ЛАБОРАТОРНАЯ РАБОТА №8 УПРАВЛЕНИЕ ДОСТУПОМ В ФАЙЛОВОЙ СИСТЕМЕ EXT3FS

Цель работы: практическое знакомство со средствами обеспечения безопасности в ОС Linux и методами управления доступом к данным в файловой системе ОС Linux ext3fs

1 КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

1.1 Права доступа и группы пользователей

В ОС Unix и Linux каждому файлу файловой системы ext3fs (или ext4fs) соответствует *набор прав доступа*, представленный в виде 9-ти битов режима. Он определяет, какие пользователи имеют право читать файл, записывать в него данные или выполнять его. Вместе с другими тремя битами, влияющими на запуск исполняемых файлов, этот набор образует *код режима доступа к файлу*. Двенадцать битов режима хранятся в 16-битовом поле индексного дескриптора вместе с 4-мя дополнительными битами, определяющими тип файла. Последние 4 бита устанавливаются при создании файлов и не подлежат изменению. Биты режима (далее права) могут изменяться либо владельцем файла, либо суперпользователем с помощью команды **chmod**.

В многопользовательской ОС Linux с целью защиты информации пользователей и обеспечения безопасности самой ОС используются группы пользователей со следующими свойствами:

- каждый пользователь входит в минимум одну группу. Группа, присваиваемая пользователю при его создании, называется основной. Все остальные группы в которые будет включен пользователь, будут являться дополнительными.
- группа пользователей может содержать некоторое количество пользователей, но не может содержать или включаться в другие группы.
- группа может быть пустой, т.е. не содержать в себе ни одного пользователя.

Чтобы добавить пользователя в ту или иную группу, используя командную строку, следует отредактировать файл **/etc/group**. Структура записи файла:

имя_группы: пароль: GID: список_пользователей

Например:

syslog:x:103:

klog:x:104:

scan:x:105:hpl,alexset

nvram:x:106:

fuse:x:107:alexset

В приведенном примере в группу **scan** входят пользователи **hpl** и **alexset**. Чтобы добавить в эту группу новых пользователей, следует перечислить их символьные имена через запятую. В графическом режиме для управления учетными записями пользователей используется утилита управления пользователями и группами, запускаемая через System/Administration/Users and Groups.

Существует три способа управления доступом к файлу или каталогу. Было определено, что каждый файл должен иметь владельца (*owner*), группового владельца (*group owner*), а также может потребоваться доступ для всех остальных пользователей (*everyone*). Эти названия обычно приводятся как **пользователь**/**группа**/**остальные** (user/group/others) или коротко **ugo**. Реализация управления доступом к файлам и каталогам в ОС Linux разрешает или запрещает доступ с помощью трех бит: бита чтения (**Read**), бита записи (**Write**), бита выполнения (**eXecute**). Они представляются следующим образом:

flag user group other

rwx rwx rwx

Флаг типа (flag) может быть одним из следующих (табл. 1).:

Таблица 1.

Типы Флаг

Флаг	**Описание**
-	Отсутствие флага
l	Символическая ссылка (symbolic link)
d	Директория (directory)
b	Блочное устройство (block device)
c	Символьное устройство (character device)
p	Канал, устройство fifo (fifo device)
s	Unix сокет (unix domain socket)

Стандартные права

Посмотреть права доступа на объекты можно командой **ls** с ключом **-**

l. Также можно добавить ключ **-a** для того, чтобы были отображены скрытые объекты:

```
/media/Work/applicat > ls -l
итого 1308
-rwx------  1 alex nogroup 638116 2015-06-25 20:42 autcolor.zip
drwxr-xr-x 13 alex nogroup   4096 2015-05-31 14:58 phpBB3
drwx------ 10 alex nogroup   4096 2015-06-25 14:29 phpMyAdm-3.2.0-all-languages
-rwx------  1 alex nogroup 677334 2015-06-25 20:42 pro_ubuntu.zip
drwxr-xr-x  2 alex nogroup   4096 2015-06-25 14:29 безымянная папка
```

В листинге используются буквенные обозначения прав (т.н. маска режима доступа), но часто используются также цифровые обозначения в восьмеричном представлении, показанные в табл. 2:

Таблица 2.

Обозначение

OCT	**BIN**	**Mask**	**Комментарий**
0	000	- - -	отсутствие прав
1	001	- - x	права на выполнение
2	010	- w -	права на запись
3	011	- w x	права на запись и выполнение
4	100	r - -	права на чтение
5	101	r - x	права на чтение и выполнение
6	110	r w -	права на чтение и запись
7	111	r w x	полные права

Для назначения обычных прав используются три восьмеричных цифры (9 битов).

Первая цифра определяет права для владельца файла, вторая - права для основной группы пользователя, третья - для всех остальных пользователей. Так, например, чтобы задать права на файл всем, нужно установить права **777** (**rwxrwxrwx**). Существуют также специальные биты, такие как **SUID**, **SGID** и **Sticky**-бит. Они влияют на запуск выполняемого файла. При их применении необходимо использовать не три восьмеричных цифры, а 4. Зачастую в технической литературе права обозначаются именно четырьмя цифрами, например **0744.**

Пример 1:

```
-rwx------  1 alexserv nogroup 677334 2016-06-25 20:42 pro_ubuntu.zip
drwxr-xr-x  2 alexserv nogroup   4096 2015-06-25 14:29 безымянная папка
```

Для первой строки примера 1:

- Первый символ (флаг) пустой: - для файлов.

- Следующие три символа (**rwx**) обозначают права для владельца файла, в данном случае полные права для пользователя **alexserv**. По умолчанию, при создании объекта (файла или каталога) устанавливаются полные права для его владельца.
- Следующие три (- - -) - определяют права для группы **nogroup**, в нашем примере для всех пользователей группы **nogroup** доступ запрещен.
- последние три символа (- - -) определяют права для всех остальных пользователей, в нашем случае доступ запрещен.

Восьмеричное обозначение прав для файла **pro_ubuntu.zip**: **0700**.
Для второй строки (это каталог, о чем свидетельствует флаг «d»):

- Для владельца каталога **alexserv** - полные права (rwx).
- Для группы **nogroup** - права только на листинг каталога (r-x).
- Для пользователя «все остальные» - права только на листинг каталога (r-x).

Восьмеричное обозначение в этом примере: **0755.**–Для того, чтобы пользователи могли просматривать каталог, необходимы права на чтение и выполнение каталога, т.е. минимальные права на каталог 0555 (r-xr-xr-x), прав 0444 (r - - r - - r - -) будет недостаточно для входа в каталог

1.2 Установка прав - команда chmod

Права устанавливаются командой **chmod**. По умолчанию использовать ее может только **root**. Команда **chmod** поддерживает установку прав как в восьмеричном представлении, так и в символьном (маска режима доступа).
Синтаксис команды:
chmod <опции> <права> <объект или регулярное выражение>
Опции:

Из самых полезных и часто используемых опций можно выделить одну:
-R - рекурсивное назначение прав. Т.е. назначить права всем объектам, руководствуясь регулярным выражением.

Пример 2:
chmod -R 755 * - Назначение прав всем объектам текущего каталога, включая подкаталоги.
chmod -R 700 z* - Назначить полные права для владельца и исключить права для группы и всех остальных для всех объектов, которые начинаются именоваться на **z**, находящиеся в текущем каталоге и его подкаталогах. Применение прав к объектам в подкаталогах произойдет только в том случае, если сам подкаталог начинает именоваться на **z**. Т.е. рекурсия будет применяться только и только к тем объектам, которые

удовлетворяют регулярному выражению

1.3 Запись прав

Права можно записывать как в восьмеричном представлении так и в символьном. В восьмеричном представлении, для стандартных прав, указываются 3 восьмеричные цифры (первая для владельца, вторая для группы, третья для всех остальных). См. таблицу.

Пример 2:

- **chmod 744 mydat.txt** - установит права для файла **mydat.txt** - (r w x r - - r - -);
- **chmod -R 775 doc** - установит права на каталог **doc** и на все объекты, что внутри этого каталога, включая содержимое подкаталогов (r w x r w x r - x);
- **chmod 700 *** - установит права только для владельца на все файлы и каталоги в текущем каталоге, включая подкаталоги и их объекты (rwx - - - - - -).

Другой способ назначения прав - это использование маски режима доступа (символьное представление). Помимо прав также задается пользователь или группа пользователей, которым эти права предоставляются:

- **u** - владельцу объекта;
- **g** - группе объекта;
- **o** - пользователю «все остальные»;
- **a** - все вышеперечисленное.

Для назначения прав используются три знака: минус, плюс или равно:

- **-** - убрать указанные права с объекта;
- **+** - добавить указанные права к существующим правам объекта;
- **=** - заменить права объекта на указанные.

Пример:

- **chmod g+w mydat.txt** - добавить пользователям группы файла mydat.txt права на запись в этот файл;
- **chmod a=rwx fdoc.doc** - заменить существующие права на файле fdoc.doc на полные права всем;
- **chmod o-w test.cgi** - убрать права на запись для пользователя «Все остальные».
- **chmod ug=rw spisok.doc** - выставить права на чтение и запись файлу spisok.doc для владельца и группы. Обратите внимание, что если у пользователя «все остальные» были какие-либо права, они сохранятся в неизменном виде.

Биты SUID, SGID и Sticky

Linux использует не символьные имена владельцев и групп, а их

идентификаторы (**UID** - для пользователей и **GID** для групп). Эти идентификаторы хранятся в файлах **/etc/passwd** и **/etc/group** соответственно. Символьные эквиваленты идентификаторов используются только для удобства, например, при использовании команды **ls**, идентификаторы заменяются соответствующими символьными обозначениями.

Зачастую, при создании пользователя, если не указано иное, идентификатор группы пользователя равен идентификатору пользователя.

Что касается процессов, то с ними связаны не два идентификатора, а четыре: реальный и эффективный пользовательский (UID), а также реальный и эффективный групповой (GID). Реальные номера применяются для учета использования системных ресурсов, а эффективные для определения прав доступа к процессам. Как правило, реальные и эффективные идентификаторы совпадают. Владелец процесса может посылать ему сигналы, а также изменять приоритет.

Процесс не может явно изменить ни одного из своих четырех идентификаторов, но есть ситуации, в которых происходит косвенная установка новых эффективных идентификаторов процесса. Существуют два специальных бита: **SUID** (**Set User ID** - бит смены идентификатора пользователя) и **SGID** (**Set Group ID** - бит смены идентификатора группы). Когда пользователь или процесс запускает исполняемый файл с установленным одним из этих битов, файлу временно назначаются права его (файла) владельца или группы (в зависимости от того, какой бит задан). Таким образом, пользователь может даже запускать файлы от имени суперпользователя.

Восьмеричные значения для **SUID** и **SGID** равны **4000** и **2000, с**имвольные: **u+s** и **g+s.**

Установка битов **SUID** или **SGID** позволит пользователям запускать исполняемые файлы от имени владельца (или группы) запускаемого файла. Например, как говорилось выше, команду **chmod** по умолчанию может запускать только **root**. Если мы установим **SUID** на исполняемый файл **/bin/chmod**, то обычный пользователь сможет использовать эту команду без использования **sudo**, так, что она будет выполняться от имени пользователя **root**. По такому принципу работает, например, команда **passwd**, с помощью которой пользователь может изменить свой пароль.

Если установить **SGID** для каталога, то все файлы, созданные в нем при запуске будут принимать идентификатор группы каталога, а не группы владельца, который создал файл в этом каталоге. Одним словом, если пользователь поместил исполняемый файл в такой каталог и запустил его, процесс запустится от имени владельца (группы) каталога, в котором находится этот файл.

Установить **SUID** и **SGID** можно командой **chmod**:

- **chmod 4755 doc.pl** - устанавливает на файл doc.pl бит SUID и

заменяет обычные права на 755 (rwxr-xr-x).

- **chmod u+s doc.pl** - тоже самое, только обычные права не перезаписываются.
- **chmod 2755 doc.pl** - устанавливает на файл doc.pl бит SGID и заменяет обычные права на 755 (rwxr-xr-x).
- **chmod g+s doc.pl** - тоже самое, только обычные права не перезаписываются.

Догадайтесь, что произойдет после выполнения такой команды:

- **chmod 6755 doc.pl**

Снять установленные биты можно различными способами:

- **chmod g-s doc.pl** - убираем SUID
- **chmod u-s doc.pl** - убираем SGID
- **chmod 0644 doc.pl** - убираем все дополнительные биты и меняем права на 644.

Пример 3. Отображение **SGID** и **SUID**:

/media/Work/test > ls -l
итого 20
drwxr-xr-x 2 root root 4096 2016-06-15 16:18 alex
drwx------ 2 root root 4096 2015-06-15 14:20 qqq
-rwxrwsrwx 1 root root 0 2015-07-24 19:42 qwer

В примере 3 видно, что для файла **qwert** установлен **SGID**, о чем свидетельствует символ «**s**» (-rwxrwsrwx). Символ «**s**» может быть как строчная буква (**s**), так и прописная (**S**). Регистр символа только лишь дает дополнительную информацию об исходных установках, т.е. был ли до установки SGID установлен бит, в данном случае на выполнение (rwxrw**s**rwx). Если **s** строчная, то права на выполнение у группы этого файла были до установки **SGID**. Если **S** прописная, то группа для этого файла ранее не имела прав на выполнение до установки **SGID.**

Еще одно важное усовершенствование касается использования **sticky-бита** в каталогах. В отличии от установки **sticky** на каталог, на файл такой бит устанавливать не имеет смысла. Каталог с установленным **sticky-битом** означает, что удалить файл из этого каталога может только владелец файла или суперпользователь. Другие пользователи лишаются права удалять файлы, даже если имеют права 7 (rwx), хотя писать (создавать) файлы в таких каталогах они могут, при условии что имеют права 7 (rwx) Установить **sticky-бит** в каталоге может только суперпользователь. **Sticky-бит** каталога, в отличие от **sticky-бита** файла, остается в каталоге до тех пор, пока владелец каталога или суперпользователь не удалит каталог явно или не применит к нему **chmod.** Владелец каталога может удалить **sticky-бит**, но не может его установить.

- Восьмеричное значение stiky-бита: **1000**
- Символьное: **+t**

Установить **sticky-бит** на каталог можно используя команду **chmod**:

- **chmod 1755 alex** - с заменой прав;
- **chmod +t alex** - добавление к текущим правам.
  Убрать **sticky-бит** на каталог можно:
- **chmod -t allex**
  Отображение **sticky-бит**:
  /media/Work/test > ls -l
  итого 20
  drwxr-xr-t 2 root root 4096 2016-06-15 16:18 alex
  drwx------ 2 root root 4096 2015-06-15 14:20 qdir
  -rwxr--r-T 1 root root 0 2016-07-24 19:42 qwert
  -rw-r--r-- 1 root root 4099 2015-06-11 14:14 sources.list

Видно, что **sticky-бит** установлен на каталоге **alex**, а также на файле **qdir**, о чем свидетельствует символ (**t**). Символ «**t**» может быть представлен как строчной буквой (**t**), так и прописной (**T**). Строчная буква отображается в том случае, если перед установкой **sticky bit** произвольный пользователь уже имел право на выполнение (**x**), а прописная (**T**) — если такого права у него не было. Конечный результат один и тот же, но регистр символа дает дополнительную информацию об исходных установках.

Итак, использование **sticky** позволяет реализовать каталоги общего пользования, в которые пользователи могут писать файлы, но не могут удалять чужие файлы.

Пример 4. Установка **sticky** на каталог:
/media/Work/test > ls -l
итого 20
drwxrwxrwx 2 root root 4096 2015-07-24 20:54 alex
drwx------ 2 root root 4096 2015-06-15 14:20 qqq
После установки sticky-бита:
/media/Work/test > chmod +t alex
/media/Work/test > ls -l
итого 20
drwxrwxrwt 2 root root 4096 2015-07-24 20:54 alex
drwx------ 2 root root 4096 2015-06-15 14:20 qqq

1.4 Списки управления доступом (ACL-списки)

Возможностей стандартных прав ОС Linux недостаточно для реализации сложных схем доступа. Списки управления доступом (**ACL** – Access Control List списки) предоставляют более тонкую организацию управлению доступом пользователей к указанным каталогам и файлам по сравнению с традиционными правами доступа ОС Linux [1].

При использовании ACL-списков можно указать способы, позволяющие каждому из нескольких пользователей иметь доступ к

каталогу или файлу. Недостаток ACL-списков в снижении производительности системы, поэтому их использование не следует разрешать на файловых системах, хранящих системные файлы, где вполне достаточно традиционных прав доступа Linux. Следует проявлять осторожность при перемещении, копировании или архивировании файлов, поскольку не все утилиты сохраняют ACL-списки. Кроме того, невозможно скопировать ACL-списки в файловые системы, которые их не поддерживают, или в файловую систему, в которой выключена поддержка этих списков.

ACL-список включает в себя набор *правил*. Правило определяет, как указанный пользователь или группа может получить доступ к файлу, с которым связан ACL-список. Есть два вида правил: *правила доступа* (access ACLs) и *правила по умолчанию* (default ACLs).

Правило доступа определяет информацию о доступе для отдельного файла или каталога. ACL-список по умолчанию принадлежит исключительно каталогу, он определяет информацию доступа по умолчанию (ACL-список) для любого имеющегося в каталоге файла, которому не предоставлен явно заданный ACL-список.

1.4.1 Предоставление возможности использования ACL-списоков

Перед тем как получить возможность использования ACL-списков, нужно установить программный пакет acl. Linux поддерживает ACL-списки на файловых системах ext2, ext3 и ext4. Для использования ACL-списков на файловой системе ext нужно смонтировать устройство с данной файловой системой с ключом acl. Сначала необходимо отредактировать файл со списком файловых систем fstab, указать в нем, на каких разделах HDD следует учитывать ACL права, затем перемонтировать устройство, как указано в [1].

Для указания поддержки файловой системой ACL-списков необходимо в строку с описанием выбранной файловой системы добавить параметры grpquota,acl,suid и перезагрузить компьютер.

В данной работе указанные операции не выполняются. Желающие проверить поддержку ACL на своем компьютере могут при необходимости получить дополнительные консультации.

1.4.2 Работа с правами доступа

Утилита getfacl отображает ACL-список указанного файла. При использовании утилиты getfacl для получения информации о файле, который не имеет ACL-списка, она показывает ту же информацию, что и команда ls –l, хотя и в ином формате.

$ ls –l test

-rw-r—r—1 mx mx 9537 Jan 12 23:15 test

$ getfact test

file: test
owner: aa
group: aa
user:: rw-
group:: r—
other:: r—

Первые три строки выведенной информации содержат заголовок. В них указывается имя файла, его владелец и группа, с которой связан файл. В строке, которая начинается со слова user, два двоеточия указывают на то, что в строке показаны права владельца файла. В следущей строке показываются права той группы, с которой связан файл. Со всеми остальными пользователями (other) не могут быть связаны никакие имена.

Чтобы посмотреть, установлены ли ACL-списки на объектах, достаточно воспользоваться командой **ls -l**: Символ "+" в конце списка стандартных прав сообщает о наличии установленных прав ACL:

Утилита **setfacl** с ключом –m позволяет добавить или изменить одно или несколько правил в ACL-списке файла. Списки ACL можно задать:

- **на уровне пользователей** - назначаются ACL конкретным пользователям;
- **на уровне групп** - назначаются ACL конкретным группам;
- **с помощью маски эффективных прав** - ограничение максимальных прав для пользователей и/или групп;
- **для пользователей, не включённых в группу данного файла** - это т.н. пользователь «Все остальные»;

Рассмотрим синтаксис командной строки **setfacl**:

setfacl <опции> <ключ> <список правил> <объект>

- **<опции>** - задает дополнительные опции;
- **<ключ>** - задает режим работы утилиты;
- **<список правил>** - собственно, сами правила доступа к объекту;
- **<объект>** - объект к которому применяется ACL, в большинстве случаев это файл или каталог.

Часто используемые ключи табл. 3:

Таблица 3.

Ключи

Ключ	Описание
--set или --set file*	- Устанавливает новые указанные права ACL, удаляя все существующие. Необходимо, чтобы наравне с задаваемыми правилами ACL были также указаны стандартные права Linux, в противном случае будет давать ошибку;
-m или -M file*	- Модифицирует указанные ACL на объекте. Другие существующие ACL сохраняются.

Ключ	Описание
-x или -X file*	- Удаляет указанные ACL права с объекта. Стандартные права Linux не изменяются.

* - При использовании **-M, - -set, -X** - разрешения будут браться из указанного файла **file**, который должен быть заранее подготовлен в формате вывода ACL разрешений командой **getfacl**. Подготовить файл можно либо вручную, в формате вывода **getfacl**, либо использовать команду **getfacl -R file > file_out**. Где **file** это файл(ы) или каталог(и) с которых нужно снять ACL, а **file_out** - текстовый файл, в который запишутся снятые ACL права.

Часто используемые опции табл. 4:

Таблица 4

Опции

Опция	Описание
-b	-Удаляет все ACL права с объекта, сохраняя основные права;
-k	-Удаляет с объекта ACL по умолчанию. Если таковых на объекте нет, предупреждение об этом выдаваться не будет;
-d	-Устанавливает ACL по умолчанию на объекте.
-restore=file	-Восстанавливает ACL права на объекте из ранее созданного файла с правами.
-R	-Рекрусивное назначение (удаление) прав, то есть с просмотром всех подкаталогов.

Проверить, поддерживает ли тот или иной раздел винчестера ACL, можно попытавшись установить ACL командой **setfacl**:

/root > setfacl -m u:alexser:rw-,g:root:rw- qqq

setfacl: qqq: Operation not supported

Вывод сообщения **Operation not supported** - операция не поддерживается - признак того, что ACL-список на том разделе винчестера, где лежит файл **qqq** не активирован.

Примеры **использования утилиты setfacl.**

Рассмотрим примеры использования назначения, модификации ACL. Выше был приведен вывод команды **getfacl** для файла **test**:

$ getfact test
# file: test
# owner: aa
# group: aa
user:: rw-
group:: r—
other:: r—

Теперь добавим к пользователю этого файла пользователя **ab**:

```
$ setfacl -m u:ab:rwx test
$ getfacl test
# file: test
# owner: aa
# group: aa
user::rw-
user:ab:rwx
user:child:rw-
group::r--
mask::rwx
other::r—
```

1.4.3 Маска эффективных прав

Строка, начинающаяся со слова mask, указывает на маску *эффективных прав*. Эта маска ограничивает эффективные права, выделенные с помощью ACL-списков группам и пользователям. Она не затрагивает права владельца файла или группы, с которой связан файл. Но, поскольку утилита setfacl всегда устанавливает маску эффективных прав на минимальные ограничительные права доступа к файлу, устанавливаемые в ACL-списке, эта маска не оказывает никакого влияния до тех пор, пока она не будет установлена явным образом после установки для файла ACL-списка. Маску можно установить, указав mask вместо ugo и не указывая имя в команде setfacl. Пример:

```
$ setfacl –m mask::r--test
$ getfacl test
# file: test
# owner: aa
# group: aa
user::rw-
user:ab:rwx                    #effective:r--
user:child:rw-
group::r--
mask::rwx
other::r--
```

1.4.4 Установка правил по умолчанию для каталога

Рассмотрим наследование прав объектов. Если не задано иное, то все объекты, создаваемые в каталоге, у которого есть ACL-списки, не наследуют права ACL каталога, в котором они создаются. Но иногда возникают ситуации, когда необходимо, чтобы все объекты, создаваемые внутри каталога, наследовали его ACL права. Эта возможность называются ACL по умолчанию. Для добавления правил по умолчанию для каталога в команде setfacl используется ключ –d (default). Эти правила применяются

ко всем файлам, создаваемым в каталоге, у которых нет явно установленных ACL-списков.

```
$ ls –ld dir
drwxr-xr-x 2 aa aa 4096 2015-01-06 20:05 dir
$ getfacl dir
# file: dir
# owner: aa
# group: aa
user::rwx
group::r-x
other::r-x
```

Добавление правил по умолчанию для каталога dir

```
$ setfacl –d –m g:ab:r-x dir
```

Получение ACL-списка

```
getfacl dir
# file: dir
# owner: aa
# group: aa
user::rwx
group::r-x
other::r-x
default:user::rwx
default:group::r-x
default:group:ab:r-x
default:mask::r-x
default:other::r-x
```

Каждое из правил по умолчанию, которое отображает утилита getfacl, начинается со слова default:. Первые два правила по умолчанию и последнее правило по умолчанию определяют права доступа для владельца файла, для группы, с которой связан этот файл, и для всех остальных пользователей. Эти три правила определяют традиционные права доступа Linux и имеют приоритет над другими ACL- правилами. Третье правило определяет права доступа для группы ab. Далее следует маска эффективных прав.

Следует помнить, что правила по умолчанию относятся к файлам, которые содержатся в каталоге и которым ACL-списки не присвоены явным образом.

Удалить права по умолчанию можно так: **setfacl -k dir**.

1.4.5 Копирование ACL прав с одного объекта на другой.

Принцип копирования:

- Снять ACL с одного объекта
- Записать их на другой

В документации приведен следующий пример:
getfacl file1 | setfacl --set-file=- file2
где **file1** - объект с которого снимаем ACL командой **getfacl**, далее устанавливаем ACL командой **setfacl** на объект **file2**. Обратите внимание на запись **- -set-file=-** в конце указан символ »-», который берет информацию от команды **getfacl** (со стандартного вывода).
Справедлива будет также такая запись:
getfacl file1 | setfacl -M- file2
В этом случае права на **file2** не заменяются как при использовании **- -set**, а добавляются к уже существующим правам ACL.

1.4.6 Копирование прав ACL каталога в права по умолчанию этого же каталога

getfacl --access dir | setfacl -d -M- dir

В этом примере **getfacl** получает все права, которые были установлены на каталог **dir** и устанавливает их на этот же каталог **dir,** делая их правами по умолчанию. Обратите внимание на ключ **- -access** у команды **getfacl**. При вызове **getfacl** без параметров, она отображает все права ACL, включая права по умолчанию. Здесь же ключ **- -access** заставляет **getfacl** показать только права ACL на каталог, а права по умолчанию (если таковые имеются у каталога) - скрыть. Обратный ключ - это ключ **-d**

2 МЕТОДИКА ВЫПОЛНЕНИЯ

1. Ознакомиться с теоретическими сведениями.
2. Создать группу пользователей с именем g<номер_бригады>1 и пользователя с именем a в этой группе, используя режим командной строки.
3. Создать группу пользователей с именем g<номер_бригады>2 и пользователя с именем b в этой группе, используя графический интерфейс пользователя.
4. В домашнем каталоге создать по одному каталогу и файлу на каждого пользователя.
5. Разрешить группе чтение, владельцу - чтение и запись файла. Для каталога группе разрешить чтение и выполнение. Для выполнения задания использовать запись прав в 8 сс и маску прав.
6. На один из созданных каталогов установить sticky-бит.
7. Записать в каталог со sticky-битом по копии файла от каждого пользователя бригады, выполнить удаление записанных файлов (проверка действия sticky-бита).
8. Скопировать один из выполняемых файлов, созданных в работе 5 в один из созданных каталогов и установить ему бит SGID. С помощью

команды ls –l получить результаты установки.

9. Проверить, установлена ли поддержка ACL-списков на компьютере, на котором выполняется лабораторная работа.

10. На компьютере с поддержкой ACL-списков установить для одного из созданных каталогов правила по умолчанию и получить результаты установки с помощью утилиты getfacl.

11. Ответить на контрольные вопросы.

3 ОТЧЕТ О РАБОТЕ

Готовится в письменном виде один на бригаду. Содержание отчета:

1. Ход выполнения заданий 2 - 6 – использованные команды и полученные результаты.
2. Результаты выполнения заданий 7 - 10.

4 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Группы пользователей, назначение, создание и использование.
2. Типы файлов файловой системы ext3fs.
3. Управление доступом к файлам и каталогам в ОС Linux.
4. Команда просмотра прав доступа на объекты.
5. Стандартные права доступа к объектам файловой системы ОС Linux и формы их записи.
6. Установка прав доступа с помощью команды chmod.
7. Назначение битов SUID, SGID.
8. Назначение бита Sticky.
9. Способы установки битов SUID, SGID, Sticky.
10. Необходимость использования ACL-списков.
11. Недостатки ACL-списков.
12. Виды ACL-списков. Содержимое ACL-списков.
13. Подключение ACL-списков.
14. Назначение утилит getfacl и setfacl.
15. Проверка наличия ACL-списка у файла или каталога.
16. Маска эффективных прав – назначение и использование.
17. Установка правил по умолчанию для каталога.
18. Копирование ACL- списков.
19. Создание нового пользователя в режиме командной строки.
20. Создание нового пользователя в графическом режиме. Управление пользователями.