**Лабораторная работа № 5 – Контроль доступа в ОС Linux**

**Цель работы –** изучение подсистемы контроля доступа ОС Linux.

**Теоретические сведения**

Традиционный дискреционный контроль доступа (*DAC, Discretionary Access Control*) UNIX-подобных систем подразумевает наличие владельца у каждого защищаемого объекта. Данный владелец определяет права доступа пользователей к объекту. Каждый процесс запускается от имени определённого пользователя и права доступа к объектам определяются на основании идентификации субъектов или групп. Субъекты также могут передавать права другим субъектам. В Unix каждый объект представляется файлом. Для каждого файла задаются 3 группы прав: для владельца, для членов своей группы и для всех прочих. В каждую группу входят права на чтение, запись и исполнение (прохождение для файлов-каталогов).

POSIX ACL (*Access control list*) – cписки контроля доступа. Списки контроля доступа расширяют классическую проверку прав доступа Unix и дают большую гибкость, чем стандартный набор полномочий DAC, позволяя управлять доступом отдельных субъектов (не только UID, GID) к объектам.

Существуют два типа ACL:

1. ACL для доступа;
2. ACL по умолчанию.

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, он использует правила, определённые в ACL по умолчанию, связанном с каталогом. ACL по умолчанию являются необязательными.

Security-Enhanced Linux (*SELinux*) – это система мандатного контроля доступа, реализованная на уровне ядра. Система SELinux унаследовала архитектуру подсистемы безопасности от исследовательской операционной системы Flask. Основной особенностью Flask было использование концепции наименьших привилегий (least privilege), чтобы предоставлять пользователю или приложению только те права доступа, которые необходимы для осуществления запрошенных действий. Эта концепция реализована с помощью принудительной типизации доступа, благодаря чему мандатный доступ в SELinux действует в рамках модели «домен-тип». В этой модели каждый процесс-субъект запускается в определенном контексте (домене) безопасности (то есть имеет какой-то уровень доступа), а всем ресурсам-объектам операционной системы (файлам, директориям, сокетам и пр.) ставится в соответствие определенный тип (уровень секретности).

Благодаря принудительной типизации доступа, возможности SELinux по разграничению прав значительно превосходят возможности базовой дискреционной модели доступа, используемой в UNIX-подобных системах. Например, с помощью SELinux можно строго ограничить номер сетевого порта, к которому будет обращаться сетевой сервер, можно разрешить создание и запись в отдельные файлы, но не их удаление и т.п. Такая градация объектов операционной системы позволяет ограничить системные и пользовательские процессы с помощью явно заданного набора прав доступа к конкретным ресурсам. Если какая-то из служб, контролируемых SELinux, будет взломана, злоумышленник, даже имея права суперпользователя, не сможет пробраться дальше «песочницы», ограниченной набором правил.

Список правил, определяющих разрешения на доступ определенных доменов к определенным типам, и составляет политику безопасности. Политика безопасности задается один раз в момент установки системы и представляет собой набор текстовых файлов, которые загружаются в память ядра Linux при старте системы. Например, в указанном ниже правиле для домена http-сервера разрешается чтение неких файлов, содержащих сетевую конфигурацию:

allow httpd\_t net\_conf\_t:file { read getattr lock ioctl };

Структуру и принцип работы меток, определяющих контекст безопасности объектов и субъектов операционной системы, SELinux унаследовал вместе с моделью «домен-тип» от подсистемы безопасности Flask.

Домен – список действий, которые может выполнять процесс. Обычно в качестве домена определяется минимально-возможный набор действий, при помощи которых процесс способен функционировать. Таким образом, если процесс дискредитирован, злоумышленнику не удастся нанести большого вреда.

Роль – список доменов, которые могут быть применены. Если какого-то домена нет в списке доменов какой-то роли, то действия из этого домена не могут быть применены.

Тип – набор действий, которые допустимы по отношения к объекту. Тип отличается от домена тем, что он может применяться к пайпам, каталогам и файлам, в то время как домен применяется к процессам.

Контекст безопасности – все атрибуты SELinux – роли, типы и домены. Для полноценной защиты, контексты безопасности должны быть определены для каждого объекта и для каждого субъекта системы. Метки имеют следующий вид:

<user>:<role>:<type>

Например, распространенным контекстом безопасности является метка следующего вида: **system\_u:object\_r:httpd\_exec\_t**. Для SELinux пользователь **system\_u** обычно является стандартным обозначением для демонов системы. Роль **object\_r** назначается системным объектам типа обычных файлов или устройств. Тип **httpd\_exec\_t** применяется к исполняемому файлу httpd, расположенному по адресу /usr/sbin/httpd.

На рисунке 1 представлена обобщенная схема обеспечения мандатного контроля доступа с помощью SELinux.

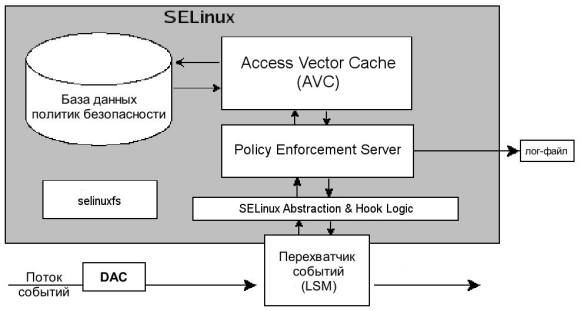


Рисунок 1 – Схема работы SELunx

SELinux состоит из пяти основных компонентов: вспомогательных модулей для работы с файловой системой и для реализации взаимодействия с перехватчиком событий Linux Security Modules, основного механизма организации контроля доступа — Policy Enforcement Server, базы данных политик безопасности системы и Access Vector Cache (AVC) — вспомогательного механизма для повышения производительности.

Работа SELinux организована следующим образом:

1. Субъект операционной системы (процесс) пытается выполнить над определенным объектом (файлом, процессом, сокетом) некоторое действие, разрешенное в рамках стандартной дискреционной системы безопасности (DAC) Linux. Это приводит к запуску потока обращений к объекту.
2. Каждый запрос (обращение) на выполнение действия с объектом перехватывается модулем безопасности Linux Security Modules и вместе с контекстом безопасности субъекта и объекта передается подсистеме SELinux Abstraction & Hook Logic, отвечающей за взаимодействие с LSM.
3. Полученная информация от подсистемы SELinux Abstraction & Hook Logic пересылается основному модулю Policy Enforcement Server (серверу реализации политики безопасности), непосредственно ответственному за принятие решения о доступе субъекта к объекту.
4. Для получения решения о разрешении/запрете действия, сервер реализации политики безопасности обращается к специальной подсистеме Access Vector Cache, кэширующей наиболее часто используемые правила.
5. Если AVC не содержит кэшированного решения для соответствующей политики, то запрос необходимой политики безопасности перенаправляется дальше – в базу данных политик безопасности.
6. Найденная политика безопасности передается серверу политик, принимающему решение.
7. Если запрашиваемое действие удовлетворяет найденной политике, то операция разрешается. В противном случае операция запрещается, а вся информация о принятии решения записывается в лог-файл SELinux.

Помимо принятия решения о разрешении/запрете определенных действий, модуль Policy Enforcement Server отвечает также за выполнение вспомогательных задач, таких как управление метками безопасности (назначение, удаление).

**Формулировка задания**

В ходе выполнения лабораторной работы необходимо выполнить следующие действия:

1. Изучить механизмы разграничения доступа в ОС Linux: стандартный дискреционный контроль доступа (DAC) Linux, списки контроля доступа (ACL) и систему мандатного управления доступом SELinux.
2. Описать три ситуации, когда стандартного механизма разграничения доступа недостаточно, и продемонстрировать эти ограничения, используя классическую проверку прав доступа и списки контроля доступа.
3. Сконфигурировать SELinux таким образом, чтобы предотвратить возможные нарушения доступа из пункта 2.