sysfs, nfs, mqueue mounts are tagged with other namespaces (e.g. sysfs - net)
- Solution: Maybe we need some general ioctl for it? [6]
Virtuozzo

Links:

1. Patch "mnt: allow to add a mount into an existing group":
   https://lkml.org/lkml/2017/1/23/712
2. Mounts v2 implementation:
   https://src.openvz.org/projects/OVZ/repos/criu/browse/criu/mount-v2.c?at=refs%2Fheads%2Fvz7-u15
3. Check-mounts:
   https://src.openvz.org/projects/OVZ/repos/criu/commits/5770d09b34c2
4. Complex sharing options test:
5. O_MNT https://patchwork.kernel.org/patch/11243249/
6. Sysfs ioctl to get netns tag https://patchwork.criu.org/patch/13168/
7. Mounts-v2 full algorithm https://criu.org/Mounts-v2
Additional materials…
A few words about Virtuozzo

- Working on container virtualization since 1999
- Here are our current logos together with some logos we were a part of previously
- Virtuozzo containers are System containers
- CRIU is our core technology for container migration
Why mounts c/r is at all important?

• We can’t let mounts disappear under running process
• System container can have “everything” inside
  ○ Complex mounts created by user
  ○ Mount namespaces per-service from systemd
  ○ Docker app containers
  ○ All this mixes up due to propagation
Mounts-v2 intro [2]

- --check-mounts[3] detects many problems with mount c/r on our tests
- Previously I had several attempts to improve old mounts algorithm (sharing)
  - Helped in small cases
  - New heuristics added - hard to understand
  - Not merged in mainstream CRIU
- Mounts-v2 started from Andrew’s out of tree patch (MS_SET_GROUP [1])
And some more…

• … problems:
  ○ MNT_LOCKED should not (dis)appear (users)
  ○ No fs-root mount
  ○ MS_MOVE fails with deleted mount and shared parent
  ○ Distinguishing file/dir bind-mounts (deleted)
  ○ External slavery == unexpected external mounts
  ○ Detached mounts
Virtuozzo

Links:

1. Patch "mnt: allow to add a mount into an existing group":
   https://lkml.org/lkml/2017/1/23/712
2. Mounts v2 implementation:
   https://src.openvz.org/projects/OVZ/repos/criu/browse/criu/mount-v2.c?at=refs%2Fheads%2Fvz7-u15
3. Check-mounts:
   https://src.openvz.org/projects/OVZ/repos/criu/commits/5770d09b34c2
4. Complex sharing options test:
5. O_MNT https://patchwork.kernel.org/patch/11243249/
6. Sysfs ioctl to get netns tag https://patchwork.criu.org/patch/13168/
7. Mounts-v2 full algorithm https://criu.org/Mounts-v2