ЛАБОРАТОРНАЯ РАБОТА № 1

«Исследование файловых объектов с правами пользователя»

Цель работы: Изучение основ OS Linux, касающихся прав доступа к файлам, действий над обычными файлами и использования жестких и символических ссылок.

Основные теоретические сведения и методические рекомендации

Права доступа к файлу

С самого начала OS Unix разрабатывалась как многопользовательская среда, в которой разным пользователям предоставляются различные права доступа. Но помимо этого пользователи могут быть приписаны к группам, и затем права доступа к файлам и программам могут предоставляться и пользователям, и группам. Вся эта многопользовательская поддержка и точный контроль над файловой системой были перенесены в Linux, сделав ее одной из самых надежных и безопасных вычислительных сред. Достигается это сочетанием различных концепций, касающихся пользователей, групп и прав доступа к файлам.

У всех файлов есть набор атрибутов, определяющих, что можно делать с файлом, а чего нельзя, а также у кого есть права доступа к файлу. Это и есть права доступа (разрешения), которые бывают трех основных типов.

* Чтение (Read) — файл с таким правом доступа можно читать, но нельзя в него писать, и его нельзя выполнять.
* Запись (Write) — в файл с таким типом разрешений можно записывать информацию.
* Выполнение (Execute) — файл с таким правом доступа можно выполнять. Другими словами, это разрешение применимо к приложениям.

Типы файлов

* В большинстве реализаций файловых систем определены семь типов файлов:
* обычные файлы;
* каталоги;
* файлы байт-ориентированных (символьных) устройств;
* файлы блочно-ориентированных (блочных) устройств;
* локальные сокеты;
* именованные каналы (реализующие принцип обслуживания FIFO - First in First Out), т.е. “первым поступил — первым обслужен”;
* символические ссылки.

Каталоги

Каталог хранит именованные ссылки на другие файлы. Он создается командой *mkdir* и удаляется (при условии, что он пуст) командой *rmdir*. Непустые каталоги можно удалить командой *rm -г*.

Специальные ссылки “ . ” и “ . . ” обозначают сам каталог и его родительский каталог соответственно. Такие ссылки нельзя удалить. Поскольку корневой каталог находится на вершине иерархии, ссылка “ . . ” в нем эквивалентна ссылке “ . ”.

Имя файла в действительности хранится в родительском каталоге, а не в самом файле. На файл можно ссылаться из нескольких каталогов одновременно и даже из нескольких элементов одного и того же каталога, причем у всех ссылок могут быть разные имена. Это создает иллюзию того, что файл одновременно присутствует в разных каталогах.

Использование «жестких» и символических ссылок

“Жесткие” (фиксированные) ссылки (которые следует отличать от символических, или “мягких”) можно считать синонимами для исходных файлов, и с точки зрения файловой системы все ссылки на файл эквивалентны. Файловая система подсчитывает количество жестких ссылок на каждый файл и при удалении файла не освобождает блоки данных до тех пор, пока не будет удалена последняя жесткая ссылка на него. Ссылки не могут указывать на файл, находящийся в другой файловой системе.

Жесткие ссылки создаются командой *ln* и удаляются командой *rm*. Важно понимать, что жесткие ссылки не являются особым типом файлов, просто файловая система позволяет создавать несколько ссылок на один файл.

***Не только содержимое, но и атрибуты файла, в частности, права доступа и идентификатор владельца, являются общими для всех жестких ссылок.***

Символическая, или “мягкая”, ссылка позволяет вместо имени файла указывать его псевдоним. Столкнувшись при поиске файла с символической ссылкой, ядро извлекает хранящееся в ней имя.

***Разница между жесткими и символическими ссылками состоит в том, что жесткая ссылка является прямой, т.е. указывает непосредственно на индексный дескриптор файла, тогда как символическая ссылка указывает на файл по имени.***

Файл, адресуемый символической ссылкой, и сама ссылка представляют собой разные объекты файловой системы. Символические ссылки создаются командой *ln* -*s* и удаляются командой rm. Они могут содержать произвольное имя, т.е. разрешается указывать на файлы, хранящиеся в других файловых системах, и даже на несуществующие файлы.

В традиционной модели файловой системы UNIX и Linux каждому файлу соответствует набор из девяти битов режима. Они определяют, какие пользователи имеют права читать файл, записывать в него данные или запускать его на выполнение. Вместе с другими тремя битами, которые в основном влияют на работу исполняемых файлов, этот набор образует код, или режим доступа к файлу.

Двенадцать битов режима доступа хранятся вместе с четырьмя дополнительными битами, определяющими тип файла. Эти четыре бита устанавливаются при создании файла и не подлежат изменению. Биты режима доступа могут изменяться владельцем файла или суперпользователем с помощью команды *chmod (“change mode”* — изменить режим).

Код доступа удобно записывать в виде восьмеричного числа, так как каждая цифра в нем представляется тремя битами. Три старших бита (в коде доступа им соответствуют восьмеричные значения 400, 200 и 100) служат для управления доступом к файлу его владельца. Вторые три бита (40, 20 и 10) задают доступ для членов группы. Последние три бита (4, 2 и 1) определяют доступ к файлу остальных пользователей. Старший бит каждой триады — это бит чтения, средний — бит записи, младший — бит выполнения.

Каждый пользователь попадает только в одну из категорий прав доступа. В случае неоднозначности выбираются самые строгие права. Для обычного файла бит чтения означает возможность открывать и читать файл. Бит записи позволяет изменять содержимое файла, в том числе полностью очищать его. Правом удаления и переименования файла управляют биты, заданные для его родительского каталога, поскольку именно там хранится имя файла.

Установкой бита выполнения задается разрешение запускать файл. Существует два типа исполняемых файлов: бинарные файлы, которые выполняются непосредственно центральным процессором, и сценарии, обрабатываемые интерпретатором команд или какой-нибудь другой аналогичной программой. В соответствии с принятыми соглашениями, сценарии начинаются со строки примерно такого вида:

*#! /usr/bin/perl*

Здесь задается соответствующий интерпретатор. Текстовые исполняемые файлы, в которых не указан конкретный интерпретатор, считаются сценариями *bash* или *sh*.

Биты, которым в коде доступа соответствуют восьмеричные значения 4000 и 2000, — это биты смены идентификатора пользователя ( *setuid* ) и идентификатора группы ( *setgid* ) . Если эти биты установлены для исполняемых файлов, они позволяют программам получать доступ к файлам и процессам, которые при прочих обстоятельствах недоступны пользователю, выполняющему эти программы.

Пользователи и группы

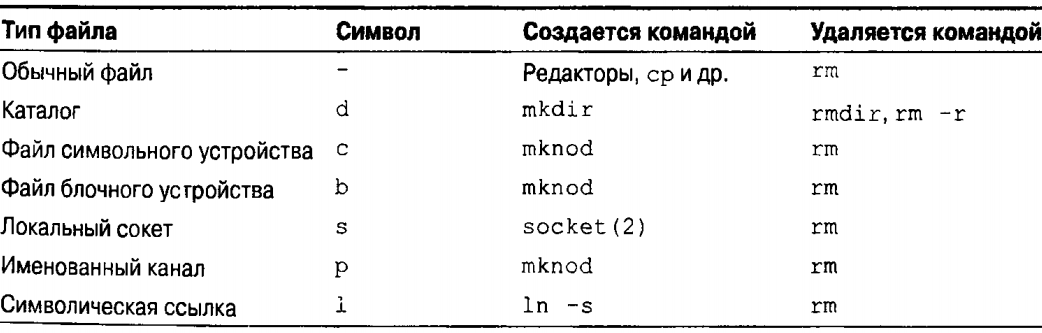
У каждого файла есть два дополнительных атрибута, представляющих строковые значения: один для Владельца (Owner) и другой для Группы (Group), которой принадлежит файл. Есть и третий атрибут, который называется Остальные (World или иногда Others) и автоматически присваивается, если пользователя нет в атрибутах владельца и группы.

Если бит setgid установлен для каталога, то создаваемые в нем файлы будут принимать идентификатор группы каталога, а не группы, в которую входит владелец файла. Это упрощает пользователям, принадлежащим к одной группе, совместный доступ к каталогам. Следует также учитывать, что для исполняемого файла бит setgid имеет другое значение.

Бит setgid можно устанавливать для файлов, не являющихся исполняемыми. Тем самым запрашивается специальный режим блокирования файлов при их открытии. Бит, которому в коде доступа соответствует восьмеричное значение 1000, называется дополнительным (sticky bit). Если дополнительный бит установлен для каталога, то файловая система позволит удалять и переименовывать его файлы только владельцу каталога, владельцу файла или суперпользователю. Иметь одно лишь право записи в каталог недостаточно. Такая мера позволяет несколько лучше защитить каталоги вроде */tmp*.

Команда *ls*: просмотр атрибутов файла

Файловая система хранит для каждого файла около сорока информационных полей, большая часть из которых используется самой файловой системой. Администратора, в основном, интересует количество жестких ссылок, владелец, группа, код доступа, время последнего обращения и последней модификации, размер и тип файла. Всю эту информацию можно получить с помощью команды *ls* - *l* (или команды *ls* - *ld* в случае каталога; без флага - d команда *ls* перечисляет содержимое каталога). Пример символов в листинге выполнения команды *ls* приведен в таблице:



Для каждого файла хранится также время последнего изменения атрибутов. Традиционное название этого поля (“ctime” от “change time” — время изменения) многих вводит в заблуждение, так как они полагают, что это время создания файла. На самом деле здесь зафиксировано время последнего изменения одного из атрибутов файла (владелец, код доступа и так далее), но не его содержимого.

Рассмотрим следующий пример записи, полученной по команде  ***$ l s- l /bin /gzip :***

***-rwxr-xr-x 3 root root 62100 May 08 2014 /bin/gzip***

В первом поле задается тип файла и режим доступа к нему. Поскольку первый символ — дефис, значит, перед нами обычный файл (см. табл. выше). Следующие девять символов — это три набора битов режима. Порядок наборов таков: владелец, группа, другие пользователи. В листинге команды *ls* биты режима представляются буквами г, w и х (чтение, запись и выполнение). В показанном примере владелец имеет все права доступа к файлу, а остальные пользователи — только право на чтение и выполнение.

Второе поле листинга представляет собой счетчик ссылок на файл. В данном случае здесь стоит цифра 3, свидетельствующая о том, что / b i n / g z i p — одно из трех имен файла (два других — / b i n / g u n z i p и / b i n / z c a t ) . Каждый раз при создании жесткой ссылки на файл этот счетчик увеличивается на единицу. Символические ссылки в счетчике не учитываются. Любой каталог имеет минимум две жесткие ссылки: из родительского каталога и из специального файла “ . ” внутри самого каталога.

Следующие два поля определяют владельца и группу файла. В данном примере файл принадлежит пользователю root и одноименной группе. В действительности ядро хранит эти данные не в строковом виде, а в виде идентификаторов. Если символьные версии идентификаторов определить невозможно, в этих полях будут отображаться числа.

В следующем поле отображается размер файла в байтах. Рассматриваемый файл имеет размер 62100 байт. Далее указана дата последней модификации файла: 8 мая 2014 г.

В последнем поле листинга приведено имя файла: / b i n / g z i p .

При поиске жестких ссылок может пригодиться команда *ls -i* , отображающая для каждого файла номер его индексного дескриптора. Не вдаваясь в детали реализации файловой системы, скажем, что этот номер представляет собой индекс таблицы, в которой перечислены все файлы системы. Именно на индексные дескрипторы ссылаются файловые записи каталогов. Жесткие ссылки, указывающие на один и тот же файл, будут иметь одинаковый номер.

Для того чтобы составить представление о системе ссылок можно воспользоваться командой *ls -li* для определения числа ссылок и номера индексного дескриптора какого-нибудь файла, а затем найти его “двойников” с помощью команды find. Другими важными опциями команды *ls* являются - а , отображающая все записи в каталоге (даже те файлы, имена которых начинаются с точки), - t , выполняющая сортировку файлов по времени изменения (или - t r , осуществляющая сортировку в обратном хронологическом порядке), -F , отображающая имена файлов с выделением каталогов и исполняемых файлов.

Команда *chmod*: изменение прав доступа

Код доступа к файлу можно изменить с помощью команды *chmod*. Такое право есть лишь у владельца файла и пользователя root . В первых UNIX-системах код доступа задавался в виде восьмеричного числа. В современных версиях поддерживается также система мнемонических обозначений. Первый способ удобнее для системного администратора, но при этом можно задать только абсолютное значение кода доступа. В случае использования мнемонического синтаксиса разрешается сбрасывать и устанавливать отдельные биты режима.

Первым аргументом команды *chmod* является спецификация прав доступа. Второй и последующий аргументы — это имена файлов, права доступа к которым подлежат изменению. При использовании восьмеричной формы записи первая цифра относится к владельцу, вторая — к группе, а третья — к другим пользователям. Если необходимо задать биты *setuid/setgid* или дополнительный бит, следует указывать не три, а четыре восьмеричные цифры: первая цифра в этом случае будет соответствовать трем специальным битам.

Команды *chown* и *chgrp* предназначены для изменения владельца файла и группы соответственно. Синтаксис этих команд подобен синтаксису команды *chmod*, за исключением того, что первый аргумент содержит нового владельца или группу соответственно. Для того чтобы изменить группу файла, необходимо либо быть владельцем файла и входить в назначаемую группу, либо быть пользователем root. В большинстве систем предусмотрены средства настройки поведения команды *chown* в зависимости от выполняемого процесса.

Как и команда *chmod*, команды *chown* и *chgrp* имеют флаг -R, который задает смену владельца или группы не только самого каталога, но и всех его подкаталогов и файлов.

Программа работы:

1. Изучите теоретический материал, посвященный правам доступа к файлам, действиям над обычными файлами и использованию жестких и символических ссылок.

2. Зарегистрируйтесь в первой консоли как *root* .

3. Для исследования файловых объектов вам потребуются еще два пользователя, учетные записи которых предстоит создать. Для этого воспользуйтесь краткой формой команды useradd –m user1 .

Аргумент –m предписывает создание домашнего каталога пользователя с его именем. После этого пользователю необходимо присвоить пароль. Это делается командой passwd user1.

На запрос введите простой пароль 1234567890.

4. Повторите команды п. 3 для создания учетной записи пользователя user2. Помните, что оба пользователя по умолчанию являются членами одной группы users.

5. Аналогично откройте третий текстовый терминал и зарегистрируйтесь как user1.

С помощью Alt-F2 откройте второй текстовый терминал и зарегистрируйтесь как user2 с паролем 1234567890.

7. Нажатием Alt-F1 вернитесь в первую консоль. Теперь, переключая консоль, вы можете работать с объектами операционной системы от имени двух разных пользователей и администратора системы.

Основная часть задания выполняется с правами обычных пользователей. Переходите в первую консоль и используйте права root только при выполнении соответствующих пунктов задания.

Будьте внимательны при вводе команд!

Помните, что операционная система не контролирует действий администратора и ваше ошибочное действие может привести к краху системы. Ориентируйтесь на характерную форму приглашения к вводу команд, в котором отражены ранг и имя пользователя.

8. С правами user1 с помощью команды cd /root попробуйте войти в каталог суперпользователя. Объясните полученный результат.

Затем с помощью команды ls -la / просмотрите список основных каталогов и укажите, в какие из них вы имеете право войти.

Проверьте права доступа к домашним каталогам пользователей /home/user1 и /home/user2: они должны быть установлены в 755.

10. Переключитесь в консоль администратора и создайте два новых временных каталога: mkdir -m 777 /home/temp1 и mkdir -m 1777 /home/temp2

11. Вернитесь в консоль user1 и, пользуясь командой mkdir, создайте в домашнем каталоге пользователя /home/user1 четыре каталога с именами qu1, qu2, qu3, qu4. При создании каталогов объявите следующие права доступа к ним: (qu1 - 777, qu2 - 404, qu3 - 1333, qu4– 505).

Пример: cd; mkdir -m 777 qu1

С помощью команды ls /home/user1 убедитесь в том, что каталоги созданы. Какие из предоставленных прав кажутся вам лишенными смысла? Почему?

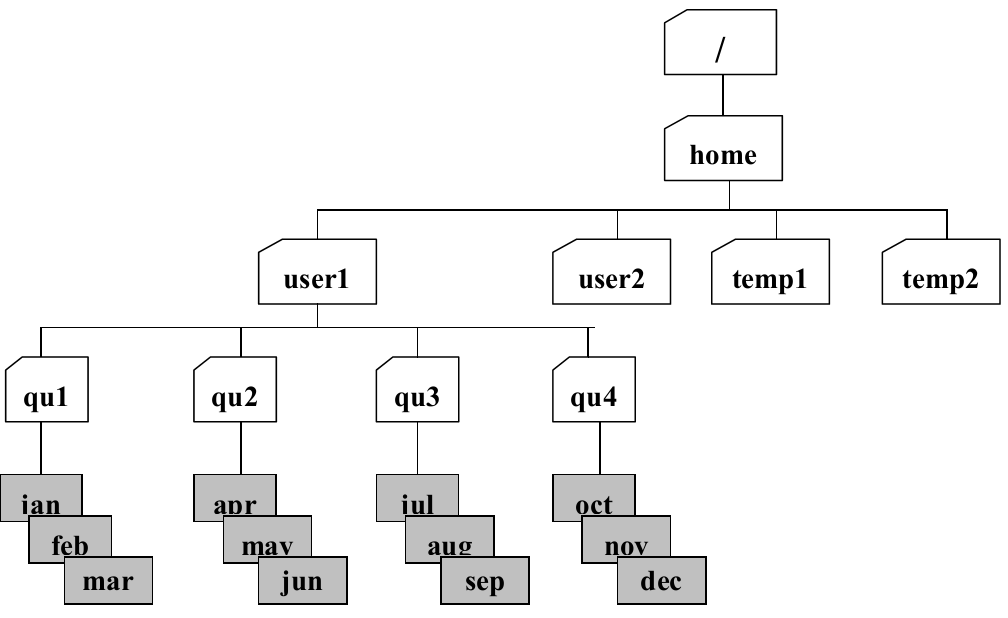


Рис. Пояснение к п.п.10-13.

12. Задайте права доступа к файлам «по умолчанию». Для этого установите маску доступа umask 022.

Поясните, какие права к вновь создаваемым файлам и каталогам будут предоставляться пользователю, членам его группы и остальным пользователям.

13. В каждом из каталогов создайте по три текстовых файла с именами (jan, feb, mar), (apr, may, jun), (jul, aug, sep), (oct, nov, dec). В каждый файл запишите календарь на определенный месяц текущего года. Например, команда cal 1 2020 >jan создает в текущем каталоге файл jan и записывает в него календарь на январь 2020 года.

Не забывайте, что использование относительного (короткого) имени файла требует, чтобы вы находились в нужном каталоге. В противном случае следует указывать полный путь к создаваемому файлу. Для навигации по каталогам используйте команды cd и pwd.

В каком случае создание файлов не удалось? Почему?

14. С помощью команды chmod установите нужные права доступа в «недоступные» каталоги qu2, qu4 и создайте там указанные файлы. После этого верните каталогам прежние права доступа.

15. С помощью команд cd и ls войдите в каждый из созданных каталогов и просмотрите список созданных файлов. При просмотре содержимого каталогов используйте два режима: ls без аргументов и ls -l.

В каких случаях не удалось войти в каталог? В каких случаях не удалось посмотреть список файлов? Почему?

16. Прочитайте содержимое одного из файлов в «темном» каталоге (например, cd /home/user1/qu3; cat aug). Сделайте выводы по результату чтения.

17. Перейдите во 2-ю консоль и с правами пользователя user2 войдите в каталог /home/user1/qu1. Создайте в каталоге /home/user2 новый файл quart1 путем конкатенации нескольких имеющихся: cat jan feb mar >/home/user2/quart1

С помощью команды file определите тип созданного файла. Попробуйте вывести его на экран командой cat. Что представляет собой данный файл?

18. С помощью команды chmod установите права доступа 077 на созданный файл quart1. Вновь попробуйте прочесть его.

Ответьте, почему владельцу файла запрещается доступ, если файл доступен для всех? Что необходимо сделать, чтобы вернуть владельцу права на доступ?

19. Установите для файла quatr1 права на доступ 4700.

Кому и какие права вы при этом предоставили? Как воспользоваться этими правами? Какие из предоставленных прав не имеют смысла?

20. Перейдите в консоль администратора и передайте право владения на файлы may и aug пользователю user2 (команда chown). Поочередно из консолей user1 и user2 проверьте, как изменились права владения файлами после его передачи.

Может ли пользователь user2 воспользоваться предоставленными правами (пользователи user1 и user2 при создании учетных записей по умолчанию отнесены к одной группе users)?

21. Правами пользователя user1 из каталогов /home/temp1 и /home/temp2 c помощью команды ln создайте две «жесткие» ссылки на файл dec с именами dec\_h1 и dec\_h2 ln /home/user1/qu4/dec /home/temp1/dec\_h1

Чем созданные ссылки отличаются от исходного файла?

22. С помощью команды ln -s создайте из каталогов /home/temp1 и /home/temp2 две символические ссылки на файл dec с именами dec\_s1 и dec\_s2.

Чем отличаются созданные ссылки от исходного файла? Попытайтесь прочитать содержимое файлов символических ссылок. Что они собой представляют?

23. С правами пользователя user2 с помощью команды cp создайте в каталогах /home/temp1 и /home/temp2 по одной копии файла dec с другим именем (dec\_copy1).

Чем отличаются исходный файл и его копия (обратите внимание на то, кто является владельцем исходного файла и его копии)? Чем отличаются права доступа на эти файлы?

24. Вернитесь в консоль user1. С правами пользователя user1 создайте жесткие ссылки из его домашнего каталога на файл /bin/su (доступен обычным пользователям только на исполнение) и на файл /etc/shadow (для обычных пользователей недоступен).

Ответьте, какими правами на объектовый файл нужно обладать, чтобы создать на него жесткую ссылку? Какую выгоду получает обладатель жестких ссылок на недоступные файлы?

25. С правами пользователя user1 скопируйте в его домашний каталог утилиту /bin/mount. Сравните между собой оригинал и копию и укажите все отличия.

Сможет ли пользователь применить копию опасной утилиты во вред политике безопасности? Почему?

26. С помощью команды rm удалите файл dec.

Что произошло с «жесткими» и символическими ссылками на данный файл? Что произошло с его копиями? Что нужно сделать для того, чтобы файл перестал существовать (на логическом уровне)?

27. С правами user1 удалите файлы из каталогов /home/temp1 и /home/temp2.

Какие файлы не удалось удалить? Почему?

Попробуйте удалить оставшиеся файлы с правами пользователя user2. Объясните полученный результат.

28. Попробуйте удалить любой из каталогов qu1, qu2, qu3, qu4 с помощью команды rmdir (не удаляя предварительно из них файлов).

Объясните результат.

29. Войдите в консоль администратора и с правами root, пользуясь командой сhattr, заблокируйте файл feb от любых изменений (предварительно ознакомьтесь с синтаксисом команды). Установите параметр запрета любых операций, кроме добавления данных для файла mar. Вернитесь в консоль user1. С помощью команды lsattr -l проверьте наличие дополнительных атрибутов у файлов.

30. С правами пользователя user1 добавьте одну строку finish в конец файлов feb и mar . Воспользуйтесь для этого командой echo finish >> file\_name

Убедитесь в успешном завершении операции, объясните результат.

31. С правами пользователя user1 с помощью команды rm -rf последовательно удалите ранее созданные каталоги qu2, qu3, qu4 вместе с файлами. Объясните результат.

32. С правами пользователя user1 создайте в /home/user1 два каталога mkdir –m 400 src и mkdir –m 200 dst. Внутри каталога src командой echo 1234567890 > abc создайте текстовый файл (если для этого прав доступа к каталогу недостаточно, временно измените их). Для созданного файла установите права доступа chmod 100 abc.

33. Манипулируя правами доступа на созданные файловые объекты, установите минимально необходимые права, позволяющие копировать и перемещать указанный файл из каталога src в каталог dst. Зафиксируйте результаты в отчете.

34. Определите, какие минимально необходимые права на файловые объекты нужно иметь, чтобы выполнять копирование и перемещение файла от имени администратора.

35. После окончания лабораторной работы из консоли суперпользователя удалите все созданные вами файлы.

Соблюдайте аккуратность: с правами root и с помощью утилиты rm вы можете вызвать крах системы!

37. После выполнения работы командами exit завершите пользовательские сеансы во второй и третьей консолях, а из первой консоли с правами root выполните команду останова системы halt.

Представьте преподавателю **отчет о проделанной работе** с выводами по пунктам (ответы и выводы по соответствующим пунктам должны быть представлены в **письменной форме**!).

Файл отчета с наименованием ***Фамилия\_№ группы\_номер л/р*** следует отправить на адрес электронной почты *i.svatowsky@karazin.ua*

Контрольные вопросы

1. Какие дополнительные атрибуты можно присвоить файлу в файловой системе ext\*fs? Как эти атрибуты влияют на обеспечение конфиденциальности, целостности и доступности информации?

2. Как можно создать текстовый файл без помощи текстового редактора?

3. В чем различие между копиями файла, его «жесткими» и символическими ссылками? Для чего используются символические ссылки?

4. Для каких целей могут использоваться «темные» каталоги?

5. Какие права по отношению к файлам и каталогам вам необходимо иметь для копирования файла? Как изменяются при этом атрибуты копии?

На контрольные вопросы необходимо ответить до начала лабораторной работы!