**13 Лекция. Управление доступом**

**Основные модели управления доступом:**

* дискреционная
* мандатная
* ролевая

**Дискреционное управление доступом (*discretionary access control*, DAC)**

[http://ru.wikipedia.org/wiki/Дискреционное\_управление\_доступом](http://ru.wikipedia.org/wiki/%D0%94%D0%B8%D1%81%D0%BA%D1%80%D0%B5%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%BE%D0%B5_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_%D0%B4%D0%BE%D1%81%D1%82%D1%83%D0%BF%D0%BE%D0%BC)

управление доступом субъектов к объектам на основе списков управления доступом или матрицы доступа. Субъект с определенным правом доступа может передать это право любому другому субъекту.

На курсе "Операционные системы" вы работали с этой моделью доступа.

Пример: когда вы расписываете доступ к файлу, вы указываете

1. имя владельца файла (субъект)
2. права на чтение
3. права на запись
4. права на запуск на выполнение

Примеры субъектов:

* пользователь
* программа выполняющаяся под именем пользователя

Примеры объектов:

* файлы
* каталоги
* внешние накопители (CD,DVD,USB и т.д.)
* принтер
* сетевой адаптер

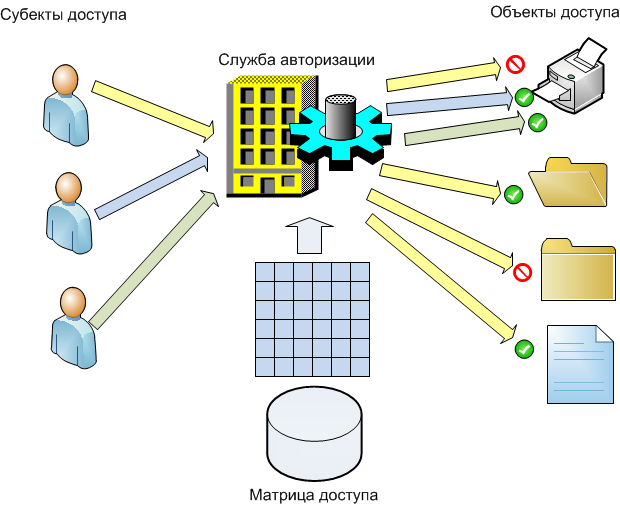


Рис. Дискреционное управление доступом

Достоинства:

* простата реализации
* гибкость (пользователь может описать доступ к своим ресурсам)

Недостатки:

* излишняя детализированность (приводит к запутанности)
* сложность администрирования
* пользователь может допустить ошибку при назначении прав

Пример дискреционного управления доступом к файлам в LINUX.

ls -l  
drwxr-xr-x. 2 root root 4096 янв. 30 18:37 anaconda  
drwxr-x---. 2 root root 4096 апр. 1 21:27 audit  
-rw-r--r--. 1 root root 12094 апр. 2 03:33 boot.log  
-rw-------. 1 root utmp 384 апр. 2 15:25 btmp  
-rw-------. 1 root utmp 1536 марта 15 07:41 btmp-20120401  
-rw--w----. 1 root 997 0 янв. 30 18:35 clamav-milter.log  
drwxr-xr-x. 2 root root 4096 янв. 30 14:44 ConsoleKit  
-rw-r--r--. 1 root root 267059 апр. 2 15:25 cron  
-rw-r--r--. 1 root root 1241791 апр. 1 03:22 cron-20120401  
-r--------. 1 root root 94710 марта 13 12:51 dracut.log-20120314  
drwx------. 2 root root 12288 апр. 1 03:22 httpd  
drwxr-xr-x. 2 root root 4096 февр. 6 12:26 iptraf-ng  
-rw-r--r--. 1 root root 292000 апр. 2 15:25 lastlog  
drwxr-xr-x. 2 root root 4096 янв. 30 18:34 mail  
-rw-r-----. 1 mysql mysql 3277 апр. 2 12:04 mysqld.log  
-rw-r-----. 1 mysql mysql 587151 февр. 17 10:18 mysqld.log-20120217  
drwxr-x---. 4 nagios nagios 4096 апр. 2 15:25 nagios  
drwxr-xr-x. 2 ntp ntp 4096 окт. 6 19:38 ntpstats  
drwx------. 2 root root 4096 июня 2 2011 ppp  
drwxr-xr-x. 2 root root 4096 янв. 30 16:44 prelink  
drwxr-x---. 2 squid squid 4096 апр. 1 03:22 squid  
drwxr-x---. 2 root root 4096 дек. 20 01:24 sssd  
-rw-rw-r--. 1 root utmp 112128 апр. 2 15:25 wtmp  
-rw-------. 1 root root 12381 марта 13 17:05 yum.log

**Мандатное управление доступом (Mandatory access control, MAC)**

[http://ru.wikipedia.org/wiki/Мандатное\_управление\_доступом](http://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D0%BD%D0%B4%D0%B0%D1%82%D0%BD%D0%BE%D0%B5_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_%D0%B4%D0%BE%D1%81%D1%82%D1%83%D0%BF%D0%BE%D0%BC)

Разграничение доступа субъектов к объектам, основанное на назначении метки (мандата) конфиденциальности для информации, содержащейся в объектах, и выдаче официальных разрешений (допуска) субъектам на обращение к информации такого уровня конфиденциальности.

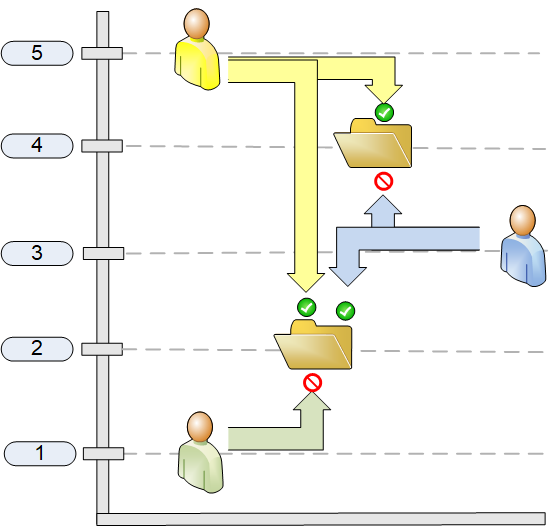


Рис. Мандатное управление доступом

Достоинства:

* простата построения общей схемы доступа
* простата администрирования
* пользователь не может назначать доступ к объекту (в некоторых случаях)

Недостатки:

* проблема разграничения пользователей одного уровня
* пользователь не может назначать доступ к объекту  (в некоторых случаях)

Мандатная модель не реализована в Windows, но можно поставить дополнительные средства защиты (например: [Secret Net](http://www.securitycode.ru/products/secret_net/), [Аккорд](http://www.accord.ru/acwin32.html) и т.д.).

В Linux система встроена на уровне ядра - SELinux (Security-Enhanced Linux — Linux с улучшенной безопасностью - <http://ru.wikipedia.org/wiki/SELinux>) .

**Ролевое управление доступом (Role Based Access Control, RBAC)**

[http://ru.wikipedia.org/wiki/Ролевое\_управление\_доступом](http://ru.wikipedia.org/wiki/%D0%A0%D0%BE%D0%BB%D0%B5%D0%B2%D0%BE%D0%B5_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_%D0%B4%D0%BE%D1%81%D1%82%D1%83%D0%BF%D0%BE%D0%BC)

развитие политики избирательного управления доступом, при этом права доступа субъектов системы на объекты группируются с учетом специфики их применения, образуя роли.

В модели присутствуют:

* пользователи
* роли
* объекты

Рассмотрим модель на примере MOODLE.

Роли в MOODLE:

* **Администратор** - все права
* **Manager** - управляющий, может создавать и изменять курсы.
* **Создатели курса** - может создать курс и преподавать в нем.
* **Преподаватель**  - может делать все в курсе (но не может создать курс).
* **Non-editing teacher** - (ассистент) может преподавать, но не может изменять курс.
* **Студент** - может обучатся на курсе (пользователь записанный на курс).
* **Guest** -  гость, пользователи без авторизации.
* **Authenticated user** - пользователь прошедшие аутентификацию (но не записанный на курс).

Роли могут быть глобальными или "локальными".

Типы контекста, где роли могут быть назначены:

* **Manager** - Система, Категория курсов, Курс.
* **Создатели курса** - Система, Категория курсов.
* **Преподаватель**  - Курс, Модуль элемента курса.
* **Non-editing teacher** - Курс, Модуль элемента курса.
* **Студент** - Курс, Модуль элемента курса.
* **Guest** - нигде.
* **Authenticated user** - нигде.