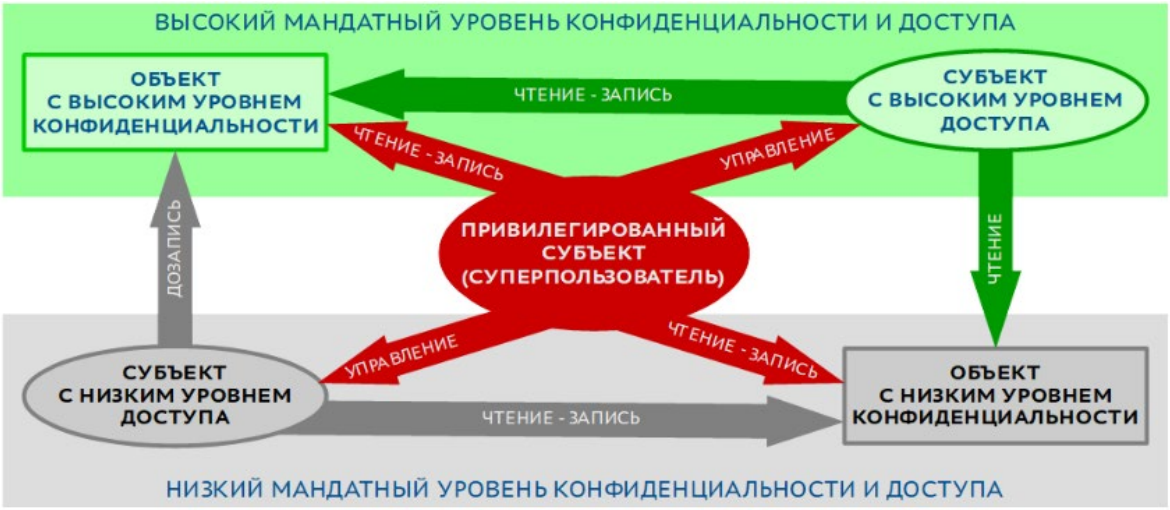
Лабораторная работа №7
«Мандатное управление доступом, мандатный контроль целостности»

1. Теоретическая часть

1.1. Мандатное управление доступом в Astra Linux

Мандатное управление доступом (Mandatory Access Control, MAC) — модель безопасности, которая используется для строгого контроля доступа к объектам (файлам, процессам, ресурсам и т. д.), где права доступа определяются системными политиками, а не владельцами объектов. В отличие от дискреционного управления доступом, где владелец объекта может самостоятельно управлять правами доступа, в модели MAC доступ контролируется централизованно и неизменно для пользователей. Управление доступом реализовано на основе модели Белла-Лападулы. Она заключается в том, что каждому субъекту (пользователю, процессу или программе, запрашивающей доступ к ресурсу) и объекту (ресурсу, к которому запрашивается доступ, например файлу, процессу или устройству) в системе назначаются определенные **уровни конфиденциальности**, исходя из которых происходит предоставление доступа к информации.



Каждый объект и субъект в системе имеет метку безопасности, которая указывает на уровень их доступа и возможные ограничения. Метка безопасности — это атрибут, который назначается системной политикой и содержит информацию о классе или уровне секретности. В примитивной форме метки могут указывать на следующие уровни:

- Не секретно
- Для служебного пользования
- Секретно
- Совершенно секретно

В системе МАС контролируется как чтение, так и запись. Например, субъект с более низким уровнем доступа не может прочитать объект с более высоким уровнем секретности (принцип "чтение вниз"), и субъект с высоким уровнем доступа не может записывать данные в объекты с более низким уровнем (принцип "запись вверх").

Обратим внимание, что уровень конфиденциальности является **иерархическим** атрибутом. Каждый высший уровень включает в себя более низкий. Т.е. человек, способный прочитать совершенно секретную информацию может прочитать и секретную и для служебного пользования и не секретные документы.

Правила МАС часто реализуются в военных, государственных и других организациях, где конфиденциальность данных является критической. Эта модель позволяет исключить случайное раскрытие данных, так как пользователи не могут изменить права доступа к объектам.

1.2. Мандатный контроль целостности в Astra Linux

Мандатное управление доступом требует наличие суперпользователя, который становится самым уязвимым местом в системе. Для решения этой проблемы был разработан мандатный контроль целостности.

Его основное отличие от мандатного управления доступом заключается в том, что вместо уровней безопасности в модели описаны **уровни целостности** объектов доступа и уровни доверия субъектов.

На одном уровне целостности субъект может производить операции чтения и записи. Между уровнями целостности происходит следующее взаимодействие – субъект с низким уровнем доверия может только читать объекты с высоким уровнем, субъект с высоким уровнем может и читать, и писать в объект с низким уровнем доверия.

Таким образом, даже если процесс выполняется от имени суперпользователя, но на низком уровне целостности, то он не сможет изменить содержимого конфиденциальных данных.

Сущностям и субъектам в Astra Linux присваиваются мандатные атрибуты:

- **Уровень конфиденциальности**

  Определяет степень секретности документа и соответствующий уровень доступа к нему. Чаще всего используется в структурах безопасности и применяется для документов с уровнями – «Не секретно», «Для служебного пользования», «Секретно», «Совершенно секретно». В такой системе пользователь с уровнем

секретно не может видеть совершенно секретные документы и не может редактировать не секретные документы. Документы с равной меткой возможно читать и редактировать.

– **Категория конфиденциальности**

  В отличии от уровня конфиденциальности не является иерархической структурой. Пользователи определенной категории видят документы этой категории и не видят другие документы. Например, возможно создать категории для каждого из отделов секретного завода – таким образом члены одного отдела не увидят документы другого отдела. Реализовано с помощью 64 битной маски, где каждый бит соответствует своей категории. Таким образом, выставив несколько битов возможно добавить объект в несколько категорий.

– **Метка целостности**

  Аналогично не является иерархической структурой и представляет из себя 8-ми битовую маску. Все нули обозначают низкий уровень доступа, наоборот все единицы – высокий, выдаваемый только администратору. Промежуточные значения отвечают за работу с определенными сервисами ОС.

Наименование	Значение	Битовая маска
Администратор	063	00111111
СУБД	016	00010000
Графический сервер	008	00001000
Специальное ПО	004	00000100
Виртуализация	002	00000010
Сетевые сервисы	001	00000001
Низкий уровень	000	00000000

1.5.2. Команды для управления мандатным доступом в Astra Linux

Управление мандатным доступом реализовано на основе подсистемы безопасности PARSEC. Все настройки системы хранятся в директории /etc/parsec.

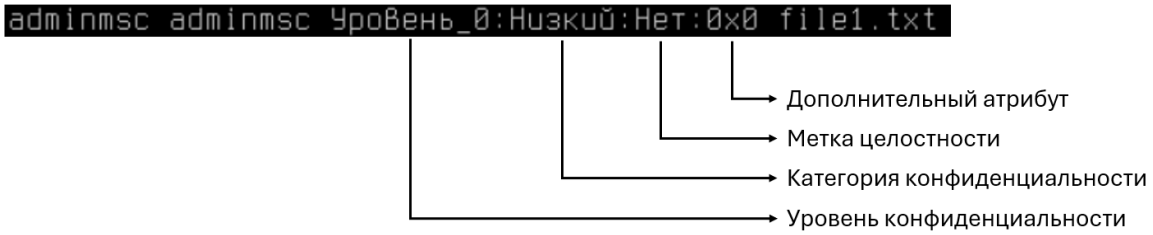
Для работы с мандатным доступом каждому объекту необходимо назначить метку безопасности (мандатная метка). Она включает в себя классификационную метку, состоящую из уровня и категории конфиденциальности и метку целостности. Также каждому объекту возможно назначить дополнительные атрибуты. Для просмотра меток конфиденциальности используется утилита *pdp-ls*.

Синтаксис команды приведен ниже

pdp-ls [опции] [файл_или_каталог]

- *файл_или_каталог* — название файла или каталога
- *опции* — флаги, передаваемые в утилиту
  - *-M* — данный флаг отображает метки конфиденциальности

Формат вывода представлен на рисунке ниже.



Обычные пользователи работают с низким уровнем целостности (0), администраторы системы должны работать с высоким уровнем целостности (63). Дочерний процесс полностью наследует метку безопасности родительского процесса. Когда процесс создает файл, то файл наследует только классификационную метку процесса. Файл получает нулевую метку целостности.

Дополнительные атрибуты представлены в таблице:

Атрибут	Объект	Описание
ccnr	Каталог	Каталог может содержать файлы и каталоги с различными классификационными метками, но не большими, чем его собственный
ehole	Файл	Файл, имеющий минимальную классификационную метку, игнорирует правила управления мандатным доступом к нему, процессы не могут прочитать данные, записанные в такие файлы (пример: /dev/null)
whole	Файл	Файл, имеющий максимальную классификационную метку, разрешает процессам, имеющим более низкую классификационную метку, записывать в них

Основные команды для работы с мандатными атрибутами:

userlev [опции]– просмотр и определение уровня конфиденциальности

- -d удалить уровень конфиденциальности
- -a добавить новый уровень конфиденциальности

- -r переименовать существующий уровень

usercat [опции] — просмотр и определение категорий конфиденциальности

- -d удалить категорию конфиденциальности
- -a добавить новую категорию конфиденциальности
- -r переименовать существующую категорию

pdpl-file [опции] [УР_КОНФ]:[УР_ЦЕЛ]:[КАТ_КОНФ]:[ФЛАГИ] [файл/каталог] — просмотр и задание меток и атрибутов безопасности файлам и каталогам.

Пример 1: выставить значение 0 уровня конфиденциальности, низкий уровень целостности и категорию 0.

```
pdpl-file 0:0:0 file
```

Пример 2: выставить значение 3 уровня конфиденциальности, низкого уровня целостности, категории 0 и атрибут ccnr.

```
pdpl-file 3:0:0:ccnr directory
```

pdpl-user – отображение и выдача метки безопасности пользователям

- -l – установка допустимых уровней конфиденциальности в виде: Минимальный:Максимальный
- -c – установка допустимых категорий конфиденциальности в виде: Минимальный:Максимальный
- -i установка максимального уровня целостности

pdp-id – вывод текущей метки безопасности процесса

2. Практическая часть

Пользователи соответствуют созданным в предыдущей лабораторной работе. Для простоты будем обозначать их user1, user2, user3. Основного пользователя системы будем называть adminstd.

2.1. Мандатное разграничение доступа

Задание 1.

1. Зайдите в систему администратором. Получите права root.
2. Переименуйте уровни конфиденциальности:
   0 — for_all
   1 — secret
   2 — very_secret
   3 — very_important
3. Настройте учетную запись для пользователя user1
   - минимальный уровень конфиденциальности — for_all
   - максимальный уровень конфиденциальности — very_secret
4. Настройте учетную запись для пользователя user2
   - минимальный уровень конфиденциальности — for_all
   - максимальный уровень конфиденциальности — secret

Задание 2.

1. Создайте каталог /home/project. Установите на каталог уровень конфиденциальности very_important и установите дополнительный атрибут ccnr.
2. Создайте каталог /home/project/secret. Установите на каталог уровень конфиденциальности secret.
3. Создайте каталог /home/project/very_secret. Установите на каталог уровень конфиденциальности very_secret.
4. Установите файловые списки управления доступом (ACL) и файловые списки управления доступом по умолчанию (default ACL) на каталоги /home/project/secret и /home/project/very_secret, позволяющие пользователям ivanov и petrov создавать и удалять файлы в этих каталогах и изменять содержимое созданных файлов.

Задание 3.

1. Зайдите в систему под учетной записью user1 с уровнем конфиденциальности secret.
2. Создайте файл file1.txt в каталоге /home/project/secret. В этот файл добавьте строку user2. Сохраните файл.
3. Удалось ли создать, изменить и сохранить файл file1.txt?
4. Виден ли каталог /tmp/project/very_secret?
5. Зайдите под учетной записью user1 в систему с уровнем конфиденциальности very_secret.
6. Создайте файл file2.txt в каталоге /tmp/project/very_secret. В этот файл добавьте строку “Hello!”. Сохраните файл.
7. Удалось ли создать, и изменить и сохранить файл file2.txt?
8. Виден ли каталог /tmp/project/secret?
9. Виден ли файл /tmp/project/secret/file1.txt?

Задание 4.

1. Добавьте в файл /tmp/project/secret/file1.txt строку “Hello2!”.
2. Удалось ли изменить содержимое этого файла?
3. Зайдите в систему под учетной записью пользователем user2 с уровнем конфиденциальности secret.
4. Добавьте в файл /home/project/secret/file1.txt строку user2.
5. Удалось ли изменить содержимое этого файла?
6. Можете ли Вы прочитать содержимое файла /tmp/project/very_secret/file2.txt?
7. Сделайте пользователя user2 администратором. Проверьте, что данный пользователь может выполнять команды от имени пользователя root.

Контрольные вопросы

1. В чем необходимость введения мандатного уровня доступа?
2. В чем отличие уровня от категории конфиденциальности?
3. Какие дополнительные атрибуты МД вы знаете?