**Содержание**

[**1. Цели работы** 2](#_Toc86776671)

[**2. Задачи** 2](#_Toc86776672)

[**3. Ход выполнения работы** 3](#_Toc86776673)

[**4. Вывод** 9](#_Toc86776674)

# **Лабораторная работа №5 – Учетные записи. Фоновый и диалоговый режимы исполнения процессов**

# **1. Цели работы**

Манипуляции с правами доступа при создании в системе учетных записей и исследование влияния прав на файловые операции. Изучение специфик фонового (background) и диалогового (foreground) режимов исполнения процессов и способов переключений между этими режимами.

**2. Задачи**

1. Создать несколько учетных записей для нескольких пользователей (не задавая им прав администратора) и объедините их в две группы. Заходя в систему под разными аккаунтами, создайте в соответствующих домашних каталогах файлы, варьируя при этом права доступа для пользователя, для группы, для всех. Убедиться, что права доступа разделяются в соответствии с тем, как это задано. Провести операцию слияния файлов с различными правами доступа и проверить, какие при этом получаются права у результирующего файла.

2. Запустить в фоновом режиме командный файл (процесс), выдающий в цикле с некоторой задержкой сообщение на консоль. Запустить другой командный файл (процесс), требующий диалога, в обычном режиме. Убедиться, что вывод двух этих процессов на консоль перемежается. Остановить фоновый процесс сигналом kill. Запустить его снова, организовав предварительно перенаправление его вывода в файл. Убедиться, что теперь вывод двух процессов разделен.

3. Доработать предыдущее задание так, чтобы показать возможность перевода фонового процесса в диалоговый режим для выполнения операции ввода с клавиатуры и затем возврата его обратно в фоновый режим. Продемонстрировать возможность оставления фонового процесса на исполнение после завершения пользовательского сеанса работы в ОС.

4. Разработать командный файл для выполнения архивации каталога через определенные интервалы времени. Запустить командный файл в режиме background. Имя архивируемого каталога, местоположение архива и время архивации передаются при запуске командного файла в виде параметров

**3. Ход выполнения работы**

1. Создам несколько учетных записей для нескольких пользователей (не задавая им прав администратора) и объединю их в две группы. На рисунке 5.1 представлено сколько пользователей имеется на данный момент. Создам два пользователя user1 и user2. Создание приведено на рисунке 5.2. Задам пользователям пароли, операция приведена на рисунке 5.3. Список обновленных пользователей приведен на рисунке 5.3. Также было создано две группы и них добавлены пользователи, результат приведен на рисунке 5.4. Под пользователями теперь можно авторизоваться. Выбор пользователей приведен на рисунке 5.5. Зайду под каждым новым пользователем и создам текстовый файл в домашней директории.

|  |
| --- |
| Рис. 5.1 – Выполнение команды ls -l /home |
| Рис. 5.2 – Создание двух пользователей и задание им пароля |
| Рис. 5.3 – Обновленный список пользователей |
| Рис. 5.4 – Создание двух групп и добавление туда пользователей |
| Рис. 5.5 – Выбор пользователя |

Зайду в директории созданных пользователей и посмотрю все файлы, которые содержит каталог новых пользователей. Результат выполнения приведен на рисунке 5.6.

|  |
| --- |
| Рис. 5.6 Просмотр директорий других пользователей |

Из полученных результатов видно, что у файлов, который были созданы другими пользователями, нет флагов для модификации файла. Записана комбинация -rw-rw-r--, которая говорит, что пользователь и его группа может редактировать и читать файл, а остальные только read only. Проверю, что будет при слиянии файлов с разными правами доступа. Для этого выполню команду cat слияния двух файлов и помещу полученный файл в директорию оригинального пользователя. Результаты приведены на рисунке 5.7. Зайду в каталог пользователя и проверю права доступа, изображение приведено на рисунке 5.8.

|  |
| --- |
| Рис. 5.7 – Слияние двух файлов user1 и user2 |
| Рис. 5.8 – Права доступа к файлу file.txt |

Видно, что редактировать файл может пользователь и его группа, поэтому при слиянии двух файлов, права на редактирование получает пользователь, который выполнил слияние.

2. Создам два командных файла, который приведены в листингах 5.1 и 5.2, которые выводят сообщения в терминал. Запущу первый файл в фоновом режиме. Второй же файл запущу в режиме foreground. Результаты вывода в терминал приведены на рисунке 5.9. По полученным данным видно, что данные перемежаются. Остановлю фоновый процесс сигналом kill c соответствующим параметром. Модифицирую файл, чтобы запись была перенаправлена в определенный файл. Для этого достаточно указать то, куда будут записываться строки: echo “информация”>>text.txt. запуск программ в фоновом режиме приведен на рисунке 5.10. Содержимое текстовых файлов представлено на рисунках 5.11 И 5.12. Из полученных результатов можно сделать вывод, что вывод двух файлов разделен.

|  |
| --- |
| Листинг 5.1 – script1.sh |
| #!/bin/bash  COUNTER=0  echo "its the first script"  while [ $COUNTER -lt 1000 ]  do  echo "in first script"  sleep 10  COUNTER=$(($COUNTER+1))  done |

|  |
| --- |
| Листинг 5.2 – script2.sh |
| #!/bin/bash  COUNTER=0  echo "its the second script"  while [ $COUNTER -lt 1000 ]  do  echo "in second script"  sleep 10  COUNTER=$(($COUNTER+1))  done |

|  |
| --- |
| Рис. 5.9 – Результаты работы двух командных файлов |
| Рис. 5.10 – Запуск в фоновом режиме модифицированных файлов |

|  |
| --- |
| Рис. 5.11 – Содержимое файла text1.txt через некоторое время |
| Рис. 5.12 – Содержимое файла text2.txt через некоторое время |

3. Доработаю предыдущее задание так, чтобы показать возможность перевода фонового процесса в диалоговый режим для выполнения операции ввода с клавиатуры и затем возврата его обратно в фоновый режим. Для этого необходимо, чтобы командный файл запрашивал информацию у пользователя. Для этого пропишу в файле дополнительно echo -n “Введите информацию” и read info, а потом выведу через echo “$info”. Результат выполнения в диалоговом режиме представлен на рисунке 5.13.

|  |
| --- |
| Рис. 5.13 – Результат выполнения script3.sh с введением информации |

Для тестирования запущу процесс в фоновом режиме и переведу его в диалоговый для тестирования ввода информации. Результаты приведены на рисунке 5.14. Сначала запускаю процесс в диалоговом режиме. Проверяю работоспособность, перевожу процесс в фоновый режим, восстанавливаю его и проверяю работоспособность, можно сказать, что процесс можно перевести обратно в диалоговый режим и продолжить вводить информацию с клавиатуры.

|  |
| --- |
| Рис. 5.14 – Тестирование ввода при возвращении в диалоговый режим |

Протестирую ситуацию, когда пользователь завершает сеанс. Для этого запущу процесс, переведу его в фоновый режим, завершу сеанс пользователя, при повторном входе проверю работоспособность процесса. При вызове команды jobs не было отображено каких-либо процессов, а это означает, что при завершении сеанса или закрытии терминала фоновые процессы завершаются. Важно отметить, что данную проблему можно решить с помощью создания процессов, используя tmux, так как сеансы tmux являются постоянными и поэтому даже при закрытии терминала программы продолжают работать. Важно уточнить, что при смене пользователя процесс продолжает свою работу.

5. Необходимо разработать командный файл, который архивирует каталог через определенные интервалы времени. Листинг кода приведен в листинге 5.3. Запущу командный файл в фоновом режиме. Результаты приведены на рисунке 5.15. Наличие архивированного каталога приведено на рисунке 5.16. По полученным результатам видно, что фоновый процесс работает, tar успешно архивирует нужный каталог и выводит сообщение о том, какие файлы были помещены в архив. Пользователь третьим параметром задает период архивации. Важно отметить, что архив либо создается, если в папке назначения не было архива с тем же именем, либо перезаписывает сам себя.

|  |
| --- |
| Листинг 5.3 – Содержимое файла scriptForArchive.sh |
| #!/bin/bash  isWorking=1  echo "start to archive"  name\_of\_catalogue=$1  way\_to\_archive=$2  period=$3  while [ $isWorking ]  do  cd ~  zip -r archive.zip $1  mv archive.zip $2  sleep $3  done |

|  |
| --- |
| Рис. 5.15 – Тестирование архивации фоновым процессом |
| Рис. 5.16 – Архивированный каталог |

**4. Вывод**

В ходе выполнения работы были получены навыки создания пользователей, определения прав к файлам, также были определены права файла, который был получен при слиянии нескольких файлов с разными правами доступа. Также в лабораторной работе были написаны командные файлы, которые помогли разобраться с фоновыми и диалоговыми режимами работы. При этом был создан скрипт, который архивирует каталог с определенной периодичностью и протестирован на реальном примере.