Лабораторная работа № 2

SUID, SGUID, Sticky bit, ACL, IPTables.

Цель работы: Изучить модификаторы доступа SUID, SGUID, расширенные настройки прав доступа к файлам и каталогам, утилита для управления работой межсетевым экраном iptables.

Операционные системы: Debian 8.

Теоретический материал

В Unix-подобных системах приложение запускается с правами пользователя, вызвавшего указанное приложение. Это обеспечивает дополнительную безопасность, так как 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, независимо от того — кто запускает эту программу, она при выполнении имеет права хозяина этого файла. Аналогичная ситуация с GUID - запуск будет выполнен с правами группы, которой принадлежит файл.

- Восьмеричные значения для SUID и SGID - **4000** и **2000**.
- Символьные: **u+s** и **g+s**.

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

- **chmod u+s file1.sh** - устанавливает на файл file1.sh бит SUID
- **chmod g+s file1.sh** - устанавливает на файл file1.sh бит GUID

Проверить установку модификатора доступа можно просмотрев права доступа:

```
-rwxrwsrwx 1 root root    0 2016-04-08 16:16 file1
```

Видно, что для файла установлен SGID, о чем свидетельствует символ «**s**» (-rwxrwsrwx).

Еще одно важное усовершенствование касается использования sticky-бита в каталогах. Каталог с установленным sticky-битом означает, что удалить файл из этого каталога может только владелец файла или суперпользователь. Другие пользователи лишаются права удалять файлы. 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. Заметьте, что владелец 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 dir** - с заменой прав;
- **chmod +t dir** - добавление к текущим правам.

Проверить установку модификатора доступа можно просмотрев права доступа:

```
drwxr-xr-t 2 root root 4096 2016-04-08 16:18 student
```

Видно, что для файла установлен Sticky-бит, о чем свидетельствует символ «**t**» (drwxr-xr-t).

Access Control List - списки контроля доступа

Для реализации сложных структур прав доступа используются расширенные права - ACL (Access control list - списки контроля доступа). Списки контроля доступом (ACL) дают большую гибкость, чем стандартный набор полномочий «пользователь/группа/остальные».

Существуют два типа ACL:

- **ACL для доступа** — это список управления доступом для заданного файла или каталога. Проще говоря - это сами права на объект, которые будут контролировать доступ к этому объекту.
- **ACL по умолчанию** - может быть связан только с каталогом, и, если файл в этом каталоге не имеет ACL для доступа, используются правила, определённые в ACL по умолчанию. ACL по умолчанию являются необязательными.

Управления ACL списками осуществляется всего лишь двумя командами: setfacl, getfacl.

Утилита getfacl: Выводит листинг ACL прав для указанных объектов.

Примеры использования:

- **getfacl *** - отобразит права ACL для всех объектов в текущем каталоге;
- **getfacl file.txt** - отобразить ACL для файла file.txt;
- **getfacl -R *** - отобразит ACL для всех объектов (включая подкаталоги и их содержимое) текущего каталога.

Чтобы посмотреть, установлены ли ACL на объектах, достаточно воспользоваться командой **ls -l**:

```
-rwxr-x---+ 1 root root   19 2016-04-08 16:20 file
```

Видно, что для файла установлен ACL, о чем свидетельствует символ «+» (-rwxr-x—+).

Пример вывода команды getfacl для file:

```
# file: qwert    - Имя файла
# owner: root    - Владелец файла (основные права Unix)
# group: root    - Группа файла (основные права Unix)
user::rwx        - Права для владельца файла (основные права Unix)
user:child:rw-   - Права ACL для пользователя child
group::r--       - Права для группы файла (основные права Unix)
mask::rw-        - Эффективная маска
other::---       - Права для пользователя "все остальные"
```

Утилита setfacl: предназначена для установки, модификации или удаления ACL.

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 простой синтаксис **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:

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 Unix, в противном случае будет давать ошибку;
`-m` или `-M file*`	- Модифицирует указанные ACL на объекте. Другие существующие ACL сохраняются.
`-x` или `-X file*`	- Удаляет указанные ACL права с объекта. Стандартные права Unix не изменяются.

Часто используемые 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 объект.
`-R`	- Рекурсивное назначение (удаление) прав

Формирование списка правил:

Синтаксис	Описание
`u:<uid>:<perms>*`	- Назначает ACL для доступа заданному пользователю. Здесь можно указать имя или UID пользователя. Это может быть любой пользователь, допустимый в данной системе.
`g:<gid>:<perms>*`	- Назначает ACL для доступа заданной группе. Здесь можно указать имя или GID группы. Это может быть любая группа, допустимая в данной системе.
`o:<perms>*`	- Назначает ACL для доступа пользователям, не включённым в группу файла. Это пользователь «все остальные», как в стандартных правах Unix.

Пример установки прав:

```
setfacl -m u:student:rw file.txt
```

Назначает пользователю student права на чтение и запись file.txt.

Дополнительную информацию по утилитам getfacl и setfacl смотрите в соответствующих man руководствах.

IPTables

IPTables утилита командной строки, является стандартным интерфейсом управления работой межсетевого экрана (брандмауэра) netfilter.

Ключевыми понятиями в IPTables являются:

- **Правило** — состоит из *критерия*, *действия* и *счетчика*. Если пакет соответствует критерию, к нему применяется действие, и он учитывается счетчиком. Критерия может и не быть — тогда неявно предполагается критерий «все пакеты». Указывать действие тоже не обязательно — в отсутствие действия правило будет работать только как счетчик.
- **Критерий** — логическое выражение, анализирующее свойства пакета и/или соединения и определяющее, подпадает ли данный конкретный пакет под действие текущего правила.
- **Действие** — описание действия, которое нужно проделать с пакетом и/или соединением в том случае, если они подпадают под действие этого правила. О действиях более подробно будет рассказано ниже.

- **Счетчик** — компонент правила, обеспечивающий учет количества пакетов, которые попали под критерий данного правила. Также счетчик учитывает суммарный объем таких пакетов в байтах.
- **Цепочка** — упорядоченная последовательность правил. Цепочки можно разделить на *пользовательские* и *базовые*.
- **Базовая цепочка** — цепочка, создаваемая по умолчанию при инициализации таблицы. Каждый пакет, в зависимости от того, предназначен ли он самому хосту, сгенерирован им или является транзитным, должен пройти положенный ему набор базовых цепочек различных таблиц. Схема следования пакетов приведена на рисунке. Кроме того, базовая цепочка отличается от пользовательской наличием «действия по умолчанию» (default policy). Это действие применяется к тем пакетам, которые не были обработаны другими правилами этой цепочки и вызванных из нее цепочек (см. переходы). Имена базовых цепочек всегда записываются в верхнем регистре (PREROUTING, INPUT, FORWARD, OUTPUT, POSTROUTING).
- **Пользовательская цепочка** — цепочка, созданная пользователем. Может использоваться только в пределах своей таблицы. Рекомендуется не использовать для таких цепочек имена в верхнем регистре, чтобы избежать путаницы с базовыми цепочками и встроенными действиями.
- **Таблица** — совокупность *базовых* и *пользовательских* цепочек, объединенных общим функциональным назначением. Имена таблиц (как и модулей критериев) записываются в нижнем регистре, так как в принципе не могут конфликтовать с именами пользовательских цепочек. При вызове команды iptables таблица указывается в формате `-t имя_таблицы`. При отсутствии явного указания, используется таблица filter. Более подробно таблицы будут рассмотрены ниже.

Таблицы IPTABLES

У IPTABLES имеется 4 встроенных типа таблиц.

1. Filter Table

Является таблицей по-умолчанию. Если при создании/изменении правила не указана таблица – используется именно `filter`. Используется в основном для фильтрации пакетов. К примеру, тут можно выполнить `DROP, LOG, ACCEPT` или `REJECT` без каких либо сложностей, как в других таблицах. Использует 3 встроенных цепочки:

- `INPUT` chain – входящие пакеты, используется только для пакетов, цель которых – сам сервер, не используется для транзитного (роутинга) трафика
- `OUTPUT` chain – исходящие пакеты, созданные локально и отправленные "за пределы" сервера;
- `FORWARD` chain – пакеты, предназначенные другому сетевому интерфейсу (роут на другие машины сети, например).

2. NAT table

Таблица `nat` используется главным образом для преобразования сетевых адресов (Network Address Translation). Через эту таблицу проходит только первый пакет из потока. Преобразования адресов автоматически применяется ко всем последующим пакетам. Это один из факторов, исходя из которых мы не должны осуществлять какую-либо фильтрацию в этой таблице.

- `PREROUTING` chain – преобразование адресов `DNAT` (Destination Network Address Translation), фильтрация пакетов здесь допускается только в исключительных случаях;
- `POSTROUTING` chain – выполняется преобразование адресов `SNAT` (Source Network Address Translation), фильтрация пакетов здесь крайне нежелательна;
- `OUTPUT` chain – `NAT` для локально сгенерированных пакетов;

3. Mangle table

Таблица `Mangle` предназначена только для внесения изменения в некоторые заголовки пакетов – `TOS` (Type of Service), `TTL` (Time to Live), `MARK` (особая метка для **IPTABLES** или других служб). **Важно**: в действительности поле `MARK` не изменяется, но в памяти ядра заводится структура, которая

сопровождает данный пакет все время его прохождения через машину, так что другие правила и приложения на данной машине (и только на данной машине) могут использовать это поле в своих целях.

Включает в себя такие цепочки:

- PREROUTING chain
- OUTPUT chain
- FORWARD chain
- INPUT chain
- POSTROUTING chain

4. Raw table

Применяется до передачи пакета механизму определения состояний (state machine, connection tracking – система трассировки соединений, при помощи которой реализуется межсетевой экран на сеансовом уровне (stateful firewall), позволяет определить, к какому соединению или сеансу принадлежит пакет, анализирует все пакеты кроме тех, которые были помечены NOTRACK в таблице raw).

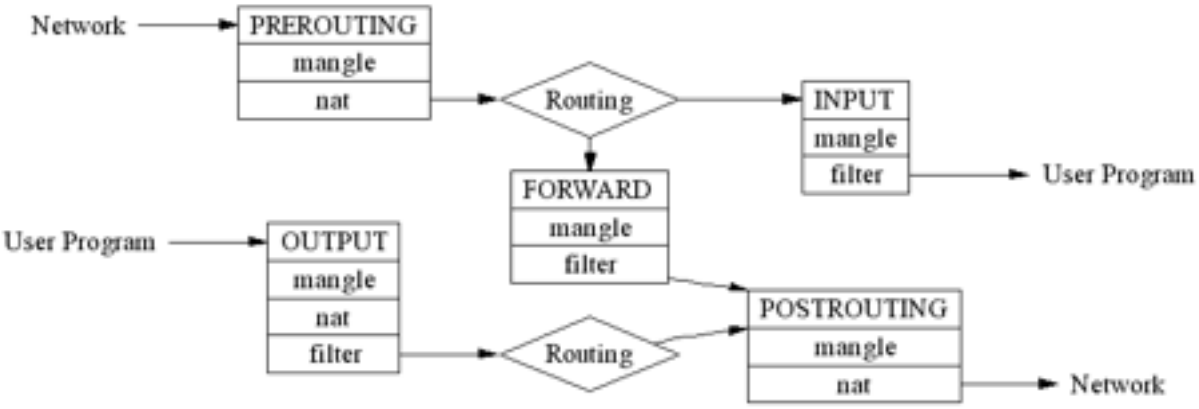
- PREROUTING chain
- OUTPUT chain

Цепочки IPTABLES

Существует 5 типов стандартных цепочек, встроенных в систему:

- PREROUTING — для изначальной обработки входящих пакетов;
- INPUT — для входящих пакетов адресованных непосредственно локальному процессу (клиенту или серверу);
- FORWARD — для входящих пакетов перенаправленных на выход (заметьте, что перенаправляемые пакеты проходят сначала цепь PREROUTING, затем FORWARD и POSTROUTING);
- OUTPUT — для пакетов генерируемых локальными процессами;
- POSTROUTING — для окончательной обработки исходящих пакетов.

Схематично путь пакетов через **IPTABLES** хорошо представлен на следующей схеме:



Правила IPTABLES

Правила имеют следующую структуру:

правило > цель > счётчик

Если пакет соответствует правилу, к нему применяется цель, и он учитывается счетчиком. Если правило (или критерий) не задан – то цель применяется ко всем проходящим через цепочку пакетам. Если не указаны ни цель, ни правило – для правила будет срабатывать только счётчик пакетов. Если пакет не попадает под правило и цель – он передаётся следующему правилу в списке.

Цели (targets) IPTABLES

Правило может содержать одно из следующих целей (наиболее часто встречаемые, полный список в соответствующем man руководстве):

- `ACCEPT` – принять пакет, и передать следующей цепочке (или приложению, или передать для дальнейшего роутинга);
- `DNAT` – (Destination Network Address Translation) используется для преобразования адреса места назначения в **IP** заголовке пакета; если пакет подпадает под критерий правила, выполняющего `DNAT`, то этот пакет, и все последующие пакеты из этого же потока, будут подвергнуты преобразованию адреса назначения и переданы на требуемое устройство, хост или сеть;

действие DNAT может выполняться только в цепочках PREROUTING и OUTPUT таблицы nat, и во вложенных под-цепочках; важно запомнить, что вложенные подцепочки, реализующие DNAT не должны вызываться из других цепочек, кроме PREROUTING и OUTPUT;

- DROP – просто “сбрасывает” пакет и **IPTABLES** “забывает” о его существовании; “сброшенные” пакеты прекращают свое движение полностью, т.е. они не передаются в другие таблицы, как это происходит в случае с действием ACCEPT; следует помнить, что данное действие может иметь негативные последствия, поскольку может оставлять незакрытые “мертвые” сокеты как на стороне сервера, так и на стороне клиента, наилучшим способом защиты будет использование действия REJECT, особенно при защите от сканирования портов;
- REDIRECT – выполняет перенаправление пакетов и потоков на другой порт той же самой машины; можно пакеты, поступающие на **HTTP** порт перенаправить на порт HTTP-proxy; удобен для выполнения “прозрачного” проксирования (transparent proxy), когда машины в локальной сети даже не подозревают о существовании прокси; может использоваться только в цепочках PREROUTING и OUTPUT таблицы nat;
- REJECT – используется, как правило, в тех же самых ситуациях, что и DROP, но в отличие от DROP, команда REJECT выдает сообщение об ошибке на хост, передавший пакет;
- RETURN – прекращает движение пакета по текущей цепочке правил и производит возврат следующему правилу в вызывающей (предыдущей) цепочке, если текущая цепочка была вложенной, или, если текущая цепочка лежит на самом верхнем уровне (например INPUT), то к пакету будет применена политика по-умолчанию; обычно, в качестве политики по-умолчанию назначают действия ACCEPT или DROP;

Критерий так же может использовать состояние пакета для принятия решений:

- NEW — пакет открывает новый сеанс, например — пакет TCP с флагом SYN;
- ESTABLISHED — пакет является частью уже существующего сеанса;
- RELATED — пакет открывает новый сеанс, связанный с уже открытым сеансом, например, во время сеанса пассивного **FTP**, клиент подсоединяется к порту 21 сервера, сервер сообщает клиенту номер второго, случайно выбранного порта, после чего клиент подсоединяется ко второму порту для передачи файлов; в этом случае второй сеанс (передача файлов по второму порту) связан с уже существующим сеансом (изначальное подсоединение к порту 21);
- INVALID — все прочие пакеты.

Управление IPTables осуществляется с помощью команды iptables, все ключи можно узнать прочитав man руководство.

Задание

1. Описать положительный и отрицательные стороны установки модификаторов доступа SUID, SGUID и Sticky-bit.
2. Установить на файл права с использованием ACL, затем задать более жесткие (ограничивающие) базовые права (при помощи chmod) для пользователя. Посмотрите что произойдет, какие права будут работать, сделайте вывод.
3. Создать каталог, установить владельца пользователя teacher. Установить права для teacher - rwx, остальным запретить все. Задать для каталога ACL по умолчанию. В каталоге создать, файл и подкаталог, проверить их права. Сделать вывод.
4. Произвести настройку межсетевого экрана с использованием iptables:
    - Запретить все соединения (изначально все что не разрешено - запрещено).

- Разрешить локальный траффик.
- Разрешить все уже инициированные входящие соединения, а так же их дочерние.
- Разрешить все новые, уже инициированные исходящие соединения, а так же их дочерние.
- Открыть порт для SSH (порт 22).
- Открыть порт для HTTP(порт 80).
- Открыть порт для HTTPS(порт 443).
- Разрешить POP3(порт 110) только с IP 173.194.71.17.
- Разрешить DNS (порт 53)

Отчет необходимо оформить по шаблону с сайта «ЛЭТИ»(титульник, цель работы, ход выполнения работы, вывод) и сопроводить скриншотами с этапами выполнения задания, все задания выполняются с использованием терминала!

Отчеты присылать на почту ZOC.leti@yandex.ru . В теме письма должно содержаться имя, фамилия, группа, номер лабораторной работы и слово «Linux».