Лабораторная работа №1
Дискреционный контроль доступа

Цель работы: Изучение и практическое освоение системных разрешений и механизмов управления доступом в ОС “Альт Рабочая станция”.

Приветствуем Вас в компании ООО “AU_Team”! Мы рады видеть Вас в качестве нового кандидата на позицию специалиста по безопасности нашей информационной системы. Ваша первая задача заключается в настройке разграничения доступа в ОС “Альт Рабочая станция”.

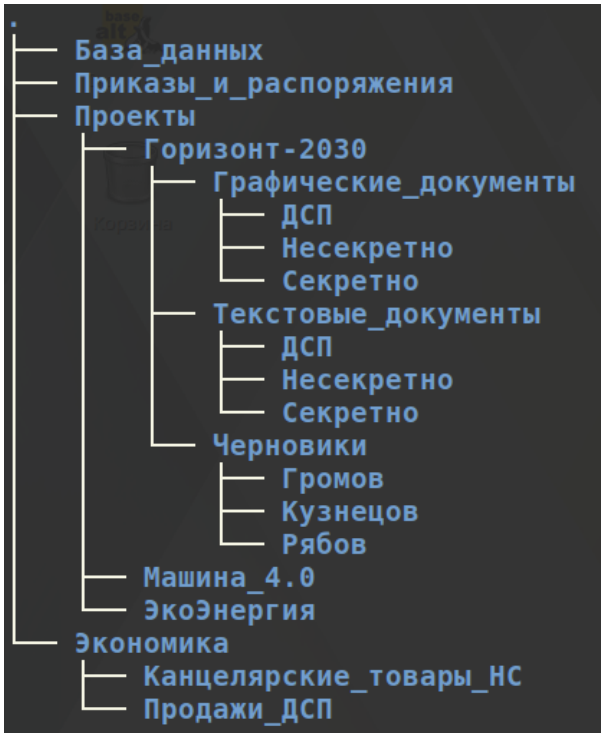
Для построения защищенной информационной системы рассмотрим организацию ООО “AU_Team”, занимающуюся разработкой инновационных технологий в различных областях промышленности. Несколько не связанных между собой групп специалистов ведут разработку самостоятельных инженерных проектов “ЭкоЭнергия”, “Машина 4.0”, “Горизонт-2030”. Сами проекты являются конфиденциальными программами, а их документация — охраняемыми данными.

Руководит организацией ООО “AU_Team” Петров С.В., он имеет максимальный уровень допуска к информации и возможность работы с документацией любого проекта. Экономист Лебедев М.Г. изучает финансовые данные и предлагает идеи для повышения прибыли. Администратор информационной системы (администратор безопасности) Чистяков А.В. имеет полный доступ к любым документам, может управлять настройками информационной системы и реализует на практике политику безопасности предприятия, в части, касающейся информационных технологий.

Документация проектов представляет собой ряд текстовых и графических электронных документов, обрабатываемых в единой системе и имеющих различный уровень конфиденциальности: «открытые данные», «конфиденциально» и «строго конфиденциально». Уровней конфиденциальности может быть и больше: «ограниченного доступа», «особо конфиденциально» и т. д. Система уровней конфиденциальности определяется градацией информационных ресурсов в зависимости от величины и характера (качества) ущерба при неограниченном распространении соответствующей информации. В СЗИ различных производителей уровни конфиденциальности имеют различное наименование. Так, «открытые данные» иногда именуют «несекретными» или «общедоступными»; «конфиденциальные» называют «ДСП», «служебная тайна» или «конфиденциальные документы»; «строго конфиденциальные документы» — «секретными». В дальнейшем мы для обозначения уровней конфиденциальности будем пользоваться терминами «Несекретно», «ДСП», «Секретно».

К документации каждого из инженерных проектов имеют доступ конкретные сотрудники предприятия, которые в рамках соответствующих проектов имеют различные уровни допуска к информации. Для удобства работы всех пользователей АС в ее состав включена база данных, содержащая нормативно-правовые документы, требования ЕСПД, технические справочники. Администратор следит за состоянием базы данных, своевременно обновляет ее. Руководитель предприятия издает приказы и указания и размещает их в электронном виде в соответствующем каталоге. Сотрудники предприятия могут беспрепятственно знакомиться с содержимым базы данных и распоряжениями руководителя предприятия, копировать необходимую им информацию, но вносить изменения в эти каталоги они не имеют право.

Структура документации ООО “AU_Team”:



Пусть к документации проекта “Горизонт_2030” имеют доступ инженеры Громов В.В., Кузнецов М.Р. и Рябов И.В., имеющие уровни допуска к секретной, ДСП и несекретной информации соответственно.

Уровень допуска	Сотрудники
Несекретно	Громов В.В.
ДСП	Кузнецов М.Р., Лебедев М.Г.
Секретно	Рябов И.В., Петров С.В.

В зависимости от своих функциональных обязанностей сотрудники могут осуществлять различные действия с документами проекта (защищаемыми данными): редактировать, просматривать, удалять, копировать, распечатывать. В общем случае специалисты организации одновременно могут работать над несколькими проектами, но в нашем примере инженеры Громов, Кузнецов и Рябов заняты только проектом «Горизонт_2030» и только к нему имеют доступ. В то же время они выполняют весь необходимый объем работы по данному проекту, поэтому доступ остальных инженеров предприятия к его документации запрещен. Для предварительной проработки проектной документации инженеры могут создавать черновики документов. Черновики создаются в специальном каталоге для индивидуального пользования, они доступны только авторам, администратору ИС и руководителю организации.

Права доступа сотрудников к документации предприятия разрешенного уровня конфиденциальности находят свое отражение в матрице доступа, которая вместе с установленной системой допусков и уровней конфиденциальности информационных ресурсов формализует политику разграничения доступа. Возможный вариант матрицы, приведен в таблице ниже, где буквой «П» обозначен тип доступа полный доступ, буквой «Ч» — только чтение, пробелом — запрет доступа.

Каталог	Громов (инженер)	Кузнецов (инженер)	Рябов (инженер)	Чистяков (админ.)	Ювченко (экономист)	Иванов (директор)
./Экономика/ Канцелярские_товары_НС				П	П	П
./Экономика/ Продажи_ДСП				П	П	П
./Приказы_и_ распоряжения	Ч	Ч	Ч	П	П	П
./База_данных	Ч	Ч	Ч	П	Ч	Ч
./Проекты/ Горизонт-2030/ Графические_документы/ Несекретно	П	П	П	П		П
./Проекты/ Горизонт-2030/ Графические_документы/ ДСП		П	П	П		П
./Проекты/ Горизонт-2030/ Графические_документы/ Секретно			П	П		П
./Проекты/ Горизонт-2030/ Текстовые_ документы/ Несекретно	П	П	П	П		П
./Проекты/ Горизонт-2030/ Текстовые_ документы/ ДСП		П	П	П		П
./Проекты/ Горизонт-2030/ Текстовые_ документы/ Секретно			П	П		П
./Проекты/ Горизонт-2030/ Черновики/ Громов	П			П		П
./Проекты/ Горизонт-2030/ Черновики/ Кузнецов		П		П		П
./Проекты/ Горизонт-2030/ Черновики/ Рябов			П	П		П

1. Основные системные разрешения

Поскольку ОС “Альт Рабочая станция” с самого начала разрабатывалась как многопользовательская система, в ней предусмотрен дискреционный контроль доступа (DAC — Discretionary Access Control). Данная модель позволяет разграничить полномочия пользователей, работающих в системе. В частности, права доступа позволяют отдельным пользователям иметь «личные» файлы и каталоги. Если пользователь ivanov создал в своём домашнем каталоге файлы, то он является владельцем этих файлов и может определить права доступа к ним для себя и остальных пользователей. Например, полностью закрыть доступ к своим файлам для остальных пользователей или разрешить им читать свои файлы, запретив изменять и исполнять их.

Правильная настройка прав доступа позволяет повысить надежность системы, защитив от изменения или удаления важные системные файлы. Наконец, поскольку внешние устройства с точки зрения Linux также являются объектами файловой системы, механизм прав доступа можно применять и для управления доступом к устройствам.

Любой файл или каталог в ОС “Альт Рабочая станция” имеет пользователя-владельца и группу-владельца.

Права доступа определяются по отношению к 3-м типам действий: чтение, запись и выполнение. Они могут быть предоставлены трем классам пользователей: владельцу файла, группе, которой принадлежит файл, а также всем остальным пользователям, не входящим в эту группу. Право на чтение даёт пользователю возможность читать содержимое файла или, если такой доступ разрешен к каталогам, просматривать содержимое каталога. Право на запись даёт пользователю возможность записывать или изменять файл, а право на запись для каталога — возможность создавать новые файлы или удалять файлы из этого каталога. Наконец, право на исполнение позволяет пользователю запускать файл как программу или сценарий командной оболочки. Владение правами на исполнение для каталога позволяет перейти (командой cd) в этот каталог.

Таблица 1 — Базовые права доступа

Разрешение	Влияние на файлы	Влияние на каталоги
r (чтение)	Разрешено читать содержимое файлов	Разрешено отображать содержимое каталога (имена файлов)
w (запись)	Разрешено изменять содержимое файлов	Разрешено создавать и удалять любые файлы в каталоге
x (выполнение)	Разрешено запускать файлы как программы	Разрешено переходить в каталог

Чтобы получить информацию о правах доступа, используйте следующую команду:

```
$ ls -l
```

При этом будет выведена подробная информация о файлах и каталогах, в которой будут, среди прочего, отражены права доступа.

Зайдем под учетной записью суперпользователя:

```
$ su -
```

Создадим учетную запись *testuser* с паролей *P@ssw0rd*:

```
# useradd testuser
# passwd testuser
```

Зайдем под учетной записью *testuser* и создадим в домашнем каталоге файл *file.txt* с текстом “Проверка прав доступа”:

```
# su - testuser
$ echo “Проверка прав доступа” > file.txt
```

Также создадим каталог *demodir*:

```
# mkdir demodir
```

Теперь выведем информацию о правах доступа:

```
$ ls -l
```

```
testuser@alt ~ $ ls -l
итого 8
drwxr-xr-x 2 testuser testuser 4096          demodir
-rw-r--r-- 1 testuser testuser   41          file.txt
```

Первый символ указывает на тип файла. Это может быть обычный файл, каталог (d), символическая ссылка (l) или файлы других специальных типов. Следующие девять символов представляют права доступа к файлам, три тройки по три символа в каждой. Первая тройка показывает разрешения владельца, вторая — групповые разрешения, а последняя тройка показывает разрешения всех остальных. Если указан набор rwx, то предоставляются все три разрешения. Если буква заменена на -, то значит разрешение не предоставляется. Первое имя указывает на владельца файла, а второе — на владельца-группу.

В примере владелец файла *testuser* имеет права на чтение и запись, а группа *testuser* и другие пользователи — только на чтение. Владелец каталога *testuser* имеет все права, а группа *testuser* и другие пользователи имеют права на чтение и выполнение.

Выполним команду ls с опцией -d, чтобы отобразить подробную информацию о каталоге, но не о его содержимом:

```
$ ls -ld
```

```
testuser@alt ~ $ ls -ld
drwx------ 9 testuser testuser 4096          .
```

В примере владелец каталога *testuser* имеет все права на каталог, а у группы *testuser* и остальных пользователей разрешений нет.

Проверим механизм разграничения прав доступа к файлам и каталогам. Для этого, выйдем из учетной записи *testuser*, зайдем под учетной записью *user* и с помощью команды cd перейдём в каталог /home/testuser:

```
$ exit
# su - user
$ cd /home/testuser
```

```
testuser@alt ~ $ exit
выход
alt ~ # su - user
user@alt ~ $ cd /home/testuser/
-bash: cd: /home/testuser/: Отказано в доступе
```

Поскольку пользователь *user* принадлежит к остальным пользователям, доступ к каталогу /home/testuser для него ограничен в соответствии с установленными правами доступа.

Для изменения разрешений из командной строки используется команда **chmod**, которая означает “change mode” (смена режима).

Команда **chmod** принимает инструкцию разрешения, за которой следует список файлов или каталогов, которые необходимо изменить. Инструкция разрешения может быть отправлена в символьном или числовом виде.

Синтаксис команды **chmod** (символьный метод):

$ **chmod** [options] WhoWhatWhich file | directory

где Who: u — user, g — group, o — other, a — all;

What: + добавление, - удаление, = точная установка;

Which: r — чтение, w — запись, x — выполнение.

Для символьного метода не обязательно задавать полный набор разрешений.

Команда **chmod** поддерживает опцию -R для рекурсивной установки разрешений файлов во всем дереве каталогов.

Выйдем из учетной записи *user* и зайдем в учетную запись *testuser*:

```
$ exit
# su - testuser
```

Удалим для группы *testuser* и других пользователей разрешения на чтение и выполнение в каталоге *demodir*:

```
$ chmod go-rx demodir
```

Добавим для всех разрешение на выполнение файла file.txt:

```
$ chmod a+x file.txt
```

ИЛИ

```
$ chmod +x file.txt
```

Проверяем изменения:

```
$ ls -l
```

```
testuser@alt ~ $ ls -l
итого 8
drwx------ 2 testuser testuser 4096 demodir
-rwxr-xr-x 1 testuser testuser   41 file.txt
```

Синтаксис команды **chmod** (числовой метод):

$ **chmod** [options] ??? file | directory

В числовом методе разрешения представлены восьмеричным числом из 3-х разрядов (или 4-х в случае настройки дополнительных разрешений). Один такой разряд может представлять любое значение от 0 до 7. Также в трехразрядном восьмеричном (числовом) каждый разряд обозначает один уровень доступа (слева направо): пользователь, группа и остальные.

Разряд **?** рассчитывается путем сложения цифр каждого разрешения, которое необходимо добавить: 4 — чтение, 2 — запись, 1 — выполнение.

В качестве примера разберем разрешение 744:

Владелец: rwx = 4+2+1 = 7

Группа: r-- = 4+0+0 = 4

Прочие: r-- = 4+0+0 = 4
В результате получается трехзначное значение 744.

Установим для пользователя *testuser* разрешение на чтение и запись файла *file.txt*, для группы *testuser* — только на чтение, а для остальных пользователей не предоставим разрешений:

$ chmod 640 file.txt

Проверяем изменения:

$ ls -l file.txt

```
testuser@alt ~ $ ls -l file.txt
-rw-r----- 1 testuser testuser 41                file.txt
```

Каждые новые файлы принадлежат создавшему его пользователю. Также по умолчанию новые файлы принадлежат к основной группе пользователя, который создал файл. В ОС “Альт Рабочая станция” основная группа пользователя обычно является частной группой которая состоит только из одного этого пользователя. Чтобы предоставить доступ к файлу на основе участия в группе, может потребоваться изменить группу, являющуюся владельцем файла.

Только пользователь root может изменить владельца файла! Однако назначить группу-владельца может как пользователь root, так и владелец файла. Пользователь root может назначить группу владельцем файла, только если они входят в эту группу.

Изменить владельца файла можно с помощью команды **chown**, которая означает “change owner” (смена владельца).

Зайдем под учетной записью суперпользователя и изменим владельца файла *file.txt* на *user*. Воспользуемся опцией -c для подробного вывода всех выполняемых изменений:

cd /home/testuser
chown -c user file.txt

```
alt ~ # cd /home/testuser/
alt testuser # chown -c user file.txt
изменён владелец 'file.txt' с testuser на user
```

Проверим изменения:

ls -l file.txt

```
alt testuser # ls -l file.txt
-rw-r----- 1 user testuser 41                file.txt
```

Теперь изменим группу-владельца на root для файла *file.txt*:

chown -c .root file.txt

```
alt testuser # chown -c .root file.txt
изменён владелец 'file.txt' с user:testuser на :root
```

Проверим изменения:

ls -l file.txt

```
alt testuser # ls -l file.txt
-rw-r----- 1 user root 41                file.txt
```

Также для изменения группы-владельца можно использовать команду chgrp.

Синтаксис команды chgrp:

$ **chgrp** [options] group file | directory

Для выполнения команды пользователю необходимо иметь соответствующие права на изменение группы файла.

Вернем владельца и владельца-группу одной командой для файла *file.txt*:

```
# chown -c testuser:testuser file.txt
```

```
alt testuser # chown -c testuser:testuser file.txt
изменён владелец 'file.txt' с user:root на testuser:testuser
```

Проверим изменения:

```
# ls -l file.txt
```

```
alt testuser # ls -l file.txt
-rw-r----- 1 testuser testuser 41                file.txt
```

Команду **chown** можно использовать с опцией -R для рекурсивного изменения всего дерева каталогов.

Если необходимо помимо основных типов разрешений задать дополнительные функции, то можно воспользоваться специальными разрешениями: SUID, SGID, Sticky Bit.

Таблица 2 — Расширенные права доступа

Специальное разрешение	Влияние на файлы	Влияние на каталоги
u+s (SUID)	Файл выполняется от имени пользователя, которые владеет файлом, а не пользователя, который вызвал его выполнение	Не влияет
g+s (SGID)	Файл выполняет от имени группы, которая владеет файлом	Для новых файлов, созданных в каталоге, в качестве группы-владельца задается группа-владелец каталога
o+t (Sticky Bit)	Не влияет	Пользователи с разрешением на запись в каталог могут удалять только те файлы, которыми они владеют. Они не могут удалять или принудительно сохранять данные в файлы, принадлежащие другим пользователям.

Разрешение SUID позволяет пользователю запускать исполняемый файл с правами владельца этого файла. Другими словами, использование этого бита позволяет нам поднять привилегии пользователя в случае, если это необходимо.

Классический пример использования этого разрешения — команда sudo:

```
# ls -l /usr/bin/sudo
```

```
alt ~ # ls -l /usr/bin/sudo
-rws--x--- 1 root wheel 186928                /usr/bin/sudo
```

На месте, где обычно установлен классический бит x (на исполнение), выставлен специальный бит s.

Добавим разрешение SUID для файла *file.txt*:

```
# chmod u+s file.txt
```

Проверяем изменения:

```
# ls -l file.txt
```

```
alt testuser # ls -l file.txt
-rwSr----- 1 testuser testuser 41             file.txt
```

Обратим внимание, что вместо ожидаемой буквы s, видим заглавную S. Это случается, если SUID установлен, но сам владелец файла не имеет прав на его выполнение. Добавим это разрешение:

```
# chmod u+x file.txt
```

Проверяем изменения:

```
# ls -l file.txt
```

```
alt testuser # ls -l file.txt
-rwsr----- 1 testuser testuser 41             file.txt
```

Принцип работы SGID очень похож на SUID с отличием, что файл будет запускаться пользователем от имени группы, которая владеет файлом. Иллюстрирует работу этого бита команда crontab:

```
# ls -l /usr/bin/crontab
```

```
alt ~ # ls -l /usr/bin/crontab
-rwx--s--x 1 root crontab 51728               /usr/bin/crontab
```

Добавим разрешение SGID для файла *file.txt*:

```
# chmod g+s file.txt
```

Не забываем выдавать права на исполнение группе-владельцу файла:

```
# chmod g+x file.txt
```

Проверяем изменения:

```
# ls -l file.txt
```

```
alt testuser # ls -l file.txt
-rwsr-s--- 1 testuser testuser 41             file.txt
```

Последнее специальное разрешение — Sticky Bit. В случае, если этот бит установлен для папки, то файлы в этой папке могут быть удалены только их владельцем. Примером использования этого разрешения — системная папка */tmp*. Папка */tmp* разрешена на запись любому пользователю, но удалять файлы в ней могут только пользователи, являющиеся владельцами этих файлов.

```
# ls -ld /tmp
```

```
alt ~ # ls -ld /tmp
drwxrwxrwt 16 root root 380                   /tmp
```

Символ t указывает, что на папку установлено разрешение Sticky Bit.

Добавим разрешение Sticky Bit для каталога *demodir*:

chmod o+t demodir
Не забываем выдавать права на исполнение группе-владельцу файла:
chmod o+x demodir

Проверяем изменения:
ls -ld demodir

```
alt testuser # ls -ld demodir/
drwx-----t 2 testuser testuser 4096          demodir/
```

Удаляем установленные разрешение также с помощью команды **chmod**:
chmod u-s file.txt
chmod g-s file.txt
chmod -t demodir

Проверяем изменения:
ls -l

```
alt testuser # ls -l
итого 8
drwx-----x 2 testuser testuser 4096          demodir
-rwxr-x--- 1 testuser testuser   41          file.txt
```

Когда Вы создаете новый файл или каталог, ему назначаются начальные разрешения. На эти начальные разрешения влияют два фактора:
1. Создаете Вы обычный файл или каталог.
2. Текущая пользовательская маска.

Пользовательская маска — восьмеричная битовая маска, которая используется для очистки разрешений новых файлов и каталогов, создаваемых процессом. Если в маске установлен какой-либо бит, то соответствующее разрешение в новых файлах очищается. Например, маска 0002 очищает бит записи для остальных пользователей. Начальные нули показывают, что специальные разрешения пользователя и группы не удаляются. Маска 0077 очищает все разрешения группы и остальных пользователей для новых файлов.

Для установки маски прав доступа по умолчанию для новых файлов и каталогов используется команда **umask**.

Команда **umask** без аргументов отображает текущее значение пользовательской маски командной оболочки.

Зайдем в учетную запись *testuser* и выполним команду **umask** без аргументов:
# su - testuser
$ umask

```
testuser@alt ~ $ umask
0022
```

Также создадим папку *demodir2* и файл *file2.txt*:
$ mkdir demodir2
$ touch file2.txt

Посмотрим права доступа для каталога *demodir2* и файла *file2.txt*:
$ ls -ld demodir2

$ ls -l file2.txt

```
testuser@alt ~ $ ls -ld demodir2
drwxr-xr-x 2 testuser testuser 4096          demodir2
testuser@alt ~ $ ls -l file2.txt
-rw-r--r-- 1 testuser testuser 0          file2.txt
```

Для новых каталогов umask отнимает 0022, что дает права доступа rwxr-xr-x, для новых файлов — rw-r--r--, т.к. файлы по умолчанию не получают права на выполнение.

Значение маски по умолчанию для пользователей задается сценариями запуска командной оболочки. По умолчанию для всех пользователей в ОС “Альт Рабочая станция” пользовательская маска устанавливается в файле */etc/profile*.

$ cat /etc/profile

Пользователи могут переопределять системные значение по умолчанию в файлах *.bash_profile* и *.bashrc* в своих домашних каталогах.

Для изменения маски текущей командной оболочки используется команда **umask** с одним числовым аргументом. Числовой аргумент должен быть восьмеричным значением, соответствующим новой маске. Можно опустить ведущие нули в маске.

Временно зададим маску 0027:

$ umask 027

Проверим установку маски:

$ umask

```
testuser@alt ~ $ umask
0027
```

Теперь создадим каталог demodir3 и файл file3.txt:

$ mkdir demodir3

$ touch file3.txt

Посмотрим права доступа для каталога *demodir3* и файла *file3.txt*:

$ ls -ld demodir3

$ ls -l file3.txt

```
testuser@alt ~ $ ls -ld demodir3
drwxr-x--- 2 testuser testuser 4096          demodir3
testuser@alt ~ $ ls -l file3.txt
-rw-r----- 1 testuser testuser 0          file3.txt
```

Для новых файлов владелец получает разрешения на чтение и запись, а группа-владелец получает разрешение только на чтение. Остальные пользователи не получают никаких разрешений.

Для новых каталогов владелец получает все разрешения, группа-владелец получает разрешение на чтение и выполнение, а остальные пользователи не получают никаких разрешений.

2. Расширенные атрибуты файлов

Атрибуты файлов — метаданные, которые определяют различные свойства и поведение файлов и каталогов в файловой системе. Другими словами, атрибуты управляют тем, как файловая система обращается с файлами и каталогами, т.е. с объектами файловой системы. Например, атрибуты могут помечать файлы как неизменяемые или доступные только для добавления.

Расширенные атрибуты предоставляют больше информации о файле, чем стандартные атрибуты. Для отображения установленных расширенных атрибутов файлов и каталогов используется команда **lsattr**.

Синтаксис команды **lsattr**:

$ **lsattr** [options] file | directory

Чтобы отобразить атрибуты файлов в текущем каталоге, выполним команду lsattr без аргументов:

$ lsattr

```
testuser@alt ~ $ lsattr
--------------e------- ./file.txt
--------------e------- ./file3.txt
--------------e------- ./demodir2
--------------e------- ./file2.txt
--------------e------- ./demodir
--------------e------- ./demodir3
```

Чаще всего команда **lsattr** принимает имена файлов и каталогов для проверки в качестве аргументов.

Отобразим атрибуты файла *file.txt*:

$ lsattr file.txt

```
testuser@alt ~ $ lsattr file.txt
--------------e------- file.txt
```

В общем, выходные данные команды **lsattr** состоят из 2-х столбцов: атрибуты, имя файла или каталога. В столбце “attributes” показан ряд букв, представляющих различные параметры и свойства, которые установлены или отключены для каждого файла или каталога.

Атрибут “e” указывает на то, что файл *file.txt* использует непрерывный диапазон блоков для сопоставления блоков хранения.

Основные опции команды lsattr:

1. Опцию -R рекурсивно перечисляет атрибуты всех файлов в каталоге.
2. Опция -a перечисляет только файлов с установленными атрибутами.

Выполним команду **lsattr** с опцией -a:

$ lsattr -a

```
testuser@alt ~ $ lsattr -a
--------------e------- ./.
--------------e------- ./.bash_logout
--------------e------- ./.bash_history
--------------e------- ./.config
--------------e------- ./.local
--------------e------- ./file.txt
--------------e------- ./.xauthECzaw8
--------------e------- ./.xprofile
--------------e------- ./..
--------------e------- ./file3.txt
--------------e------- ./demodir2
--------------e------- ./file2.txt
--------------e------- ./demodir
--------------e------- ./.xsession.d
--------------e------- ./.lpoptions
--------------e------- ./.bashrc
--------------e------- ./.ssh
--------------e------- ./.rpmmacros
--------------e------- ./.mutt
--------------e------- ./.cache
--------------e------- ./.bash_profile
--------------e------- ./demodir3
--------------e------- ./.gtkrc-2.0
```

3. Опция -d выводит список только каталогов, а не их содержимого.

Выполним команду **lsattr** с опцией -d:

$ lsattr -d

```
testuser@alt ~ $ lsattr -d
--------------e------- .
```

Разберем функции основных атрибутов:

Атрибут “a”: Файл может быть открыт только в режиме добавления для записи.

Атрибут “i”: Файл не может быть изменен, удален или переименован.

Атрибут “s”: Ядро надежно стирает файл, когда его удаляют, перезаписывая его блоки данных нулем.

Атрибут “u”: Файл не может быть удален даже суперпользователем.

Атрибут “A”: Время доступа к файлу не обновляется при его чтении.

Атрибут “c”: Файл автоматически сжимается ядром при записи на диск.

Атрибут “D”: Любые операции записи в каталог немедленно синхронизируются с диском.

Атрибут “S”: Любые операции записи в файл немедленно синхронизируются с диском.

Атрибут “d”: Файл не является кандидатом для резервного копирования.

Атрибут “j”: Все операции записи файла сначала записываются во внешний журнал, а затем на диск.

Для изменения расширенных атрибутов файлов и каталогов используют команду **chattr**.

Синтаксис команды **chattr**:

$ **chattr** [options] WhatWhich file | directory

где What: + добавление, - удаление, = точная установка;

Which — атрибуты.

Некоторые атрибуты требуют выполнение команды **chattr** от имени суперпользователя.

Попробуем установить атрибут “i” из-под учетной записи testuser:

$ chattr +i file.txt

```
testuser@alt ~ $ chattr +i file.txt
chattr: Операция не позволена while setting flags on file.txt
```

Команда **chattr** возвращается с ошибкой.

Зайдем под учетной записью суперпользователя и добавим атрибут “i” в файл *file.txt*:

$ exit

cd /home/testuser

chattr +i file.txt

Проверим изменения:

lsattr file.txt

```
alt testuser # lsattr file.txt
----i---------e------- file.txt
```

Теперь попробуем удалить файл file.txt:

rm file.txt

```
alt testuser # rm file.txt
rm: удалить обычный файл 'file.txt'? y
rm: невозможно удалить 'file.txt': Операция не позволена
```

Когда файл *file.txt* имеет атрибут “i” (immutable), то даже после согласия на удаление, команда rm не сможет выполнить эту операцию.

3. Использование ACL

Для реализации сложных структур прав доступа используются расширенные права — ACL (Access Control List — списки контроля доступа), которые дают большую гибкость, чем стандартный набор полномочий.

Поддержка ACL на файловых системах ext4 и XFS включена по умолчанию. При необходимости можно отключить поддержку списков доступа, указав для выбранной файловой системы опцию “noacl” в файле */etc/fstab*.

ACL предоставляет дополнительные функциональные возможности ОС, **НО** у нее есть один недостаток — не все утилиты ее поддерживают. Следовательно, можно потерять настройки ACL при копировании или перемещении файлов, а ПО для резервного копирования может не выполнить резервное копирование настроек ACL.

Для установки и настройки списка доступа используется команда **setfacl**.

Синтаксис команды **setfacl**:

$ **setfacl** [options] acl_spec file | directory

где acl_spec — спецификация ACL, указывающая, какие права доступа применять.

Основные правила команды **setfacl**:

u:username:permissions — установка разрешений для пользователя;

g:groupname:permissions — установка разрешений для группы;

o:permissions — установка разрешений для всех остальных пользователей;

m:permissions — установки маски разрешений.

Основные опции команды **setfacl**:

Опция -m изменяет/добавляет правила ACL.

Опция -x удаляет правило ACL.

Опция -b удаляет ВСЕ правила ACL.

Опция -R рекурсивно применяет операции ко всем подкаталогам и файлам.

Опция -n позволяет применять изменения без изменения стандартных прав доступа, т.е. игнорировать маску.

Зайдем в учетную запись пользователя *testuser*.

Установим для пользователя *user* право на чтение и на запись файла *file2.txt*:

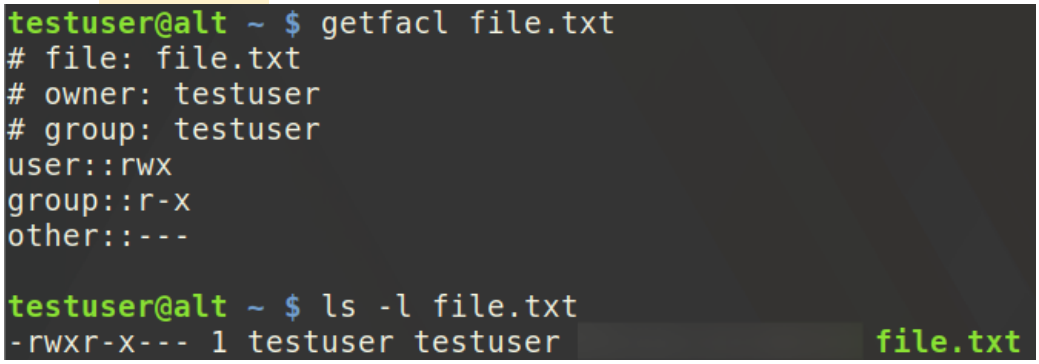
$ setfacl -m u:user:rw file2.txt

Чтобы увидеть текущие настройки ACL, используется команда **getfacl**.

Посмотрим установленные настройки на файле без ACL и сравним вывод с выводом команды *ls -l*:

$ getfacl file.txt

$ ls -l file.txt



Права доступа отображаются по-разному, но значения соответствующих полей одинаковы.

Теперь попробуем посмотреть установленные настройки на файле с установленным правилами ACL и сравним вывод с командой *ls* -l:

$ getfacl file2.txt

$ ls -l file2.txt

```
testuser@alt ~ $ getfacl file2.txt
# file: file2.txt
# owner: testuser
# group: testuser
user::rw-
user:user:rw-
group::r--
mask::rw-
other::r--

testuser@alt ~ $ ls -l file2.txt
-rw-rw-r--+ 1 testuser testuser 0 авг 14 06:07 file2.txt
```

Если у файла или каталога установлены ACL, то в выводе команды *ls -l* будет отображаться знак “+”.

Команда *ls* не выводит имя второго владельца файла и его права доступа к файлу, хотя второй владелец присутствует, как показывает команда **getfacl**.

Попробуем сделать копию файла *file2.txt*:

$ cp file2.txt file2cp.txt

Посмотрим права доступа файла *file2cp.txt*:

$ ls -l file2cp.txt

```
testuser@alt ~ $ ls -l file2cp.txt
-rw-r--r-- 1 testuser testuser 0            file2cp.txt
```

Настройки ACL НЕ копируются.

При работу с ACL нужно быть внимательнее, т.к. не все программы могут использовать ACL.

Заметим, что для копирования файла с сохранением ACL необходимо использовать опцию -p для команды cp:

$ cp -p file2.txt file2cp2.txt

Теперь посмотрим права доступа файла *file2cp2.txt*:

$ ls -l file2cp2.txt

```
testuser@alt ~ $ ls -l file2cp2.txt
-rw-rw-r--+ 1 testuser testuser 0            file2cp2.txt
```

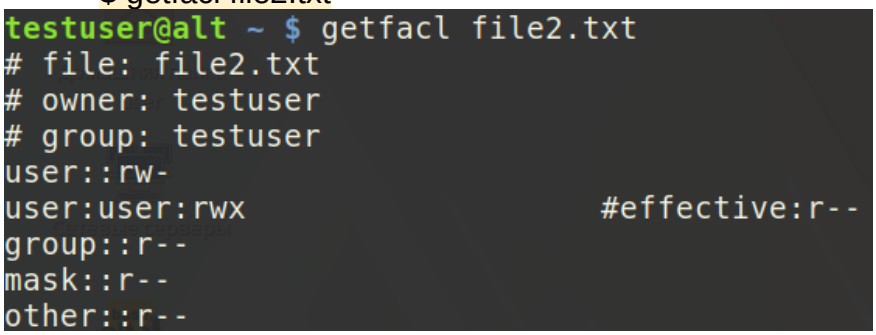
Чтобы задать максимальные права доступа для всех пользователей, необходимо использовать маску ACL. Например, маска r-- показывает, что пользователи и группы не могут иметь больших прав, чем просто чтение, даже если им назначены права доступа на чтение и запись.

Установим максимальные права доступа для пользователя *user* на файл *file2.txt*:

$ setfacl -m u:user:rwx file2.txt

Затем зададим маску ACL:
$ setfacl -m m:r-- file2.txt

Посмотрим права доступа на файл *file2.txt*:
$ getfacl file2.txt



Права на файл устанавливаются в соответствии с выполняемой командой, но маска, установленная как r--, лишает разрешения на запись и выполнение всех владельцев, кроме основного. Это подтверждает появившаяся запись в выводе команды **getfacl**.

Для отмены маски ее следует установить в значение rwx.