OpenZFS Issue #15526 Patch and Mitigation for Older Versions

> SUBSTANTIAL REVISION: Issue #15526 has been patched in ZFS versions 2.2.2 and 2.1.14, according to open source reporting. The mitigation below only applies to older versions of ZFS.

Summary

A silent data corruption bug exists in ZFS versions 2.1.4 up to 2.1.13, 2.2.0 and 2.2.1, per this GitHub issue. Versions 2.2.2 and 2.1.14 have patched the issue, according to open source reporting .

> A ZFS scrub will not identify any data corrupted by this bug. The only high-assurance method to check if any files were corrupted is to compare files within ZFS to their copies stored outside of ZFS.

If you are still using an older version of ZFS that is impacted by this issue, you may significantly lower the chance that this issue impacts your file system by setting the ZFS parameter zfs_dmu_offset_next_sync to 0. Note that this does not prevent the issue from occurring, per this GitHub comment, but it does lower the chance that silent data corruption occurs.

Details

Mitigation

Linux

Runtime

To apply the mitigation in runtime, run the following command as the root user:

```
echo 0 > /sys/module/zfs/parameters/zfs_dmu_offset_next_sync
```

Permanent

To apply the mitigation permanently, create a file in `/etc/modprobe.d/` such as:

```
/etc/modprobe.d/mitigation.conf
```

Containing the following:

```
options zfs zfs_dmu_offset_next_sync=0
```

FreeBSD

Runtime

To apply the mitigation in runtime, run the following command as the root user:

```
sysctl -w vfs.zfs.dmu_offset_next_sync=0
```

Permanent

To apply the mitigation permanently, append the following line to `/etc/sysctl.conf`:

```
vfs.zfs.dmu_offset_next_sync=0
```

Reproducing the Bug

Linux

To reproduce the bug in Linux, use the script below (copy pasted from the following gist):

```
#!/bin/bash
#
# Run this script multiple times in parallel inside your pool's mount
```

```
# to reproduce https://github.com/openzfs/zfs/issues/15526.  Like:
#
# ./reproducer.sh & ./reproducer.sh & ./reproducer.sh & ./reproducer.sh & wait
#

#if [ $(cat /sys/module/zfs/parameters/zfs_bclone_enabled) != "1" ] ; then
#	echo "please set /sys/module/zfs/parameters/zfs_bclone_enabled = 1"
#	exit
#fi

prefix="reproducer_${BASHPID}_"
dd if=/dev/urandom of=${prefix}0 bs=1M count=1 status=none

echo "writing files"
end=1000
h=0
for i in `seq 1 2 $end` ; do
	let "j=$i+1"
	cp  ${prefix}$h ${prefix}$i
	cp --reflink=never ${prefix}$i ${prefix}$j
	let "h++"
done

echo "checking files"
for i in `seq 1 $end` ; do
	diff ${prefix}0 ${prefix}$i
done
```

FreeBSD

To reproduce the bug in FreeBSD, use the script below (copy pasted from the following post):

```
#!/bin/bash
#
# Run this script multiple times in parallel inside your pool's mount
# to reproduce https://github.com/openzfs/zfs/issues/15526.  Like:
#
# ./reproducer.sh & ./reproducer.sh & ./reproducer.sh & ./reproducer.sh & wait
#
```

```
#if [ $(cat /sys/module/zfs/parameters/zfs_bclone_enabled) != "1" ] ; then
#       echo "please set /sys/module/zfs/parameters/zfs_bclone_enabled = 1"
#       exit
#fi

prefix="reproducer_${BASHPID}_"
dd if=/dev/urandom of=${prefix}0 bs=1M count=1 status=none

echo "writing files"
end=1000
h=0
for i in `seq 1 2 $end` ; do
        let "j=$i+1"
        cp  ${prefix}$h ${prefix}$i
        cp  ${prefix}$i ${prefix}$j
        let "h++"
done

echo "checking files"
for i in `seq 1 $end` ; do
        diff ${prefix}0 ${prefix}$i
done
```

Commentary

I was unable to reproduce this issue in TrueNAS Core 13.0-U5.3 (FreeBSD) but I was able to reproduce it in Proxmox 8.0.4 (Debian).

Source Description Block

> Multiple sources:
>
> Issue tracking in OpenZFS: https://github.com/openzfs/zfs/issues/15526
>
> Mitigation: https://github.com/openzfs/zfs/issues/15526#issuecomment-1823737998
> Linux reproducer script:
>
> https://gist.github.com/tonyhutter/d69f305508ae3b7ff6e9263b22031a84
>
> FreeBSD reproducer script: https://www.truenas.com/community/threads/truenas-13-0-u6-is-now-available.114337/page-3

> TrueNAS Core (FreeBSD) issue forum thread: https://www.truenas.com/community/threads/silent-corruption-with-openzfs-ongoing-discussion-and-testing.114390/
> Documentation on dmu_offset_next_sync: https://openzfs.github.io/openzfs-docs/Performance%20and%20Tuning/Module%20Parameters.html#zfs-dmu-offset-next-sync
> Data corruption bug occurs even with zfs_dmu_offset_next_sync set to 0: https://github.com/openzfs/zfs/issues/15526#issuecomment-1826348986
> Reddit thread on the bug: https://old.reddit.com/r/DataHoarder/comments/1821mpr/heads_up_for_a_data_corruption_bug_in_zfs_few/
> Reddit thread on the bug: https://old.reddit.com/r/zfs/comments/1826lgs/psa_its_not_block_cloning_its_a_data_corruption/
> Issue fixed in versions 2.2.2 and 2.1.14: https://www.phoronix.com/news/OpenZFS-2.2.2-Released

Licensing

This page **(not including the code snippets)** is licensed under a Creative Commons Universal (CC0 1.0) Public Domain Dedication. For code snippet licensing, please contact the original authors.

Revision #13
Created 2023-11-23 21:17:52 UTC by Henry Reed
Updated 2024-09-08 06:00:08 UTC by Henry Reed