Лабораторная работа №2. Разграничение прав доступа

Основные теоретические сведения

Цель: Изучение механизмов управления доступа к ресурсам, прав доступа. Постижение понятия пользователя и группы. Приобретение практических навыков управления пользователями при помощи консольных утилит. Приобретение навыков работы с правами пользователей и правами на файлы, каталоги при помощи консольных утилит.

Теоретическая часть

У каждого объекта (файла) есть уникальное имя, по которому к нему можно обращаться, и конечный набор операций, которые процессы могут выполнять в отношении этого объекта. Файлу свойственны операции read, write и execute.

Совершенно очевидно, что нужен способ запрещения процессам доступа к тем объектам, к которым у них нет прав доступа. Более того, этот механизм должен также предоставлять возможность при необходимости ограничивать процессы поднабором разрешенных операций. Например, процессу A может быть дано право проводить чтение данных из файла F, но не разрешено вести запись в этот файл.

Права доступа означают разрешение на выполнение той или иной операции (чтение, записи, исполнения).

Когда пользователь входит в систему, его оболочка получает UID и GID (UID – идентификатор пользователя, GID - идентификатор группы), которые содержатся в его записи в файле паролей, и они наследуются всеми его дочерними процессами. Представляя любую комбинацию (UID, GID), можно составить полный список всех объектов (файлов, включая устройства ввода-вывода, которые представлены в виде специальных файлов и т.д.), к которым процесс может обратиться с указанием возможного типа доступа (чтение, запись, исполнение).

Два процесса с одинаковой комбинацией (UID, GID) будут иметь абсолютно одинаковый доступ к одинаковому набору объектов. Процессы с различающимися значениями (UID, GID) будут иметь доступ к разным наборам файлов, хотя, может быть, и со значительным перекрытием этих наборов.

SUID (Set User ID)

Атрибут исполняемого файла, позволяющий запустить его с правами владельца. В операционных системах Linux приложение запускается с правами пользователя, запустившего указанное приложение. Это обеспечивает дополнительную безопасность т.к. процесс с правами пользователя не сможет получить доступ на запись к важным системным файлам, например /etc/passwd, который принадлежит суперпользователю root. Если на исполняемый файл установлен бит suid, то при выполнении эта программа автоматически меняет «эффективный userID» на идентификатор того пользователя, который является владельцем этого файла. То есть, не зависимо от того - кто запускает эту программу, она при выполнении имеет права хозяина этого файла.

SGID (Set Group ID)

Аналогичен SUID, но относиться к группе. При этом, если для каталога установлен бит SGID, то создаваемые в нем объекты будут получать группу владельца каталога, а не пользователя.

Практические примеры

Узнать права на файл/директорию

```
sit@ubuntu:~$ ls -l /bin/ls
-rwxr-xr-x 1 root root 129280 Feb 18  2016 /bin/ls
```

Права доступа состоят из трех троек символов. Первая тройка представляет права владельца файла, вторая представляет права группы файла и третья права всех остальных пользователей.

В нашем случае это :

- «rwx» - Права владельца файла
- «r-x» - Права группы файла
- «r-x» - Права всех остальных на файл.

Символ «r» означает, что чтение (просмотр данных содержащихся в файле) разрешено, «w» означает запись (изменение, а также удаление данных) разрешено и «x» означает исполнение (запуск программы разрешен).

Таким образом, если в целом посмотреть на права мы увидим, что кому угодно разрешено читать содержимое и исполнять этот файл, но только владельцу (root) разрешено как либо модифицировать этот файл. Иными словами, нормальным пользователям разрешено копировать содержимое этого файла, то только root может изменять или удалять его.

Определение текущего пользователя и групп в которых он состоит

Перед тем, как изменять владельца или группу которой принадлежит файл, необходимо уметь определять текущего пользователя и группу к которой он принадлежит. Чтобы узнать под каким пользователем вы работаете, наберите whoami:

```
sit@ubuntu:~$ whoami
sit
```

Для определения в каких группах состоит пользователь sit, необходимо воспользоваться командой groups:

```
sit@ubuntu:~$ groups
sit adm cdrom sudo dip plugdev lxd lpadmin sambashare
```

Из этого примера видно, что пользователь sit состоит в группах sit, adm, cdrom, sudo, dip, plugdev, lxd, lpadmin, sambashare. Если вы хотите посмотреть, в каких группах состоит другой пользователь, то передайте его имя в качестве аргумента.

```
sit@ubuntu:~$ groups root
root : root
```

Изменение пользователя и группы владельца

Чтобы изменить владельца или группу файла (или другого объекта) используется команды chown или chgrp соответственно. Сначала нужно передать имя группы или владельца, а потом список файлов.

```
chown sit /home/sit/itmo.txt
chgrp sit /home/sit/itmo.txt
```

Можно также изменять пользователя и группу одновременно используя команду chown в другой форме:

```
chown sit:sit /home/sit/itmo.txt
```

Предупреждение

Вы не можете использовать команду chown без прав суперпользователя, но chgrp может быть использована всеми, чтобы изменить группу-владельца файла на ту группу, к которой они принадлежат.

Знакомство с chmod

chown и chgrp используются для изменения владельца и группы объекта файловой системы, но кроме них существует и другая программа, называемая chmod, которая используется для изменения прав доступа на чтение, запись и исполнение, которые мы видим в выводе команды ls -l. Команда chmod использует два и более аргументов: метод, описывающий как именно необходимо изменить права доступа с последующим именем файла или списком файлов, к которым необходимо применить эти изменения:

```
chmod +x /home/sit/itmo.sh
```

В примере выше в качестве метода указано +x. Как можно догадаться, метод +x указывает chmod, что файл необходимо сделать исполняемым для пользователя, группы и для всех остальных. Если мы решим отнять все права на исполнение файла, то сделаем вот так:

```
chmod +x /home/sit/itmo.sh
```

Разделение между пользователем, группой и всеми остальными

Часто бывает удобно изменить только один или два набора прав доступа за раз. Чтобы сделать это, просто необходимо использовать специальный символ для обозначения набора прав доступа, который необходимо изменить, со знаком «+»» или «—» перед ним. Символ «u» для пользователя, «g» для группы и «o» для остальных пользователей.

```
chmod go-w /home/sit/itmo.sh
```

Данный пример удаляет право на запись для группы и всех остальных пользователей, но оставляет права владельца нетронутыми.

Числовые режимы

Существует еще один достаточно распространенный способ указания прав: использование четырехзначных восьмеричных чисел. Этот синтаксис, называется числовым синтаксисом прав доступа, где каждая цифра представляет тройку разрешений. Например, в 0777, 777 устанавливают флаги для владельца, группы, и остальных пользователей. Ниже таблица показывающая как транслируются права доступа на числовые значения.

```
Режим   Число
rwx     7
rw-     6
r-x     5
r--     4
-wx     3
-w-     2
--x     1
---     0
```

umask

Когда процесс создает новый файл, он указывает, какие права доступа нужно задать для данного файла. Зачастую запрашиваются права 0666 (чтение и запись всеми), что дает больше разрешений, чем необходимо в большинстве случаев. К счастью, каждый раз, когда в Linux создается новый файл, система обращается к параметру, называемому umask. Система использует значение umask чтобы понизить изначально задаваемые разрешения на что-то более разумное и безопасное. Вы можете просмотреть текущие настройки umask набрав umask в командной строке:

```
sit@ubuntu:~$ umask 0002
```

В Linux-системах значением по умолчанию для umask является 0022, что позволяет другим читать ваши новые файлы (если они могут до них добраться), но не изменять их. Чтобы автоматически обеспечивать больший уровень защищенности для создаваемых файлов, можно изменить настройки umask:

```
sit@ubuntu:~$ umask 0077
```

Такое значение umask приведет к тому, что группа и прочие не будут иметь совершенно никаких прав доступа для всех, вновь созданных файлов.

В отличие от «обычного» назначения прав доступа к файлу, umask задает какие права доступа **должны быть отключены**. Снова посмотрим на таблицу соответствия значений чисел и методов:

```
Режим   Число
rwx     7
rw-     6
r-x     5
r--     4
-wx     3
-w-     2
--x     1
---     0
```

Воспользовавшись этой таблицей мы видим, что последние три знака в 0077 обозначают —rwxrwx.

umask показывает системе, какие права доступа отключить. Совместив первое и второе становится видно, что все права для группы и остальных пользователей будут отключены, в то время как права владельца останутся нетронутыми.

Изменение suid и sgid

Способ установки и удаления битов suid и sgid чрезвычайно прост. Чтобы задать бит suid:

```
chmod u+s /home/sit/itmo.sh
```

Чтобы задать бит sgid:

```
chmod g+s /home/sit/itmo/
```

Определение первого знака прав доступа

Он используется для задания битов sticky, suid и sgid совместно с правами доступа:

```
suid    sgid    sticky  режим
on      on      on      7
on      on      off     6
on      off     on      5
on      off     off     4
off     on      on      3
off     on      off     2
off     off     on      1
off     off     off     0
```

Ниже приведен пример того, как использовать четырех значный режим для установки прав доступа на директорию.

```
sit@ubuntu:~$ chmod 4775 /home/sit/itmo
sit@ubuntu:~$ ls -l /home/sit/itmo
-rwsrwxr-x 1 sit sit 0 Sep  9 12:42 /home/sit/itmo
```

Консольные команды:

- id <печать идентификатора пользователя>
- chgrp <изменить группу файла>
- chown <изменить владельца и группу файлов>
- chmod <изменить права доступа к файлу>
- usermod <изменение параметров учетной записи пользователя>
- useradd <создание нового пользователя>
- userdel <удаление пользователя>
- whoami <определение текущего пользователя>
- umask <определение или установление маски прав доступа для вновь создаваемых файлов>
- sudo su <получение прав суперпользователя>
- groups <определение к каким группам принадлежит пользователь>

Задания к лабораторной работе

Подготовка

- Откройте два терминала. В одном из них получите права суперпользователя используя команду sudo su -
- Изучите как создать пользователя с домашним каталогом с помощью команды useradd из справочной документации man
- Используя useradd создайте пользователей «tom», «jerry», «housewife» с домашним каталогом «tom», «jerry» и «housewife» соответственно.
- Установите пароль для новых пользователей с помощью команды passwd
- Создайте группы: house, kitchen
- Создайте директорию /opt/house права: 755, владелец: housewife группа: house
- Создайте директорию /opt/house/kitchen права: 755, владелец: housewife, группа: kitchen
- Добавить пользователя «housewife» в группы «house, kitchen»
- Выйдите из суперпользователя командой exit

Знакомство

- Войдите под первым терминалом в пользователя «tom», во втором в пользователя «jerry».
- Посмотрите какой идентификатор получил пользователь «tom» и пользователь «jerry» используя команду id
- Посмотрите права доступа на домашний каталог пользователей «tom», «jerry» и «housewife», используя команду ls
- Можете ли вы зайти и посмотреть содержимое домашних каталогов других пользователей? Почему?

Командная оболочка (shells)

- Посмотрите какая оболочка стоит по-умолчанию, она не очень удобна
- Посмотрите какие командные оболочки есть в системе
- Замените командную оболочку пользователей на bash

Эксперименты

- Открыть «Терминал» зарегистрироваться под пользователем «tom»
- Создайте файл в «/opt/house/catmat» под пользователем «tom» с маской 0077 используя umask
- Открыть второй «Терминал» зарегистрироваться под пользователем «jerry»
- Создайте файл в «/opt/house/mousehole» под пользователем «jerry» с маской 0077 используя umask
- Получилось ли это сделать и почему?
- Добавьте нужные права, чтобы пользователи «tom», «jerry», смогли создать в «/opt/house/» свои каталоги
- Пусть пользователь «housewife» создаст каталог «big_fridge» (большой холодильник) и в нем файлы «cheese», «sausage», «meet»
- Внесите какую-нибудь информацию в эти файлы, используя консольный текстовый редактор vim или nano
- Посмотрите пользователи «tom», «jerry» могут увидеть содержимое «холодильника» домохозяйки?
- Пользователи «tom», «jerry» пусть попробуют прочитать содержимое файлов «cheese», «sausage», «meet» используя команду cat
- Том или джерри могут что-то добавить в холодильник ? создать файлы в каталоге «big_fridge»? Добавить в файлы «cheese», «sausage», «meet» какую-нибудь информацию.
- Может ли том или джерри что-то «украсть» из холодильника – перенести файл «cheese», «meet» из «big_fridge» в свои каталоги – коврик кота «/opt/house/catmat» или мышинный домик «/opt/house/mousehole»
- «Закроем» холодильник от Тома и Джерри – изменим права на каталог «big_fridge», чтобы Том и Джери не могли больше перемещать себе файлы из холодильника
- Закрыть так, чтобы Том и Джери не смогли даже посмотреть что в холодильнике
- Может ли домохозяйка забрать обратно в холодильник украденное? Почему?
- Пусть Том создаст дома (в папке «/opt/house») свой файл «cat_toys», Джерри – «mouse_toys»
- Может ли Джерри удалить игрушку Тома «cat_toys», почему?
- Как сделать так, чтобы Том и Джерри не выкидывали игрушки друг друга из дома, а выбрасывали только те которые принадлежат им.

- Может ли джерри украсть игрушку «cat_toys» тома к себе в /opt/house/mousehole ?
- Восполнить пропажу продуктов из холодильника. Создайте файл bash-скрипт, который сгенерирует нам продукты и запустим его. Что нужно чтобы выполнить bash сript ?
- Ознакомьтесь как удалить пользователя вместе с содержимым его домашнего каталога из справочной документации
- Удалите пользователя «tom» вместе с его домашним каталогом.

Вопросы к лабораторной работе

1. Какой uid у пользователя tom? В какие группы он входит?
2. Какие права доступа установлены на домашний каталог пользователя «jerry» ?
3. Как рекурсивно изменить права доступа на файлы в каталоге?
4. Как можно осуществлять переключение между пользователями в рамках одного терминала?
5. Как удалить пользователя при этом сохранив его домашний каталог и данные внутри него?
6. Какое значение umask нужно установить, чтобы владелец и группа имели право на чтение, запись и исполнение, а все остальные пользователи не имели никаких прав?
7. Как рекурсивно снять все suid биты с файлов в каталоге?
8. Как разрешить программе (файлу) исполняться?
9. Что такое бит sticky? Для чего он предназначен?
10. Зачем нужны uid и gid?
11. Почему uid пользователя задается больше 1000?