

Упражнение 1. Администрирование ZFS. Транзакции в ZFS

1. Создание пула

```
# zpool create tiger mirror c0t2d0s0 c0t2d0s1 mirror c0t2d0s3  
c0t2d0s4
```

Проверьте с помощью `zpool status` состояние тома, `zpool list` его размер

Дерево устройств:

2. Создание ФС и тома

```
# zfs create tiger/data1  
# zfs create -V 200m tiger/volume1
```

Выполните `zfs get all tiger/data1`. Какой параметр нужен, чтобы изменить точку монтирования?

Параметр _____

3. Установка свойств

```
# zfs set sharenfs="rw=@192.168.0.0/23" tiger/data1  
# zfs set copies=2 tiger/volume1
```

4. Мониторинг транзакций

```
# dtrace -qn '::spa_sync:entry { printf("time=%Y, txg=%d  
[%s]\n", walltimestamp, args[1], args[0]->spa_name); }'
```

Что означает параметр `args[1]`?

5. Небольшие нагрузки

```
# while : ; do mkfile 128k file-small; sleep 1; date; done
```

Какова периодичность записи транзакций на диск?

6. Большая нагрузка на диск

```
# mkfile 100m file-big
```

Изменилась ли периодичность записи на диск (коммита транзакций)?

Упражнение 2. Конвейер ZIO

1. Создайте тестовую файловую систему testzio:

```
# zfs create tiger/testzio
```

2. Создайте тестовый файл, запустите prstat на пул, а также скрипт ziostat:

```
# dd if=/dev/urandom of=/root/file-big bs=8192 count=8192
# prstat -p `pgrep zpool-tiger`
# /root/ziostat.d
```

2. В другом терминале симулируйте I/O и дождитесь коммита последней транзакции:

```
# time cp /root/file-big /tiger/testzio/ && dtrace -f spa_sync
```

По завершении копирования остановите выполнение ziostat.

Каково среднее время выполнения соответствующих операций?

zio_vdev_io_start - _____

zio_vdev_io_done - _____

zio_checksum_generate - _____

zio_write_bp_init - _____

5. Перезапустите ziostat, запустите несколько потоков копирования.

```
# for I in 1 2 3 4
> do
> cp /root/file-big /tiger/testzio/ &
> done && dtrace -f spa_sync
```

Изменилось ли время исполнения операций zio_vdev_io_start и zio_vdev_io_done?

6. Измените алгоритм компрессии и алгоритм подсчета контрольной суммы на ФС:

```
# zfs set compression=gzip-9 tiger/testzio
# zfs set checksum=sha256 tiger/testzio
```

6. Перезапустите ziostat и запустите процесс копирования

```
# time cp /root/file-big /tiger/testzio/ && dtrace -f spa_sync
```

У каких операций изменилось время выполнения и насколько?

Упражнение 3. Подсистема DMU

1. Создайте много небольших файлов на ФС (для конструкции {N..M} требуется интерпретатор bash)

```
# mkfile 128k /tiger/small-{1..128}
# zdb -bb tiger
```

Каково соотношение метаданные/данные?

2. Создайте крупные файлы

```
# rm /tiger/small* ; mkfile 16m /tiger/big-{1..12}
# zdb -bb tiger
```

Каково соотношение метаданные/данные? Изменилось ли оно?

3. Как располагаются метаданные на ФС?

```
# zdb -bb -vvvv tiger
```

Каков алгоритм компрессии метаданных? Сколько копий у блоков типов L0 DMU objset, L0 ZFS plain file?

4. Создайте файл и затем удалите его. Запишите номера транзакций до и после его удаления (необходимо дождаться коммита транзакции)

```
# echo "Hello, world" > /tiger/my-test-file
# zdb -u tiger
# rm /tiger/my-test-file
# zdb -u tiger
```

TXG до удаления _____

TXG после удаления _____

5. Используйте ZFS rewind, чтобы получить старое состояние ФС:

```
# zdb -dddd -vvvv -t <Старая_TXG> tiger | less
```

Найдите файл my-test-file и запишите его DVA _____

6. По заданному DVA восстановите файл

```
# zdb -R tiger <DVA>:r > /root/my-test-file
```

Упражнение 4. ARC и Интерфейсный уровень

1. В окне терминала запустите скрипт `arc_buf_hash.d`

```
# ./arc_buf_hash.d
```

2. Скопируйте файл `words`, скомпилируйте и в другом окне терминала запустите программу `read_mru_mfu`, нажмите дважды Enter:

```
# cp /usr/dict/words /tiger
# ./read_mru_mfu
<CR>
<CR>
```

Запишите полученный хеш _____

3. Рассчитайте смещение `arc_buf_hdr_t` в таблице `buf_hash_table` исходя из значения `ht_mask` и размера указателя 8 (4 для 32-битных платформ):

```
# mdb -k
# buf_hash_table::print
> (<hash>&<ht_mask>)*8=K
```

Смещение: _____

Используя значение `ht_table`, получите дамп заголовка буфера ARC

```
> <ht_table>+<смещение>::print void* | ::print arc_buf_hdr_t
```

В каком из кешей ARC располагается запись, что в ней хранится (данные или метаданные?)

Получите дамп буфера данных `arc_buf_t` используя поле `b_buf` заголовка:

```
> <b_buf>::print arc_buf_t b_data | ::dump
```

4. (Дополнительно) Определите, к какому объекту имеет отношение указанная запись в кеше. Для этого используйте поле `b_private` структуры `arc_buf_t`

```
> <b_buf>::print arc_buf_t b_private | ::print dmu_buf_impl_t
db_dnode | ::print struct dnode dn_bonus | ::print dmu_buf_impl_t
db_user_ptr | ::print znode_t z_vnode | ::print vnode_t
```