

**su o**

Методическое пособие

по выполнению домашнего задания курса

**«Администратор Linux. Professional»**

**Стенд с Vagrant c ZFS**

**Содержание**

| 1. Введение | 3 |
| --- | --- |
| 1. Цели домашнего задания | 4 |
| 1. Описание домашнего задания | 5 |
| 1. Пошаговая инструкция   выполнения домашнего задания | 6 |
| 1. Критерий оценивания | 17 |
| 1. Рекомендуемые источники | 18 |

# 1. Введение

ZFS(Zettabyte File System) — файловая система, созданная компанией Sun Microsystems в 2004-2005 годах для ОС Solaris. Эта файловая система поддерживает большие объёмы данных, объединяет в себе концепции файловой системы, RAID-массивов, менеджера логических дисков и принципы легковесных файловых систем.

ZFS продолжает активно развиваться. К примеру проект FreeNAS использует возможности ZFS для реализации ОС для управления SAN/NAS хранилищ.

Из-за лицензионных ограничений, поддержка ZFS в GNU/Linux ограничена. По умолчанию ZFS отсутствует в ядре linux.

Основное преимущество ZFS — это её полный контроль над физическими носителями, логическими томами и постоянное поддержание консистенции файловой системы. Оперируя на разных уровнях абстракции данных, ZFS способна обеспечить высокую скорость доступа к ним, контроль их целостности, а также минимизацию фрагментации данных. ZFS гибко настраивается, позволяет в процессе работы изменять объём дискового пространства и задавать разный размер блоков данных для разных применений, обеспечивает параллельность выполнения операций чтения-записи.

# 2. Цели домашнего задания

Научится самостоятельно устанавливать ZFS, настраивать пулы, изучить основные возможности ZFS.

# 3. Описание домашнего задания

1. Определить алгоритм с наилучшим сжатием:

* Определить какие алгоритмы сжатия поддерживает zfs (gzip, zle, lzjb, lz4);
* создать 4 файловых системы на каждой применить свой алгоритм сжатия;
* для сжатия использовать либо текстовый файл, либо группу файлов.

1. Определить настройки пула.

С помощью команды zfs import собрать pool ZFS.

Командами zfs определить настройки:

- размер хранилища;

- тип pool;

- значение recordsize;

- какое сжатие используется;

- какая контрольная сумма используется.

1. Работа со снапшотами:

* скопировать файл из удаленной директории;
* восстановить файл локально. zfs receive;
* найти зашифрованное сообщение в файле secret\_message.

# 4. Пошаговая инструкция выполнения домашнего задания

ПК на Unix c 8ГБ ОЗУ или виртуальная машина с включенной Nested Virtualization.

Предварительно установленное и настроенное следующее ПО:

Hashicorp Vagrant (<https://www.vagrantup.com/downloads>)

Oracle VirtualBox (<https://www.virtualbox.org/wiki/Linux_Downloads>).

Все дальнейшие действия были проверены при использовании Vagrant 2.4.0, VirtualBox v7.0.12 и образа CentOS 7 2004.01 из Vagrant cloud. Серьёзные отступления от этой конфигурации могут потребовать адаптации с вашей стороны.

## Создаем виртуальную машину

Создаём каталог, в котором будут храниться настройки виртуальной машины и дополнительные диски. В каталоге создаём файл с именем Vagrantfile, добавляем в него следующее содержимое:

# -\*- mode: ruby -\*-

# vim: set ft=ruby :

disk\_controller **=** 'IDE' # MacOS. This setting is OS dependent. Details https://github.com/hashicorp/vagrant/issues/8105

MACHINES **=** **{**

:zfs **=>** **{**

:box\_name **=>** "centos/7"**,**

:box\_version **=>** "2004.01"**,**

:disks **=>** **{**

:sata1 **=>** **{**

:dfile **=>** './sata1.vdi'**,**

:size **=>** 512**,**

:port **=>** 1

**},**

:sata2 **=>** **{**

:dfile **=>** './sata2.vdi'**,**

:size **=>** 512**,** # Megabytes

:port **=>** 2

**},**

:sata3 **=>** **{**

:dfile **=>** './sata3.vdi'**,**

:size **=>** 512**,**

:port **=>** 3

**},**

:sata4 **=>** **{**

:dfile **=>** './sata4.vdi'**,**

:size **=>** 512**,**

:port **=>** 4

**},**

:sata5 **=>** **{**

:dfile **=>** './sata5.vdi'**,**

:size **=>** 512**,**

:port **=>** 5

**},**

:sata6 **=>** **{**

:dfile **=>** './sata6.vdi'**,**

:size **=>** 512**,**

:port **=>** 6

**},**

:sata7 **=>** **{**

:dfile **=>** './sata7.vdi'**,**

:size **=>** 512**,**

:port **=>** 7

**},**

:sata8 **=>** **{**

:dfile **=>** './sata8.vdi'**,**

:size **=>** 512**,**

:port **=>** 8

**},**

**}**

**},**

**}**

Vagrant**.**configure**(**"2"**)** **do** **|**config**|**

MACHINES**.**each **do** **|**boxname**,** boxconfig**|**

config**.**vm**.**define boxname **do** **|**box**|**

box**.**vm**.**box **=** boxconfig**[**:box\_name**]**

box**.**vm**.**box\_version **=** boxconfig**[**:box\_version**]**

box**.**vm**.**host\_name **=** "zfs"

box**.**vm**.**provider :virtualbox **do** **|**vb**|**

vb**.**customize **[**"modifyvm"**,** :id**,** "--memory"**,** "1024"**]**

needsController **=** **false**

boxconfig**[**:disks**].**each **do** **|**dname**,** dconf**|**

**unless** File**.**exist?**(**dconf**[**:dfile**])**

vb**.**customize **[**'createhd'**,** '--filename'**,** dconf**[**:dfile**],** '--variant'**,** 'Fixed'**,** '--size'**,** dconf**[**:size**]]**

needsController **=** **true**

**end**

**end**

**if** needsController **==** **true**

vb**.**customize **[**"storagectl"**,** :id**,** "--name"**,** "SATA"**,** "--add"**,** "sata" **]**

boxconfig**[**:disks**].**each **do** **|**dname**,** dconf**|**

vb**.**customize **[**'storageattach'**,** :id**,** '--storagectl'**,** 'SATA'**,** '--port'**,** dconf**[**:port**],** '--device'**,** 0**,** '--type'**,** 'hdd'**,** '--medium'**,** dconf**[**:dfile**]]**

**end**

**end**

**end**

box**.**vm**.**provision "shell"**,** inline: **<<-**SHELL

#install zfs repo

yum install **-**y http:/**/**download**.**zfsonlinux**.**org**/**epel**/**zfs**-**release**.**el7\_8**.**noarch**.**rpm

#import gpg key

rpm **--**import **/**etc**/**pki**/**rpm**-**gpg**/**RPM**-**GPG**-**KEY**-**zfsonlinux

#install DKMS style packages for correct work ZFS

yum install **-**y epel**-**release kernel**-**devel zfs

#change ZFS repo

yum**-**config**-**manager **--**disable zfs

yum**-**config**-**manager **--**enable zfs**-**kmod

yum install **-**y zfs

#Add kernel module zfs

modprobe zfs

#install wget

yum install **-**y wget

SHELL

**end**

**end**

**end**

Результатом выполнения команды *vagrant up* станет созданная виртуальная машина, с 8 дисками и уже установленным и готовым к работе ZFS.

Заходим на сервер: *vagrant ssh*

Дальнейшие действия выполняются от пользователя root. Переходим в root пользователя: *sudo -i*

## Определение алгоритма с наилучшим сжатием

Смотрим список всех дисков, которые есть в виртуальной машине: *lsblk*

**[**root**@**zfs **~]**# lsblk

NAME MAJ**:**MIN RM SIZE RO TYPE MOUNTPOINT

sda 8**:**0 0 40G 0 disk

└─sda1 8**:**1 0 40G 0 part **/**

sdb 8**:**16 0 512M 0 disk

sdc 8**:**32 0 512M 0 disk

sdd 8**:**48 0 512M 0 disk

sde 8**:**64 0 512M 0 disk

sdf 8**:**80 0 512M 0 disk

sdg 8**:**96 0 512M 0 disk

sdh 8**:**112 0 512M 0 disk

sdi 8**:**128 0 512M 0 disk

Создаём пул из двух дисков в режиме RAID 1:

[root@zfs ~]# zpool create otus1 mirror /dev/sdb /dev/sdc

Создадим ещё 3 пула:

**[**root**@**zfs **~]**# zpool create otus2 mirror **/**dev**/**sdd **/**dev**/**sde

**[**root**@**zfs **~]**# zpool create otus3 mirror **/**dev**/**sdf **/**dev**/**sdg

**[**root**@**zfs **~]**# zpool create otus4 mirror **/**dev**/**sdh **/**dev**/**sdi

Смотрим информацию о пулах: *zpool list*

NAME SIZE ALLOC FREE CKPOINT EXPANDSZ FRAG CAP DEDUP HEALTH ALTROOT

otus1 480M 91**.**5K 480M **-** **-** 0**%** 0**%** 1**.**00x ONLINE **-**

otus2 480M 91**.**5K 480M **-** **-** 0**%** 0**%** 1**.**00x ONLINE **-**

otus3 480M 91**.**5K 480M **-** **-** 0**%** 0**%** 1**.**00x ONLINE **-**

otus4 480M 91**.**5K 480M **-** **-** 0**%** 0**%** 1**.**00x ONLINE **-**

Команда *zpool status* показывает информацию о каждом диске, состоянии сканирования и об ошибках чтения, записи и совпадения хэш-сумм. Команда *zpool list* показывает информацию о размере пула, количеству занятого и свободного места, дедупликации и т.д.

Добавим разные алгоритмы сжатия в каждую файловую систему:

* Алгоритм lzjb: *zfs set compression=lzjb otus1*
* Алгоритм lz4: *zfs set compression=lz4 otus2*
* Алгоритм gzip: *zfs set compression=gzip-9 otus3*
* Алгоритм zle: *zfs set compression=zle otus4*

Проверим, что все файловые системы имеют разные методы сжатия:

**[**root**@**zfs **~]**# zfs get all **|** **grep** compression

otus1 compression lzjb local

otus2 compression lz4 local

otus3 compression gzip-9 local

otus4 compression zle local

Сжатие файлов будет работать только с файлами, которые были добавлены после включение настройки сжатия.

Скачаем один и тот же текстовый файл во все пулы:

**[**root**@**zfs **~]**# for i in **{**1**..**4**};** **do** **wget** **-**P **/**otus**$i** https**://**gutenberg.org**/**cache**/**epub**/**2600**/**pg2600.converter.log**;** **done**

**...**

Проверим, что файл был скачан во все пулы:

**[**root**@**zfs **~]**# ls -l **/**otus**\***

**/**otus1**:**

total 22005

**-**rw-r--r--. 1 root root 40750827 Oct 2 08**:**07 pg2600.converter.log

**/**otus2**:**

total 17966

**-**rw-r--r--. 1 root root 40750827 Oct 2 08**:**07 pg2600.converter.log

**/otus3:**

**total 10945**

**-**rw-r--r--. 1 root root 40750827 Oct 2 08**:**07 pg2600.converter.log

**/**otus4**:**

total 39836

**-**rw-r--r--. 1 root root 40750827 Oct 2 08**:**07 pg2600.converter.log

Уже на этом этапе видно, что самый оптимальный метод сжатия у нас используется в пуле otus3.

Проверим, сколько места занимает один и тот же файл в разных пулах и проверим степень сжатия файлов:

**[**root**@**zfs **~]**# zfs list

NAME USED AVAIL REFER MOUNTPOINT

otus1 21**.**6M 330M 21**.**5M **/**otus1

otus2 17**.**7M 334M 17**.**6M **/**otus2

**otus3 10.8M 341M 10.7M /otus3**

otus4 39**.**0M 313M 38**.**9M **/**otus4

**[**root**@**zfs **~]**# zfs get all **|** **grep** compressratio **|** **grep** **-**v ref

otus1 compressratio 1**.**80x **-**

otus2 compressratio 2**.**21x **-**

**otus3 compressratio 3.63x -**

otus4 compressratio 1**.**00x **-**

Таким образом, у нас получается, что алгоритм **gzip-9** самый эффективный по сжатию.

## Определение настроек пула

Скачиваем архив в домашний каталог:

**[**root**@**zfs **~]**# wget -O archive.tar.gz --no-check-certificate 'https://drive.usercontent.google.com/download?id=1MvrcEp-WgAQe57aDEzxSRalPAwbNN1Bb&export=download'

Разархивируем его:

**[**root**@**zfs **~]**# tar **-**xzvf archive.tar.gz

zpoolexport**/**

zpoolexport**/**filea

zpoolexport**/**fileb

**[**root**@**zfs **~]**#

Проверим, возможно ли импортировать данный каталог в пул:

**[**root**@**zfs **~]**# zpool import -d zpoolexport**/**

pool**:** **otus**

id**:** 6554193320433390805

state**:** ONLINE

action**:** The pool can be imported using its name or numeric identifier.

config**:**

otus ONLINE

**mirror-0 ONLINE**

**/root/zpoolexport/filea ONLINE**

**/root/zpoolexport/fileb ONLINE**

**[**root**@**zfs **~]**#

Данный вывод показывает нам имя пула, тип raid и его состав.

Сделаем импорт данного пула к нам в ОС:

**[**root**@**zfs **~]**# zpool import -d zpoolexport**/** otus

**[**root**@**zfs **~]**# zpool status

pool**:** otus

state**:** ONLINE

scan**:** none requested

config**:**

NAME STATE READ WRITE CKSUM

otus ONLINE 0 0 0

mirror-0 ONLINE 0 0 0

**/**root**/**zpoolexport**/**filea ONLINE 0 0 0

**/**root**/**zpoolexport**/**fileb ONLINE 0 0 0

errors**:** No known data errors

Команда zpool status выдаст нам информацию о составе импортированного пула.

Если у Вас уже есть пул с именем otus, то можно поменять его имя во время импорта: *zpool import -d zpoolexport****/*** *otus newotus*

Далее нам нужно определить настройки: *zpool get all otus*

Запрос сразу всех параметром файловой системы: *zfs get all otus*

**[**root**@**zfs **~]**# zfs get all otus

NAME PROPERTY VALUE SOURCE

otus type filesystem **-**

otus creation Fri May 15 4**:**00 2020 **-**

otus used 2**.**04M **-**

otus available 350M **-**

otus referenced 24K **-**

otus compressratio 1**.**00x **-**

otus mounted yes **-**

otus quota none default

otus reservation none default

otus recordsize 128K local

otus mountpoint **/**otus default

otus sharenfs off default

otus checksum sha256 local

otus compression zle local

otus atime on default

otus devices on default

otus exec on default

otus setuid on default

otus readonly off default

otus zoned off default

otus snapdir hidden default

otus aclinherit restricted default

otus createtxg 1 **-**

otus canmount on default

otus xattr on default

otus copies 1 default

otus version 5 **-**

otus utf8only off **-**

otus normalization none **-**

otus casesensitivity sensitive **-**

otus vscan off default

otus nbmand off default

otus sharesmb off default

otus refquota none default

otus refreservation none default

otus guid 14592242904030363272 **-**

otus primarycache all default

otus secondarycache all default

otus usedbysnapshots 0B **-**

otus usedbydataset 24K **-**

otus usedbychildren 2**.**01M **-**

otus usedbyrefreservation 0B **-**

otus logbias latency default

otus objsetid 54 **-**

otus dedup off default

otus mlslabel none default

otus sync standard default

otus dnodesize legacy default

otus refcompressratio 1**.**00x **-**

otus written 24K **-**

otus logicalused 1020K **-**

otus logicalreferenced 12K **-**

otus volmode default default

otus filesystem\_limit none default

otus snapshot\_limit none default

otus filesystem\_count none default

otus snapshot\_count none default

otus snapdev hidden default

otus acltype off default

otus context none default

otus fscontext none default

otus defcontext none default

otus rootcontext none default

otus relatime off default

otus redundant\_metadata all default

otus overlay off default

otus encryption off default

otus keylocation none default

otus keyformat none default

otus pbkdf2iters 0 default

otus special\_small\_blocks 0 default

C помощью команды grep можно уточнить конкретный параметр, например:

Размер: *zfs get available otus*

**[**root**@**zfs **~]**# zfs get available otus

NAME PROPERTY VALUE SOURCE

otus available 350M **-**

Тип: *zfs get readonly otus*

**[**root**@**zfs **~]**# zfs get readonly otus

NAME PROPERTY VALUE SOURCE

otus readonly off default

По типу FS мы можем понять, что позволяет выполнять чтение и запись

Значение recordsize: *zfs get recordsize otus*

**[**root**@**zfs **~]**# zfs get recordsize otus

NAME PROPERTY VALUE SOURCE

otus recordsize 128K local

Тип сжатия (или параметр отключения): *zfs get compression otus*

**[**root**@**zfs **~]**# zfs get compression otus

NAME PROPERTY VALUE SOURCE

otus compression zle local

Тип контрольной суммы: *zfs get checksum otus*

**[**root**@**zfs **~]**# zfs get checksum otus

NAME PROPERTY VALUE SOURCE

otus checksum sha256 local

## Работа со снапшотом, поиск сообщения от преподавателя

Скачаем файл, указанный в задании:

**[**root**@**zfs **~]**# wget -O otus\_task2.file --no-check-certificate https://drive.usercontent.google.com/download?id=1wgxjih8YZ-cqLqaZVa0lA3h3Y029c3oI&export=download

Восстановим файловую систему из снапшота:

*zfs receive otus/test@today < otus\_task2.file*

Далее, ищем в каталоге /otus/test файл с именем “secret\_message”:

**[**root**@**zfs **~]**# find **/**otus**/**test **-**name "secret\_message"

**/**otus**/**test**/**task1**/**file\_mess**/**secret\_message

Смотрим содержимое найденного файла:

**[**root**@**zfs **~]**# cat **/**otus**/**test**/**task1**/**file\_mess**/**secret\_message

https://otus.ru/lessons/linux-hl/

Тут мы видим ссылку на курс OTUS, задание выполнено.

Ранее предложенный Vagrantfile можно использовать для выполнения домашней работы.

Для конфигурации сервера (установки и настройки ZFS) необходимо написать отдельный Bash-скрипт и добавить его в Vagrantfile. Пример добавления скрипта в Vagrantfile:

# -\*- mode: ruby -\*-

# vim: set ft=ruby :

disk\_controller **=** 'IDE' # MacOS. This setting is OS dependent. Details https://github.com/hashicorp/vagrant/issues/8105

MACHINES **=** **{**

:zfs **=>** **{**

:box\_name **=>** "centos/7"**,**

:box\_version **=>** "2004.01"**,**

:provision **=>** "test.sh"**,**

:disks **=>** **{**

…

**box.vm.provision "shell", path: boxconfig[:provision]**

**end**

**end**

**end**

Помимо Vagrantfile требуется предоставить список команд по каждому заданию и их выводы.

# 5. Критерий оценивания

Статус «Принято» ставится при выполнении следующих условий:

1. Ссылка на репозиторий GitHub.

2. Vagrantfile с Bash-скриптом, который будет конфигурировать сервер

3. Документация по каждому заданию.

Cоздайте файл README.md и снабдите его следующей информацией:

- название выполняемого задания;

- текст задания;

- описание команд и их вывод;

- особенности проектирования и реализации решения,

- заметки, если считаете, что имеет смысл их зафиксировать в

репозитории.

# 6. Рекомендуемые источники

Статья о ZFS

<https://ru.wikipedia.org/wiki/ZFS>

Статья «Что такое ZFS? И почему люди от неё без ума?» - <https://habr.com/ru/post/424651/>

Официальная документация по Oracle Solaris ZFS

<https://docs.oracle.com/cd/E19253-01/819-5461/gbcya/index.html>

Статья о ZFS (теория и практика)

<http://xgu.ru/wiki/ZFS>