Московский Государственный Технический Университет

имени Н. Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Компьютерные системы и сети»

УТВЕРЖДАЮ

Зав. кафедрой ИУ6

д.т.н., проф. \_\_\_\_\_\_\_\_\_\_\_\_ Сюзев В.В.

"\_\_\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2013 г.

Исследование методов организации внешней памяти.

Методические указания по выполнению лабораторной работы

по дисциплине "Операционные системы"

МОСКВА 2013

В настоящее время существует множество файловых систем различных категорий, например:

* Для носителей с [произвольным доступом](http://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B8%D0%B7%D0%B2%D0%BE%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%B4%D0%BE%D1%81%D1%82%D1%83%D0%BF) ([FAT32](http://ru.wikipedia.org/wiki/FAT32), [HPFS](http://ru.wikipedia.org/wiki/HPFS), [ext4](http://ru.wikipedia.org/wiki/Ext3), [NTFS](http://ru.wikipedia.org/wiki/NTFS));
* Для оптических носителей — [CD](http://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%B0%D0%BA%D1%82-%D0%B4%D0%B8%D1%81%D0%BA) и [DVD](http://ru.wikipedia.org/wiki/DVD): [ISO9660](http://ru.wikipedia.org/wiki/ISO9660), [HFS](http://ru.wikipedia.org/wiki/HFS_(%D1%84%D0%B0%D0%B9%D0%BB%D0%BE%D0%B2%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0)), [UDF](http://ru.wikipedia.org/wiki/Universal_Disk_Format);
* [Виртуальные файловые системы](http://ru.wikipedia.org/wiki/%D0%92%D0%B8%D1%80%D1%82%D1%83%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%84%D0%B0%D0%B9%D0%BB%D0%BE%D0%B2%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0): [AEFS](http://ru.wikipedia.org/wiki/AEFS) и др.;
* Сетевые файловые системы: [NFS](http://ru.wikipedia.org/wiki/Network_File_System), [CIFS](http://ru.wikipedia.org/wiki/CIFS), [SSHFS](http://ru.wikipedia.org/wiki/SSHFS), [GmailFS](http://richard.jones.name/google-hacks/gmail-filesystem/gmail-filesystem.html) и др.;
* Для флэш-памяти: [YAFFS](http://ru.wikipedia.org/wiki/YAFFS), [ExtremeFFS](http://ru.wikipedia.org/w/index.php?title=ExtremeFFS&action=edit&redlink=1), [exFAT](http://ru.wikipedia.org/wiki/ExFAT) и др.

**Цель работы:** исследование файловых систем, применяющихся в UNIX-подобных системах, а также изучение основных утилит для работы с файлами.

Для выполнения данной ЛР рекомендуется использовать учетную запись суперпользователя (root), либо использовать su. Задания, выделенные курсивом, необязательны для выполнения.

**Используемые термины**

Терминология UNIX-систем отличается от терминологии Windows. Ниже приведены термины, которые будут использованы в данной ЛР.

Индексный дескриптор (inode) – структура данных, хранящая метаинформацию. Каждый индексный дескриптор имеет уникальный идентификатор в файловой системе.

Жёсткая ссылка (hard link, hardlink) – составляющая каталога, ассоциирует имя файла с содержимым на диске. Прописывается в индексном дескрипторе файла.

Символьная ссылка (символическая ссылка, мягкая ссылка, symbolic link, symlink) – представляет собой структуру данных, которая содержит в себе только ссылку на другой файл. Символьные ссылки часто используются как псевдонимы. В Windows тоже существуют символьные ссылки. Доступ к символьной ссылке не ограничен, права доступа проверяются при попытке доступа к целевому файлу.

Архиватор – утилита для создания архивов. В отличие от Windows, где под архивом понимается файл, содержащий сжатые данные, в UNIX-подобных системах архив может быть и не сжатым, это просто информация для длительного хранения, обращение к которой происходит достаточно редко. Утилиты для сжатия файлов называются компрессорами.

Следует отметить, что все устройства в UNIX являются файлами. Все они монтируются в /dev. Примерами могут являться /dev/sda (первый жёсткий диск со SCSI-интерфейсом), /dev/hda (жёсткий диск с IDE-интерфейсом), /dev/null (специальное устройство, в него можно записать информацию любого объёма, при чтении выдаёт EOF, часто используется для перенаправления ненужных сообщений).

Специальными типами файлов являются также каналы (pipes) и порты (sockets, в некоторой литературе - гнёзда).

Имена папок в UNIX тоже имеют смысл. Например, /etc – директория, содержащая конфигурационные файлы (сокращение от et cetera – «и так далее»), /bin содержит исполняемые файлы (binary), /home – домашние директории пользователей, /lib – библиотеки (libraries) и т.д.

Каждая директория имеет минимум две ссылки: ‘.’ – ссылка на саму же директорию, и ‘..’ – ссылка на родительскую директорию. Эти ссылки являются жёсткими.

**Основные команды, используемые при работе с ФС**

В данном разделе приводятся утилиты, использующиеся при работе с файлами и файловыми системами. Некоторые из них были использованы в предыдущих ЛР.

mount – подключение (монтирование) файловой системы.

umount – отключение файловой системы.

mkfs – создание новой файловой системы. Существуют специализированные утилиты для создания ФС определённого типа (например, mkfs.ext4, mkfs.ntfs, полный список на рисунке 1).



Рисунок 1 – Утилиты mkfs

mke2fs – создание ext2/3/4 ФС.

fdisk – утилита для управления разделами диска.

df – отображает информацию об использовании диска.

du – отображает информацию об использовании диска. Разница показана на рисунке 2. Более подробную информацию можно узнать в man.

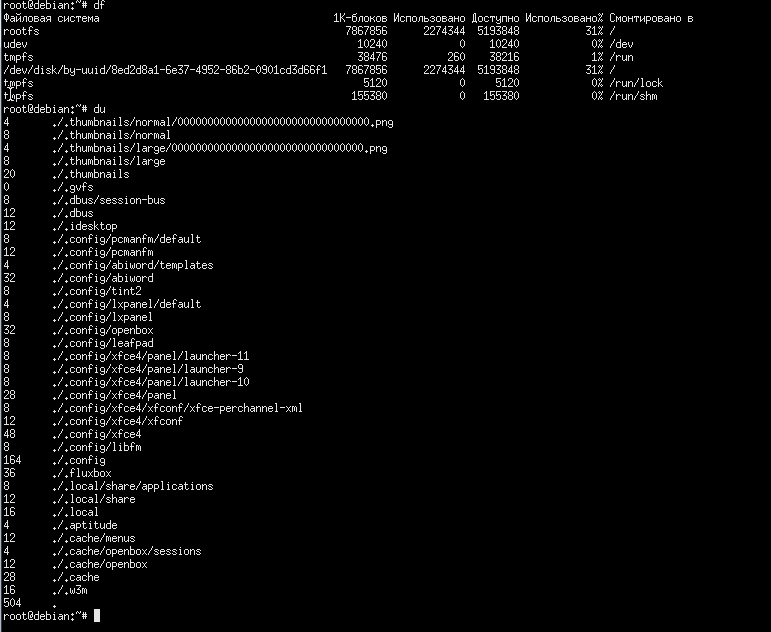


Рисунок 2 – Вывод утилит df и du

ls – вывод списка директорий (аналогичные утилиты – dir, vdir).

cp – копирование файла.

mv – перемещение файла (альтернативное применение – переименование).

rm – удаление.

cd – смена директории.

dd – утилита для копирования и конвертации файлов.

ln – создание ссылки.

pwd – вывод рабочего каталога.

mkdir – создание папки.

rmdir – удалить пустую папку.

basename – убирает из имени файла имена каталогов и суффиксы (преобразование полного пути в имя файла).

dirname – преобразует полный путь к файлу в имя родительской директории.

cat – просмотр файла.

head – вывод первых строк файла.

tail – вывод последних строк файла.

less – текстовый редактор.

od – просмотр в восьмеричном, десятичном или шестнадцатеричном формате. Пример работы показан на рисунке 3.



Рисунок 3 – Утилита od

xxd – просмотр файлов в шестнадцатеричном или двоичном формате. Сравнение с od показано на рисунке 4.

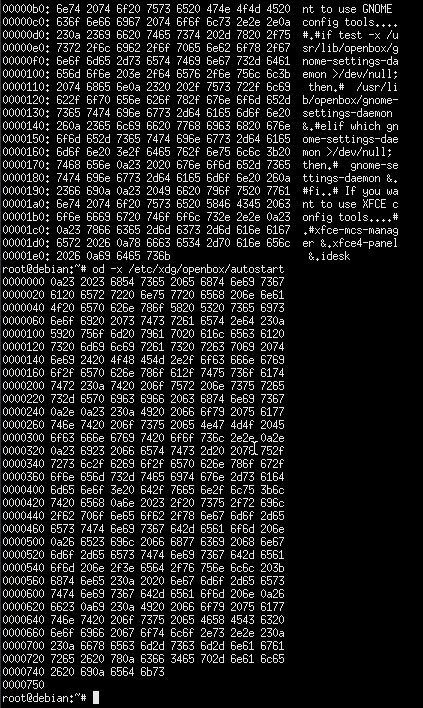


Рисунок 4 – Вывод утилит xxd и od

touch – создание пустого файла.

umask – задание маски прав доступа для новых файлов и директорий.

stat – вывод атрибутов файлов и директорий.

file – определение типа файла.

size – определение размера файла.

chown – изменение владельца или группы для файла или директории.

chgrp – изменение группы файла или директории. Может быть исполнена обычным пользователем, но в таком случае могут быть изменены только те группы, членами которых является пользователь.

chmod – смена прав доступа.

chattr – смена атрибутов файла в ext2.

cksum – расчет количества байт и контрольной суммы файла.

fuser – идентификация процессов, использующих указанные файлы или порты.

find – утилита поиска.

whereis – поиск файлов определённого приложения. Примеры использования и сравнение показаны на рисунке 5.



Рисунок 5 – Утилиты find и whereis

which – отображает полный путь к программе или скрипту. Пример показан на рисунке 6.



Рисунок 6 – Утилита which

cmp – утилита сравнения, может быть использована для файлов.

diff – утилита сравнения файлов, выводящая разницу. Утилита diff3 сравнивает 3 файла.

cpio – архиватор.

tar – компрессор tar.

xz – компрессор xz, на рисунке 7 показаны все утилиты этого пакета.



Рисунок 7 – Утилиты для работы с форматом xz

bzip2 – компрессор.

gzip – компрессор.

mkswap – создание файла подкачки (swap).

mktemp – создание временного файла.

**Файловая система ext4**

Данная секция описывает некоторые особенности файловой системы ext4 и содержит в себе практическую часть.

Некоторые особенности ext4fs:

* Экстенты (extents) – вместо поблочной адресации допустимо адресовать до 128 Мб идущих подряд блоков одним дескриптором.
* Контрольная сумма в журнале – в отличие от ext2, ext3 и ext4 являются журналируемыми файловыми системами, в ext4 хранится контрольная сумма транзакций диска.
* Мультиблочное выделение (выделение блоков группами, multiblock allocation) – файловая система хранит данные не только об отдельных свободных блоках, но и о группах свободных блоков. При записи файла производится поиск группы блоков, на которые данные могут быть записаны без фрагментации.
* Отложенное выделение (delayed allocation) – выделение блоков происходит перед физической записью на диск (а не при системном вызове write).
* Предварительное выделение (persistent pre-allocation) – возможно заранее зарезервировать место для записи файла.
* Версия inode (индексного дескриптора) – у индексного дескриптора появился номер, увеличивающийся каждый раз при изменении дескриптора.
* Обратная совместимость – файловые системы ext2 и ext3 могут быть монтированы как ext4.

Файловая система ext4 используется по умолчанию в дистрибутив Debian 7.2, который изучается в качестве примера в данной серии ЛР.

В качестве практики будет проведено исследование возможностей утилит ls и ln.

Некоторые ключи, использующиеся с утилитой ls:

-F – добавлять ‘/’ для каждого каталога, ‘|’ для каналов, ‘\*’ для исполняемых файлов.

-R – рекурсивная выдача каталогов (пример на рисунке 8).

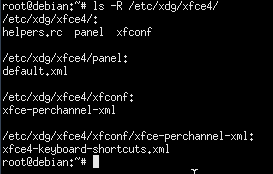


Рисунок 8 – Утилита ls с опцией рекурсивной выдачи каталогов

-a – включает в список файлы, начинающиеся с ‘.’ (пример на рисунке 9).

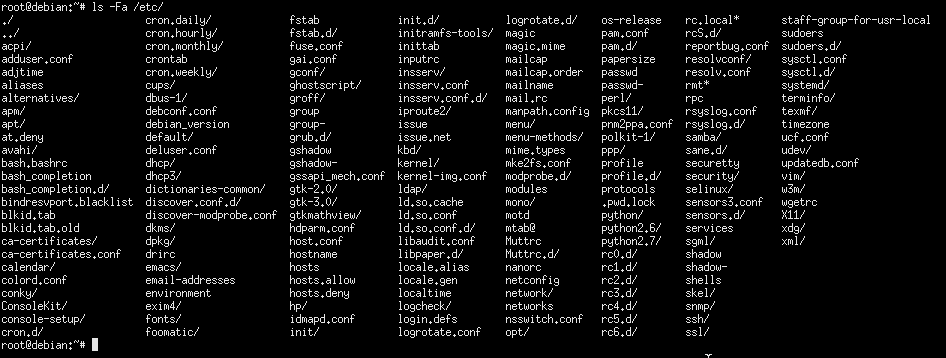


Рисунок 9 – Утилита ls с опциями суффиксов и показа всех файлов

-i – показывать номера inode (индексных дескрипторов).

-l – выдавать следующие данные: тип файла, права доступа, количество ссылок, имя владельца, имя группы, размер (в байтах) файла, штамп времени (временной штамп, timestamp) и имя файла. Кодирование типов файлов:

- – регулярный (обычный) файл;

d – каталог;

b – блочное устройство;

c – символьное устройство;

p – канал (pipe);

l – символьная ссылка;

s – порт (socket).

-d – обрабатывать каталог как файл.

-r – сортировка в обратном порядке.

-t – сортировать по штампу времени.

Примеры работы утилиты ls с некоторыми из этих опций показаны на рисунке 10.

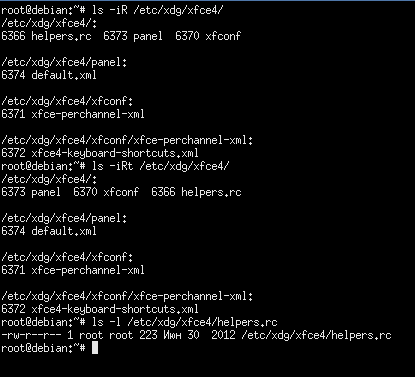


Рисунок 10 – Вывод утилиты ls с различными аргументами

*В качестве дополнительного задания покажите работу (с различными опциями) следующих утилит: cmp, diff, fuser.*

Утилита ln позволяет создавать как жёсткие, так и символьные ссылки. Создание жёсткой ссылки на файл:

ln file hardlink

С утилитой ln используются следующие ключи:

-s – создать символьную ссылку.

-d – сделать жёсткую ссылку для директории (только для суперпользователя).

-b – делать резервные копии файлов, которые будут удалены или изменены.

Пример создания символьной ссылки показан на рисунке 11.



Рисунок 11 – Работа с символьной ссылкой

Также значения ссылок можно считывать с помощью утилиты readlink:

readlink /root/symlink

Практическое задание:

1. Создайте пустой файл (с помощью touch).
2. Отобразите о нём информацию (с помощью ls).
3. Сделайте на него жёсткую и символьную ссылки. Посмотрите, как изменится информация о файле.
4. Внесите изменения в созданный файл (например, с помощью leafpad) и сохраните их. Откройте символьную ссылку с помощью leafpad. Сделайте то же самое с жёсткой ссылкой.

Ко всем пунктам задания должны быть приложены скриншоты.

**Файловая система NFS**

NFS (Network File System), по сути, представляет собой не самостоятельную файловую систему, а протокол для сетевого доступа к файловым системам. NFS абстрагируется от файловых систем сервера и клиента, можно представить, что NFS – это файловая система над файловыми системами.

Сервер NFS экспортирует свои каталоги, которые будут использоваться клиентами (обычно их список хранится в /etc/exports). Клиенты монтируют эти каталоги, монтируемые каталоги становятся частью иерархии файловой системы клиентов, хотя на самом деле они остаются на сервере и доступ к ним осуществляется с помощью механизма удалённого вызова процедур. На рисунке 12 показан пример монтирования удалённых файловых систем из книги Э.Таненбаума «Современные операционные системы», каталоги показаны в виде квадратов, файлы – в виде кружков.

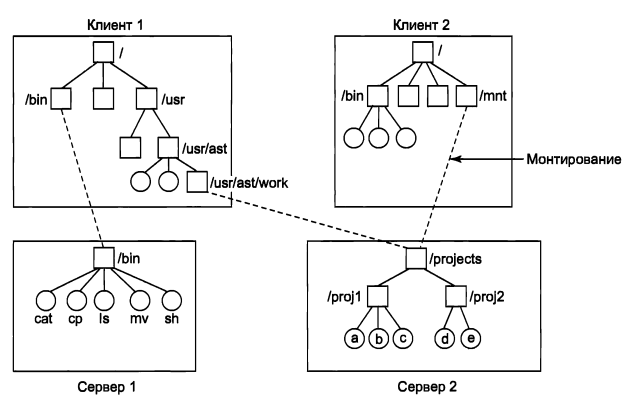


Рисунок 12 – Монтирование удалённых каталогов

NFS в Linux реализована с помощью нескольких уровней. Верхний уровень – уровень системных вызовов, он алогичен таковому для локальных ФС (обработка системных вызовов открытия файла, записи и т.д.). После обработки системный вызов передаётся на уровень виртуальной файловой системы (VFS). Виртуальная файловая система обслуживает таблицы, содержащие записи виртуальных индексных дескрипторов (virtual inode, v-узел) для всех открытых файлов. Нижний уровень – локальные файловые системы. Пример организации NFS из книги «Современные операционные системы» показан на рисунке 13.

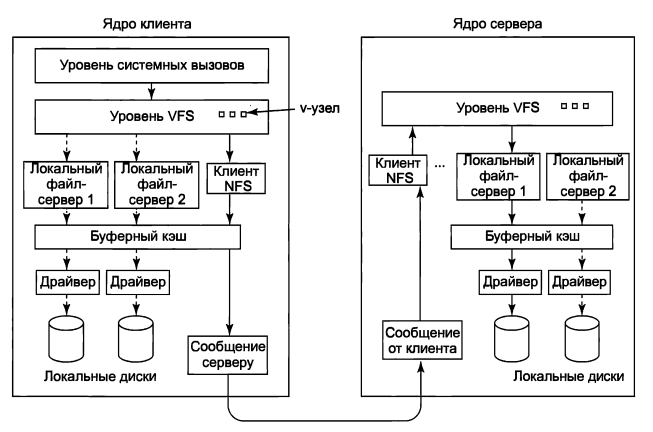


Рисунок 13 – Структура уровней файловой системы NFS

На сегодняшний день актуальной версией NFS является NFS v4. В отличие от своих предшественниц, 4 версия является ФС с сохранением состояния. Это стало возможным благодаря введению составных (compound) операций: вместо одной большой транзакции передавалось множество транзакций с малым объёмом данных.

В качестве практики будет показан краткий пример настройки NFS-сервера.

1. Выбор экспортируемых папок и настройка сервера.

Отредактируйте файл /etc/exports таким образом, чтобы он содержал записи следующего вида:

directory/FS client1 (options) client2 (options) …

Некоторые общие опции:

rw – доступ для чтения и записи (по умолчанию – только чтение, ro);

async – использовать асинхронный метод обработки (по умолчанию – синхронный, sync);

Некоторые опции отображения пользователей:

root\_squash – не позволять пользователю root обращаться к смонтированному тому (противоположность – no\_root\_squash);

all\_squash – вместо UID и GID используется запись анонимного пользователя, используется для открытых томов (по умолчанию – no\_all\_squash);

anonuid/anongid – меняют данные учётной записи анонимного пользователя на указанные.

Пример файла конфигурации показан на рисунке 14.

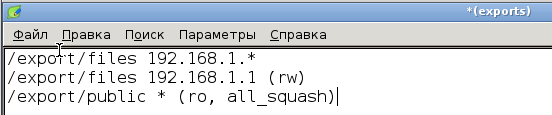


Рисунок 14 – Настройка доступа к экспортируемым каталогам

На рисунке записи означают следующее:

Предоставить доступ всем компьютерам в сети 192.168.1. Все настройки – по умолчанию.

Предоставить клиенту 192.168.1.1 права на чтение и изменение каталога.

Предоставить всем клиентам доступ к /export/public, но только для чтения и только под учётной записью анонимного пользователя.

1. Настройка клиента.

Клиент должен монтировать экспортируемую сервером файловую систему командой следующего вида:

mount server:directory local\_mount\_point

Например, сервер имеет IP 192.168.0.1, точка монтирования - /mnt:

mount 192.168.0.1:/export/files /mnt

**Файловая система GPFS**

GPFS (General Parallel File System) – параллельная кластерная файловая система от IBM, впервые представлена IBM в 1998 году. Используется в кластерах (часто в IBM x), при организации облачных инфраструктур, совместно с суперкомпьютерами (IBM Blue Gene).

Файл в GPFS может быть распределён по нескольким дискам, находящимся на различных узлах, при этом для работы с файловой системой могут использоваться обычные утилиты UNIX, как в случае с локальной ФС.

Преимущества GPFS:

* Высокая производительность – ресурсы GPFS доступны одновременно многим процессорам на многих узлах.
* Высокий показатель восстанавливаемости и высокая доступность – GPFS является журналируемой файловой системой, причём для каждого узла ведётся собственный журнал. GPFS включает в себя утилиты для гибкой настройки и администрирования.
* Гибкость – узлы могут добавляться и удаляться без перемонтирования файловой системы.

Рисунок 15 показывает компоненты GPFS и их взаимодействие.

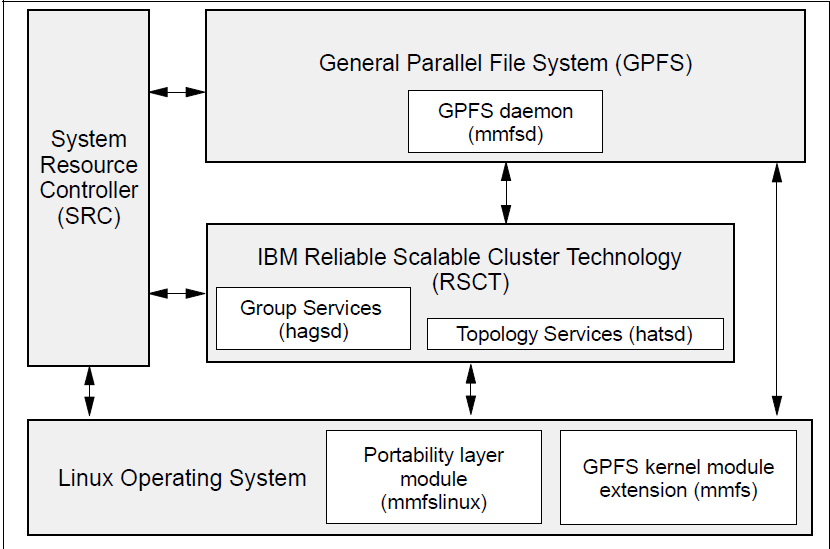


Рисунок 15 – Компоненты GPFS

IBM также отмечает, что, в отличие от NFS, GPFS является самостоятельной файловой системой, что положительно сказывается на быстродействии и удобстве администрирования.

В данной лабораторной работе будет использоваться кластер МГТУ «Королёв» (IBM System Cluster 1350 “Korolyov”). Логин и пароль для доступа можно получить у преподавателя. На рисунке 16 показан пример входа в систему с помощью утилиты ssh, изучите его и повторите с полученным логином и паролем. Соединение может занять некоторое время.

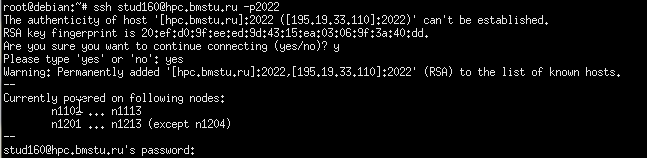


Рисунок 16 – Диалог авторизации ssh

Проверьте имя своей домашней директории (введя $HOME).

Помимо стандартных утилит UNIX, GPFS использует свои специальные утилиты. Ниже перечислены некоторые из них:

mmgetacl – получить список контроля доступа (ACL, Access Control List).

mmputacl – установить список контроля для файла.

mmeditacl – изменить список контроля для файла.

mmdelacl – удалить список контроля.

mmlscluster – показать информацию о кластере GPFS (недоступно для обычных пользователей).

mmlsmgr – отобразить управляющие узлы для различных ФС.

mmlsnsd – отобразить разделяемые диски NSD (Network Shared Disk), недоступно для пользователя, пример использования показан на рисунке 17.

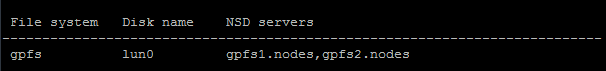


Рисунок 17 – Вывод утилиты mmlsnsd

Списки контроля доступа представляют собой обычные текстовые файлы, формат похож на формат списков доступа утилит getfacl/setfacl, рассмотренных в предыдущей ЛР. Пример такого списка из официальной документации показан на рисунке 18.

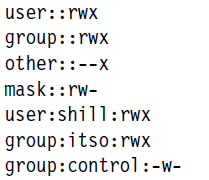


Рисунок 18 – Пример ACL в GPFS

Пример установки списка контроля доступа:

1. Создание списка ACL:

cat > acl

В файле набрать данные, показанные на рисунке 18, обратите внимание, что используется 4 бита для каждого поля. Для окончания редактирования нажать Ctrl-D. Проверить данные можно командой cat acl.

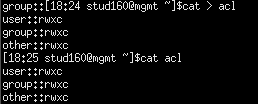


Рисунок 18 – Создание списка контроля доступа.

1. Создать файл с помощью touch:

touch $HOME/file

1. Получить список контроля доступа для файла (рисунок 19).

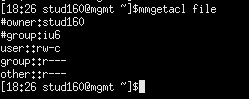


Рисунок 19 – Список контроля доступа созданного файла

1. Установить созданный список доступа проверить (рисунок 20).

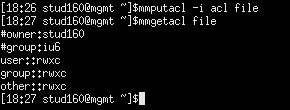


Рисунок 20 – Установка и проверка списка контроля для файла

Данные об операциях хранятся в файлах mmfs.log (ознакомьтесь с содержимым файла /var/adm/ras/mmfs.log.latest).

Практическое задание:

1. Создать свой файл в домашней директории.
2. Проверить его ACL несколькими способами.
3. Создать новый ACL.
4. Установить новый ACL, показать это выводом соответствующей команды.
5. Удалить созданные файлы.

Ко всем пунктам задания привести скриншоты.

Завершить сессию ssh можно нажатием Ctrl-D (в консоли должна появиться надпись «connection to hpc.bmstu.ru closed»).

**Выводы**

В данной ЛР были рассмотрены некоторые из применяющихся на данный момент в UNIX-подобных системах файловые системы. Также были рассмотрены основные утилиты для работы с файлами.

Файловая система ext4 представляет собой развитие файловой системы ext3. Она может содержать большее количество каталогов, адресовать б*о*льшие диски, эффективнее использует пространство при размещении больших файлов. ФС ext4 используется по умолчанию в ряде дистрибутивов, в том числе и Debian (начиная с версии 7.0).

Файловая система NFS основана на удалённом вызове процедур (RPC) и предоставляет протокол для взаимодействия с удалёнными ресурсами. NFS не является самостоятельной ФС, она позволяет монтировать папки из различных файловых систем.

GPFS является высокопроизводительной параллельной файловой системой. В отличие от NFS, GPFS является самостоятельной ФС. Эта ФС является проприетарной. Она может использоваться как в системах высокой доступности, так и в высокопроизводительных системах.

Контрольные вопросы:

1. Ярлык и символьная ссылка – одно и то же?
2. Зачем используется umask? В чём отличие от chmod?
3. Зачем может использоваться версия inode?
4. В чём достоинство NFS?

**Порядок выполнения работы.**

1. Ознакомиться с теоретическими сведениями по файловым системам.

2. Изучить и использовать утилиты и команды для работы с файлами.

3. Пройти тестирование по теоретическому материалу.

*Отчет должен включать*:

1. название работы и ее цель;
2. описание команд и результаты работы.