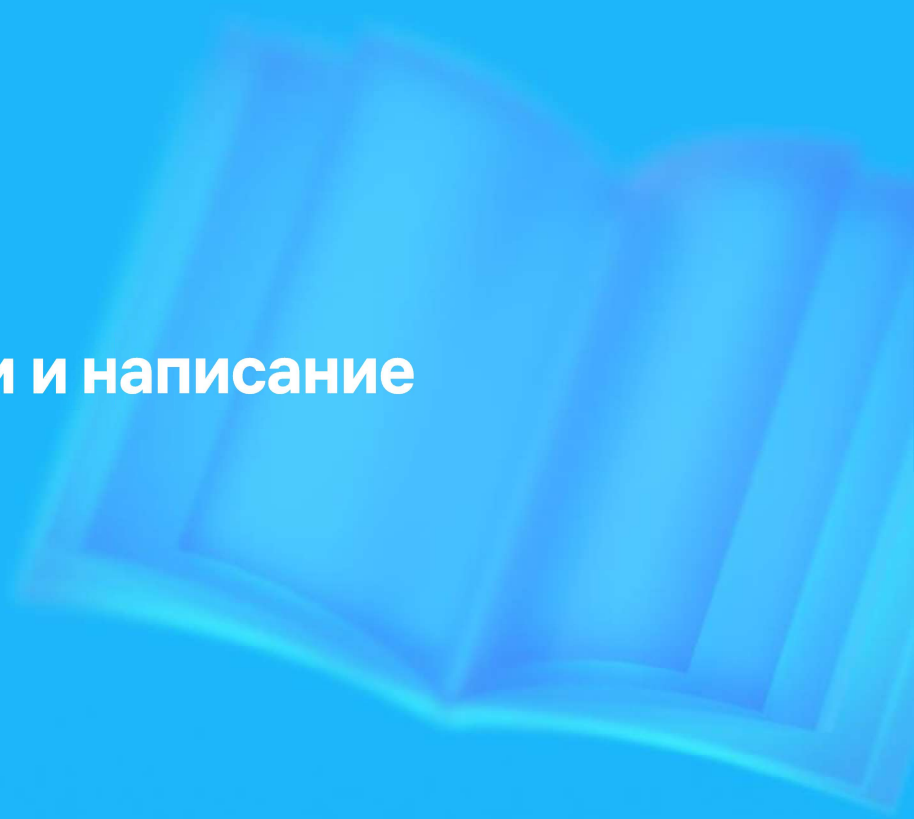




## **Вебинар №3. Работа с файлами и написание скриптов**



## Что будет сегодня



Узнаете о работе  
с файлами в ОС Astra  
Linux



Научитесь работать  
с текстовой  
информацией в ОС Astra  
Linux



Научитесь управлять  
дискреционным доступом



Узнаете о сценариях  
bash и научитесь их  
создавать



# Работа с файлами в ОС Astra Linux



## Типы файлов

Файлы бывают следующих типов:

- ( - ) Обычный файл.
- ( d ) Директория — это объект файловой системы, который упрощает работу с файлами, позволяя группировать их.
- ( l ) Символическая ссылка.
- ( b ) Файл блочного устройства.
- ( c ) Файл символьного устройства.
- ( p ) Именованные каналы.
- ( s ) Сокет.
- ( n ) Сетевой файл.

Тип файла можно увидеть в первом символе вывода `ls -l`.

```
drwxr-xr-x  2 root root    4096 мая 27 11:06 rcS.d
-rw-r--r--  1 root root      55 мая 28 12:45 resolv.conf
lrwxrwxrwx  1 root root      13 апр 23  2019 rmt -> /usr/sbin/rmt
-rw-r--r--  1 root root      887 апр 10  2018 ssh
```

## Назначение основных каталогов

В Linux существуют специальные директории, которые имеют особое назначение:

**/proc:** в этой директории хранится информация о текущих процессах и состоянии системы.

**/sys:** в этой директории хранится информация о конфигурации системы и устройствах.

**/tmp:** в этой директории хранятся временные файлы, которые создаются при работе программ.

**/var:** в этой директории хранятся переменные данные, такие как логи, кэши и т. д.

Каждый файл и директория в Linux имеет свой уникальный путь от корневой директории. Например, путь к файлу `passwd` будет выглядеть так: `/etc/passwd`.

```
admuser@astra:~$ cd /
admuser@astra:/$ ls -l
итого 58
lrwxrwxrwx 1 root root 7 мая 24 14:06 bin -> usr/bin
drwxr-xr-x 4 root root 1024 мая 24 14:18 boot
drwxr-xr-x 20 root root 4120 мая 28 12:45 dev
drwxr-xr-x 99 root root 4096 мая 28 13:33 etc
drwxr-xr-x 4 root root 4096 мая 24 14:17 home
lrwxrwxrwx 1 root root 32 мая 24 14:07 initrd.img -> boot/initrd.img-5.4.0-54-generic
lrwxrwxrwx 1 root root 32 мая 24 14:07 initrd.img.old -> boot/initrd.img-5.4.0-54-generic
lrwxrwxrwx 1 root root 7 мая 24 14:06 lib -> usr/lib
lrwxrwxrwx 1 root root 9 мая 24 14:06 lib32 -> usr/lib32
lrwxrwxrwx 1 root root 9 мая 24 14:06 lib64 -> usr/lib64
lrwxrwxrwx 1 root root 10 мая 24 14:06 libx32 -> usr/libx32
drwx----- 2 root root 16384 мая 24 14:06 lost+found
drwxr-xr-x 3 root root 4096 мая 24 14:06 media
drwxr-xr-x 2 root root 4096 июл 28 2015 mnt
drwxr-xr-x 2 root root 4096 мая 24 14:06 opt
drwxr-xr-x 2 root root 0 мая 28 12:45 parsecfs
dr-xr-xr-x 121 root root 0 мая 28 12:45 proc
drwxr-xr-x 2 root root 4096 мая 27 11:32 root
drwxr-xr-x 24 root root 700 мая 28 12:45 run
lrwxrwxrwx 1 root root 8 мая 24 14:06 sbin -> usr/sbin
drwxr-xr-x 2 root root 4096 мая 24 14:06 srv
dr-xr-xr-x 14 root root 0 мая 28 12:45 sys
drwxrwxrwt 7 root root 4096 мая 28 13:02 tmp
drwxr-xr-x 14 root root 4096 мая 24 14:06 usr
drwxr-xr-x 12 root root 4096 мая 24 14:10 var
lrwxrwxrwx 1 root root 29 мая 24 14:07 vmlinuz -> boot/vmlinuz-5.4.0-54-generic
lrwxrwxrwx 1 root root 29 мая 24 14:07 vmlinuz.old -> boot/vmlinuz-5.4.0-54-generic
admuser@astra:/$
```

Команда `cd` используется для перехода в другой каталог.

Команда `pwd` показывает текущий каталог.

Команда `ls` показывает список файлов в текущем каталоге.

Команда `mkdir` создаёт новый каталог.

Команда `rmdir` удаляет пустой каталог.

Лайфхаки:

- `cd ~` — переход в домашнюю директорию пользователя, под которым выполняется команда.  
`cd ..` — переход в предыдущую директорию относительно текущей.
- Чтобы создать вложенный каталог внутри создаваемого, используйте ключ `-p`, а чтобы создать несколько — фигурные скобки с перечислением через запятую: `{...}`

Например, создадим каталоги `test1`, `test2` и `test3` внутри каталога: `mkdir -p /home/admuser/data/{test1,test2,test3}`

```
admuser@astra:~$ ls
test1.txt
admuser@astra:~$ mkdir -p /home/admuser/data/{test1,test2,test3}
admuser@astra:~$ ls
data  test1.txt
admuser@astra:~$ ls data/
test1  test2  test3
admuser@astra:~$
```

## Операции с файлами

Команда `cp` копирует файлы из одного каталога в другой.

```
admuser@astra:~$ cp test1.txt data/test1
admuser@astra:~$ ls data/test1
test1.txt
admuser@astra:~$
```

Команда `mv` перемещает файлы из одного каталога в другой.

```
admuser@astra:~$ ls data/test1
test1.txt
admuser@astra:~$ ls data/test2
admuser@astra:~$ mv data/test1/test1.txt data/test2/
admuser@astra:~$ ls data/test1
admuser@astra:~$ ls data/test2
test1.txt
admuser@astra:~$
```

Команда `rm` удаляет файлы.

## Поиск файлов

Команда `find` используется для поиска файлов по различным критериям, таким как имя файла, размер, дата изменения и т. д.

Найдём наш файл `test1.txt` командой `find /home -type f -name "test1.txt"`

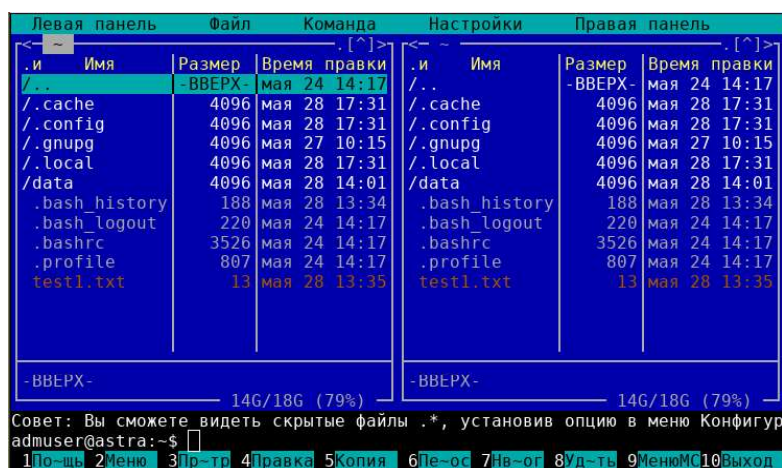
```
admuser@astra:~$ find /home -type f -name "test1.txt"
/home/admuser/test1.txt
/home/admuser/data/test2/test1.txt
admuser@astra:~$
```

Команда `grep` используется для поиска текста в файлах.



## Использование Менеджера файлов и Midnight Commander для работы с файлами и каталогами

Midnight Commander — это текстовый менеджер файлов, который позволяет управлять файлами и каталогами из командной строки. Он имеет мощные возможности поиска и фильтрации файлов, а также поддерживает работу с архивами.



The screenshot shows the Midnight Commander (MC) interface. It consists of two main panels, 'Левая панель' (Left panel) and 'Правая панель' (Right panel), each displaying a list of files and directories. The files listed include hidden files like .cache, .config, .gnupg, .local, .data, .bash\_history, .bash\_logout, .bashrc, .profile, and test1.txt. The interface also shows a 'Команда' (Command) field and a 'Настройки' (Settings) menu. At the bottom, there is a status bar with a disk usage indicator (14G/18G (79%)) and a prompt 'Совет: Вы сможете видеть скрытые файлы .\*, установив опцию в меню Конфигур' (Tip: You will be able to see hidden files .\*, by setting the option in the Configuration menu). A legend at the very bottom explains the function keys: 1Поиск (Search), 2Меню (Menu), 3Пр-тр (Preferences), 4Правка (Edit), 5Копия (Copy), 6Пе-ос (Paste), 7Нв-ог (New), 8Уд-ть (Delete), 9МенюMC (MC Menu), 10Выход (Exit).

Левая панель	Файл	Команда	Настройки	Правая панель			
.и	Имя	Размер	Время правки	.и	Имя	Размер	Время правки
./	ВВЕРХ	мая 24 14:17	./	ВВЕРХ	мая 24 14:17	./	ВВЕРХ
./cache	4096	мая 28 17:31	./cache	4096	мая 28 17:31	./cache	4096
./config	4096	мая 28 17:31	./config	4096	мая 28 17:31	./config	4096
./gnupg	4096	мая 27 10:15	./gnupg	4096	мая 27 10:15	./gnupg	4096
./local	4096	мая 28 17:31	./local	4096	мая 28 17:31	./local	4096
/data	4096	мая 28 14:01	/data	4096	мая 28 14:01	/data	4096
.bash_history	188	мая 28 13:34	.bash_history	188	мая 28 13:34	.bash_history	188
.bash_logout	220	мая 24 14:17	.bash_logout	220	мая 24 14:17	.bash_logout	220
.bashrc	3526	мая 24 14:17	.bashrc	3526	мая 24 14:17	.bashrc	3526
.profile	807	мая 24 14:17	.profile	807	мая 24 14:17	.profile	807
test1.txt	13	мая 28 13:35	test1.txt	13	мая 28 13:35	test1.txt	13

Совет: Вы сможете видеть скрытые файлы .\*, установив опцию в меню Конфигур

1Поиск 2Меню 3Пр-тр 4Правка 5Копия 6Пе-ос 7Нв-ог 8Уд-ть 9МенюMC 10Выход



# Работа с текстовой информацией в ОС Astra Linux



## Перенаправление стандартных потоков в файл или из файла

Перенаправление стандартных потоков в файл или из файла — это механизм в Linux, который позволяет перенаправлять вывод различных команд и приложений в файл, а также читать ввод из файла вместо терминала.

Основные примеры перенаправления в файл и из файла:

- Перенаправление стандартного вывода в файл.
- Перенаправление стандартного ввода из файла.
- Комбинированное перенаправление.
- Перенаправление ошибок в файл.
- Перенаправление вывода и ошибок в разные файлы.
- Перенаправление вывода и ошибок в один файл.

## Перенаправление стандартных потоков между процессами

Перенаправление стандартных потоков между процессами в Linux позволяет направлять вывод одного процесса на вход другого, тем самым образуя цепочку работающих вместе процессов.

Канал «|».

Канал «>».

Комбинированный канал «&>».

Разработка конвейеров.

Фоновый режим.

## Команды для текстовых файлов в Linux

Команда `cat`

Команда `less`

Команда `more`

Команда `head`

Команда `tail`

Текстовый редактор `vim` или `nano`

## Команды-фильтры в Linux

Команды-фильтры в Linux выполняются для обработки текстовых данных, часто используются в конвейере для фильтрации вывода команды или для преобразования данных в соответствии с определёнными критериями.

grep

sed

awk

sort

uniq

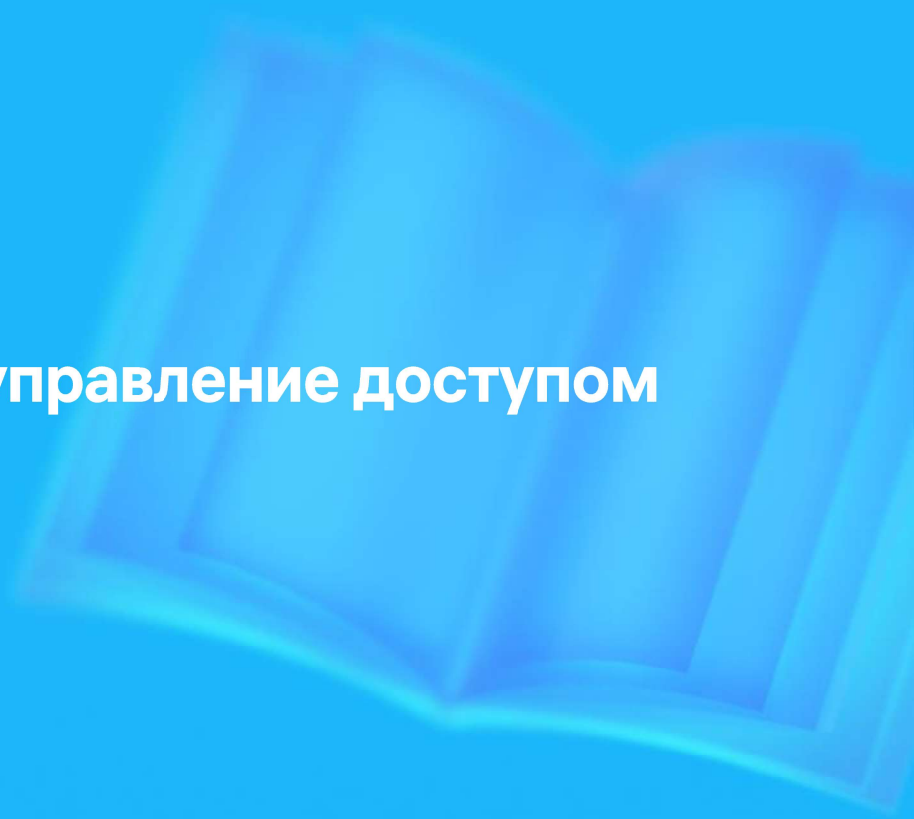
## Регулярные выражения

Регулярные выражения (Regular Expressions) в Linux используются для поиска и замены текста в файле или выводе команды в терминале. Регулярные выражения представляют собой шаблоны поиска, в которых используются специальные символы, предоставляющие множество вариантов поиска.

Основные символы регулярных выражений:

- `.` — любой 1 символ, кроме новой строки.
- `*` — любое число символов, включая 0.
- `+` — 1 или более повторений последнего символа.
- `?` — 0 или 1 повторение последнего символа.
- `[]` — символы в квадратных скобках представляют собой любой из символов в скобках. Например, `[abc]` ищет любой из символов `a`, `b`, или `c`.
- `^` — начало строки.
- `$` — конец строки.

## Дискреционное управление доступом





## Символьная и числовая формы записи прав доступа

Возможны следующие сочетания прав:

- 7 (rwx) — права на чтение, запись и выполнение.
- 6 (rw-) — права на чтение и запись.
- 5 (r-x) — права на чтение и выполнение.
- 4 (r--) — права на чтение.
- 3 (-wx) — права на запись и выполнение.
- 2 (-w-) — права на запись.
- 1 (--x) — права на выполнение.
- 0 (---) — нет прав доступа.

## Установка прав в символьной форме

```
sudo chmod u=rwx,g=rx,o= file.txt
```

- u=rwx означает, что владелец получает права на чтение, запись и выполнение.

- g=rx означает, что группа получает права на чтение и выполнение.

- o= означает обусловливание правами доступа для всех остальных пользователей, в этом случае они не имеют прав.

## Расширенные права доступа (ACL)

Для просмотра дополнительных разрешений используется команда `getfacl`.

`Setfacl` — для установки доп. разрешений.

`$ setfacl -m u:max:rw /tmp/test5/file1 #` — пользователь, который используется в этой команде, уже должен быть в системе.

Опция `-m` заставляет команду `setfacl` модифицировать права на файл:

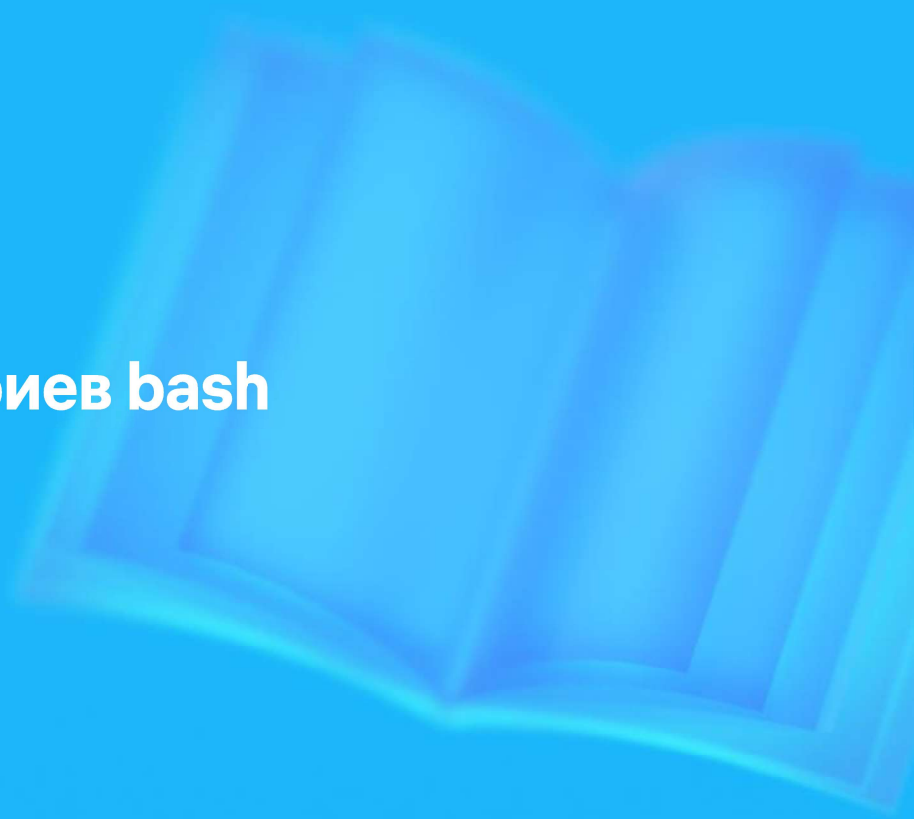
`$ setfacl -m g:max:rw /tmp/test5/file1`

С помощью ACL мы даже заранее можем назначить права по умолчанию для новых файлов.

Назначим права RW по умолчанию для вновь создаваемых файлов в директории для пользователя `max`:

`$ sudo setfacl -m default:u:max:rw /tmp/data.`

## Создание сценариев bash



```
#!/bin/bash
```

```
echo "Hello, World!"
```

```
#!/bin/bash
```

```
echo "Hello, World!"
```

### **Shebang -**

последовательность из символов решётки и восклицательного знака ("#!") в начале файла скрипта.

## Объявление переменных

- Без пробелов: `var="значение"`
- Использование: `$var` или `${var}`
- Экспорт: `export var` (для дочерних процессов)
- `$HOME`, `$PATH`, `$USER` – переменные окружения
- `$$` — PID процесса

## Условия if-else

```
if [ условие ]; then
    # код
elif [ условие ]; then
    # код
else
    # код
fi
```



## Сравнения

- Числа: -eq, -ne, -gt, -lt
- Строки: ==, !=
- Файлы: -f, -d, -e, -nt, -ot



## Синтаксис switch-case

```
case $var in
    "вариант1") команды ;;
    "вариант2") команды ;;
    *) default ;;
esac
```



## Синтаксис for

```
for итератор in список  
do  
    команды  
done
```



## Синтаксис while

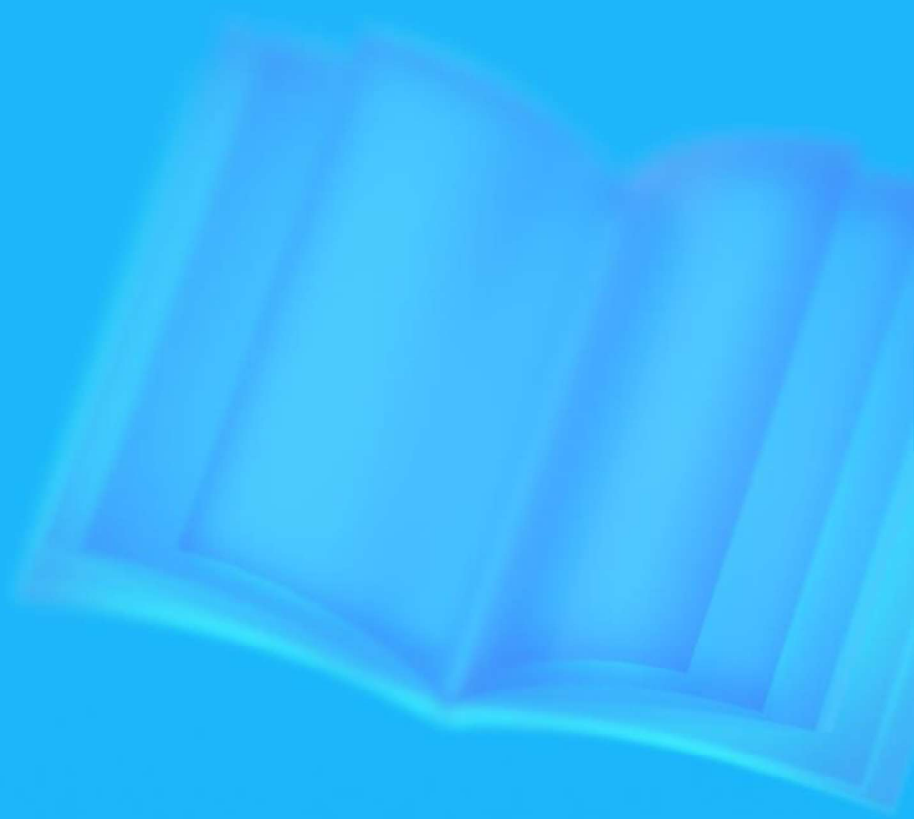
```
while [ условие ]; do
    команды
done
```



## Синтаксис until

```
until [ условие ]; do  
    команды  
done
```

## Вопросы



**Спасибо за внимание!**

