Зертханалық сабақ №8: Аутентификация қосылатын модульдері (PAM)

Linux - это многопользовательская среда и чтобы пользователь мог начать работу в системе ему нужно пройти процедуру аутентификации. **PAM** (**Pluggable Authentication Modules**) - это система (механизм), которая берет на себя работу по реализации процедур аутентификации. До появления **PAM**, разработчикам программ, которые были так или иначе связаны с аутентификацией, приходилось подстраивать свою программу под существующие механизмы аутентификации. Соответственно если менялись механизмы аутентификации, значит нужно было менять и программы которые использовали их. Поэтому была разработана система **PAM**, которая является "прослойкой" между программами и механизмами аутентификации. То есть теперь программы аутентификации (например, программа **login**) должны всего лишь уметь работать с системой **PAM**. Программа передает **PAM** параметры (например логин и пароль) и ее (программу) уже не "интересует" какой способ аутентификации реализован в системе - аутентификация по паролу или смарт-карте или другой способ. Дальше работает **PAM** и возвращает программе или успех или отказ.

Посмотрим на систему **PAM** подробнее. Основные функции, или действия, или задачи которые выполняет система **PAM** - разбиты на четыре группы которые имеют определенные названия:

группа **auth** - это действия связанные непосредственно с аутентификацией. То есть действия или функции которые позволяют определить, что вы это вы. Это может быть аутентификация по паролю, по смарт-карте, биометрическая аутентификация (отпечаток пальца и т.д.) и другие.

группа **account** - это действия связанные с управлением учетными записями. Например, даже если вы аутентифицировались в системе, то вашей учетной записи можно поставить запрет на работу в определенное время суток. Или разрешить заходить в консольном режиме, но запретить заходить в графическом режиме. И т.д.

группа **session** - действия этой группы осуществляют выделение пользователю необходимых для работы ресурсов. Самый простой пример - это разрешение на монтирование каталогов.

группа **password** - действия, которые реализуют изменение аутентификационных данных пользователя. Чаще всего это действия по управлению паролями пользователя.

Все эти действия или процедуры (функции) реализованы в виде отдельных модулей, которые расположены в каталоге **/lib/security/**. То есть можно сказать, есть модули группы **auth**, модули группы **account** и т.д. Соответственно система **PAM** является модульной и если вам необходимо реализовать биометрическую аутентификацию, то необходимо просто установить модуль, которых может это процедуру выполнить.

Основной конфигурационный файл системы **PAM** - это файл **/etc/pam.conf**. Кроме файла **/etc/pam.conf**, настройки **PAM** хранятся в файлах каталога **/etc/pam.d/**. Внутри каталога находятся текстовые файлы которые содержат в себе последовательность действий (некий алгоритм) для программ которые используют **PAM**. Например, файл **/etc/pam.d/login** содержит алгоритм работы системы **PAM** для программы **login**, а файл **/etc/pam.d/passwd** для программы **passwd**.

Рассмотрим сначала формат файла **/etc/pam.conf**. Файл состоит из строк. Файл может состоять из одной строки, а может из нескольких строк складываясь в цепочку последовательных действий. Каждая строка описывает одно правило или один шаг такой цепочки (алгоритма). Строка состоит из четырех полей. Первое поле это имя программы к которой относится данный шаг. Второе поле, это тип действия (**auth**, **account**, **session**, **password**). Третье поле это поле в котором задается поведение системы **PAM** после завершения этого шага на этом шаге алгоритма (чуть ниже остановимся подробнее на этом вопросе). Четвертое поле - это имя файла модуля. Также в строке могут присутствовать некоторые параметры передаваемые модулю.

Структура файлов находящихся в каталоге **/etc/pam.d/**, такая же. Отличие только в отсутствии первого поля - имени. Так как имя программы берется из имени самого файла. Посмотрим на пример такого файла. Назовем его **testpam**.

```
auth     sufficient     pam_rootok.so
auth     required       pam_unix.so
account  required       pam_unix.so
```

Рассмотрим первую строку. Поле **auth** говорит, что первым шагом будет аутентификация. Третье поле - это модуль, который будет выполнять аутентификацию и возвращать результат выполнения. В данном примере модуль **pam_rootok.so** проверяет является ли учетная запись рутом (**root**). Если, да то будет возвращен успех (true), если нет, то будет возвращена ошибка или отказ (false). Второе поле - это реакция или влияние полученного результата на цепочку в целом.

Реакция может быть четырех типов: **required**, **requisite**, **optional**, **sufficient**. На примере строки **auth sufficient pam_rootok.so** рассмотрим, что означают эти значения.

Если во втором поле установлено значение **requisite**, то это означает, что если модуль **pam_rootok.so** завершился с ошибкой, то дальнейшее выполнение файла **testpam** прерывается и система **PAM** возвращает приложению ошибку. Если модуль вернул положительные результат, то выполнение цепочки продолжается.

required похож на **requisite**. Если модуль **pam_rootok.so** завершился с ошибкой, то **PAM** также вернет, ошибку, но после того как будут выполнены остальные модули, то есть цепочка не прерывается. Если модуль вернул положительные результат, то выполнение цепочки продолжается.

sufficient - если модуль **pam_rootok.so** вернул успех, то система **PAM** возвращает приложению успех, и дальнейшее выполнение цепочки прерывается. Если неудача, то продолжается выполнение цепочки.

optional - этот параметр никак не влияет на ход цепочки. Указывается для тех модулей которые не выполняют никаких проверочных действий. Если в файле будут только строки с параметром **optional**, то **PAM** вернет приложению успех.

Более подробно о системе **PAM** и назначении той или иной библиотеки можно прочитать на сайте http://kernel.org/pub/linux/libs/pam/Linux-PAM-html/Linux-PAM_SAG.html. Сейчас выполним небольшое практическое упражнение которое позволить лучше понять как работает система **PAM** и как составлять конфигурационные файлы.

Перейдите в каталог **/etc/pam.d/**. Скопируйте файл **su** в домашнюю директорию (чтобы можно было восстановить его) и удалите файл в **su** из директории **/etc/pam.d/**. Попробуйте теперь выполнить команду **su** в терминале чтобы перейти в режим суперпользователя. После ввода пароля система выдаст ошибку аутентификации, так как отсутствует конфигурационный файл для программы **su**.

Создаем файл **/etc/pam.d/su** и пишем в нем такую строку:

```
auth sufficient pam_permit.so
```

Сохраняем файл. Пробуем снова выполнить команду **su**, и видим, что теперь мы становимся суперпользователем без запроса пароля. Это произошло потому, что модуль **pam_permit.so** всегда возвращает положительный результат, **sufficient** тут же прерывает выполнение цепочки и система **PAM** возвращает положительный результат. Отредактируем файл к следующему виду:

```
auth required pam_permit.so
auth requisite pam_deny.so
auth sufficient pam_permit.so
```

Модуль **pam_deny.so** всегда возвращает ошибку. Какой будет результат? Проверьте. А если заменить **requisite** на **required**?
Теперь напишем в файле следующее правило:

```
auth    required pam_unix.so
```

Теперь после выполнения команды **su** будет запрошен пароль пользователя **root**. Если пароль ввести правильно, то вы станете рутом, если пароль будет неверный, то останетесь обычным пользователем. Теперь добавим в файл еще одну строку, так чтобы получились следующие правила:

```
auth    requisite pam_wheel.so
auth    required pam_unix.so
```

Модуль **pam_wheel.so** возвращает успех если учетная запись пользователя принадлежит группе **wheel**. Если попробовать сейчас выполнить команду **su**, то она тут же завершиться с ошибкой. То есть сейчас команду **su** смогут выполнить только пользователи, который входят в группу **wheel** и знают пароль учетной записи **root**. Если создать группу **wheel** и добавить туда свою учетную запись, то команда **su** сработает.

Вот еще пример:

```
auth    requisite pam_wheel.so
auth    required pam_permit.so
```

Попробуйте ответить кто сможет успешно выполнить команду **su** и, что для этого нужно будет сделать? На этом завершим практическое упражнение (не забудьте вернуть на место оригинальный файл su).

Хочу еще раз подчеркнуть, что конфигурационные файлы в каталоге /etc/pam.d/ можно создавать только для файлов которые используют систему **PAM**. Например, если создать файл **/etc/pam.d/ls** со строкой **auth requisite pam_deny.so**, то команда **ls** все равно будет выполнятся так как она не использует систему **PAM**. Чтобы проверить использует ли команда систему PAM можно использовать команду **ldd**, которой в качестве параметра передается полный путь к файлу команды. Например:

```
ldd /bin/su
    linux-gate.so.1 =>  (0x008a5000)
    libpam.so.0 => /lib/libpam.so.0 (0x0073f000)
    libpam_misc.so.0 => /lib/libpam_misc.so.0 (0x009e8000)
    libc.so.6 => /lib/tls/i686/cmov/libc.so.6 (0x0025e000)
    libdl.so.2 => /lib/tls/i686/cmov/libdl.so.2 (0x00eed000)
    /lib/ld-linux.so.2 (0x00f75000)
```

Команда **ldd** покажет какие библиотеки использует программа и если в перечне есть **libpam.so.0**, **libpam_misc.so.0** значит программа использует систему **PAM**.

В завершении хочу еще упомянуть о таком файле как **/etc/nsswitch.conf**. Первые три строки этого файла как раз и задают какая система аутентификации будет работать в системе:

```
passwd:         compat
group:          compat
shadow:         compat
```

Ключевое слово **compat** как раз и “говорит” о том что в качестве системы аутентификации, будет использована система **PAM**.

И еще. Будьте осторожны в экспериментах с **PAM**. По незнанию или неосторожности можно запросто заблокировать свою систему. Поэтому перед тем как что-то менять обязательно сохраните исходные конфигурационные файлы, чтобы в случае проблем можно было их быстро восстановить.