Отчёт по лабораторной работе №5

Управление системными службами

Максат Хемраев

Содержание

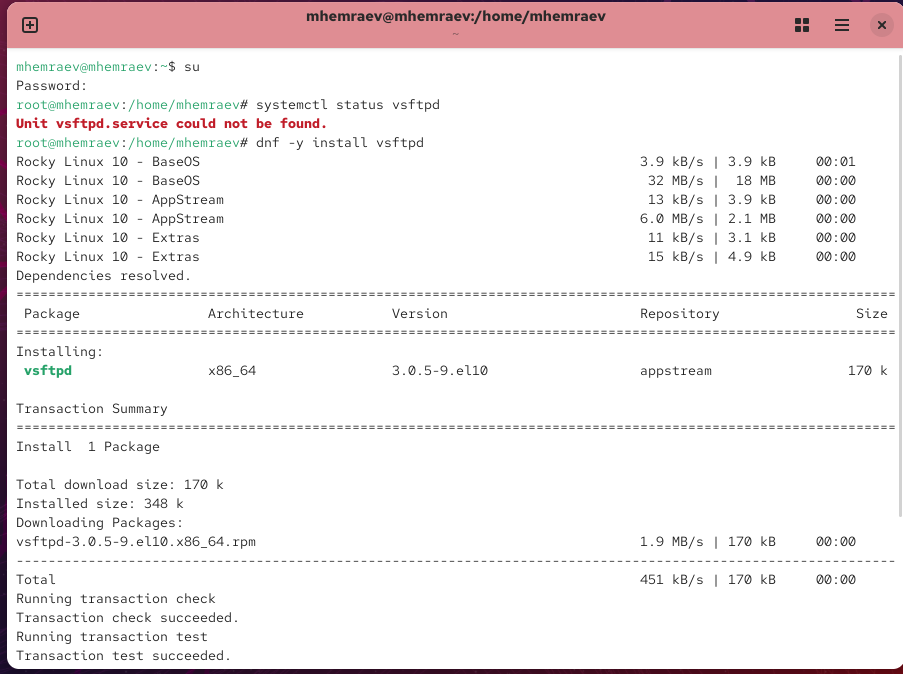
# 1 Цель работы

Получить навыки управления системными службами операционной системы посредством systemd.

# 2 Отчёт по выполнению работы

## 2.1 Управление сервисом *vsftpd*

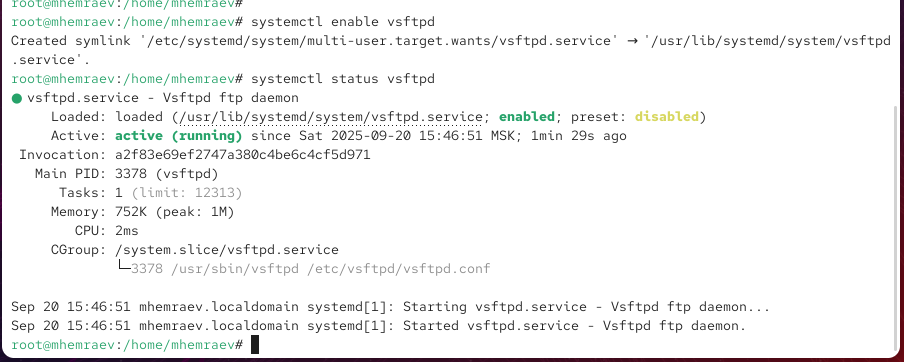
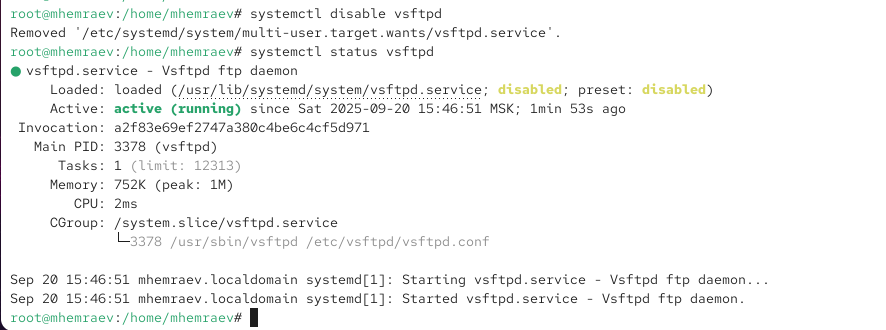
1. Получил полномочия администратора и проверил наличие службы **vsftpd**.  
   Система сообщила, что сервис не найден, так как пакет ещё не установлен.

* 
* Рис. 1: Проверка статуса службы vsftpd

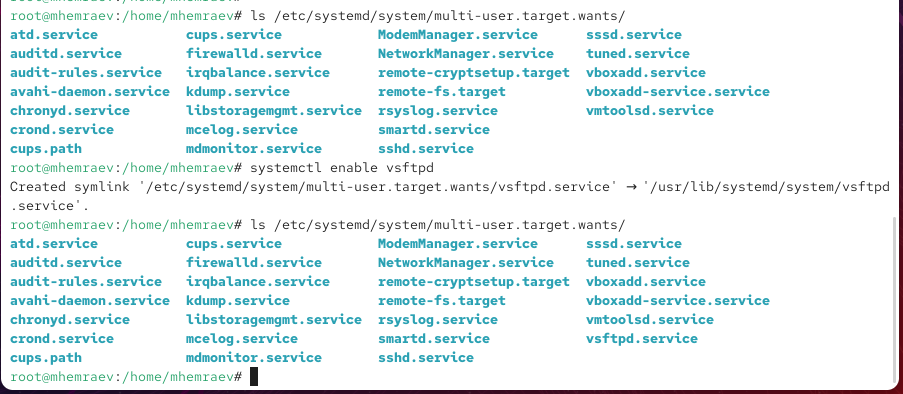
1. Установил пакет **vsftpd** с помощью менеджера пакетов **dnf**. Загрузка и установка прошли успешно.
2. Запустил службу **vsftpd** и повторно проверил её статус. Теперь сервис работает, но пока отключён в автозапуске.

* 
* Рис. 2: Запуск и проверка службы vsftpd

1. Добавил службу в автозапуск при старте системы и убедился, что её состояние изменилось на **enabled**. После этого удалил службу из автозапуска, а затем снова проверил её статус — он вернулся в состояние **disabled**.

*   
  

1. Вывел список символических ссылок, ответственных за запуск сервисов. В списке отсутствовала ссылка на **vsftpd**. После повторного включения автозапуска ссылка появилась, что подтверждает корректное создание символьной связи на системный юнит.

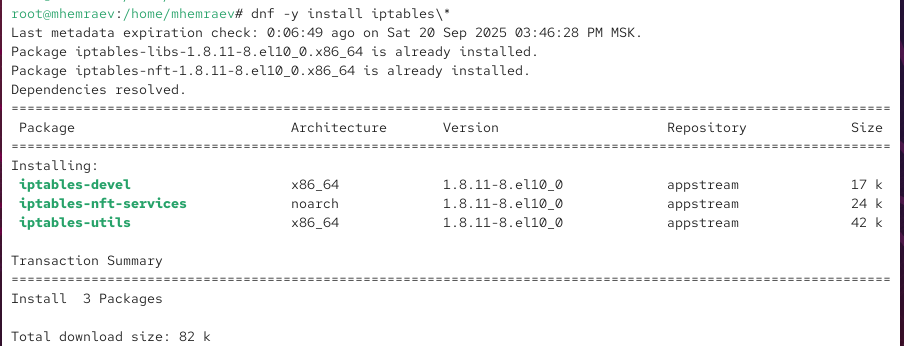
* 
* Рис. 3: Символические ссылки до и после включения автозапуска

1. Проверил список зависимостей службы **vsftpd**, а также список юнитов, зависящих от неё. Было показано, что сервис запускается в рамках цели **multi-user.target**, которая в свою очередь входит в **graphical.target**.

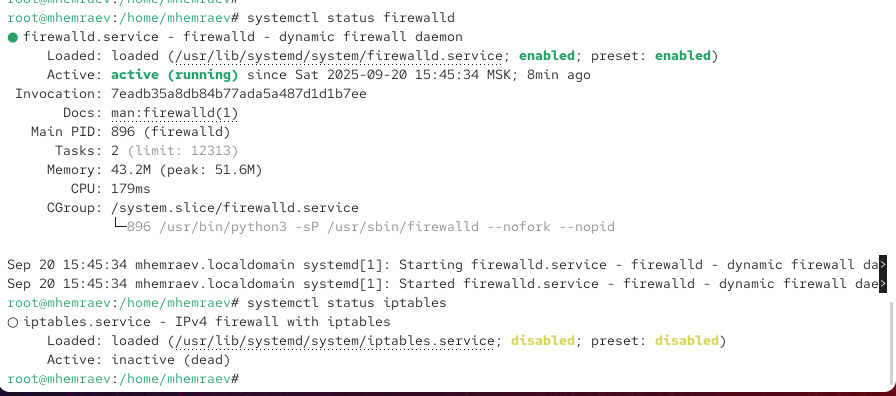
* 
* Рис. 4: Зависимости службы vsftpd

## 2.2 Конфликты юнитов: пример *iptables* и *firewalld*

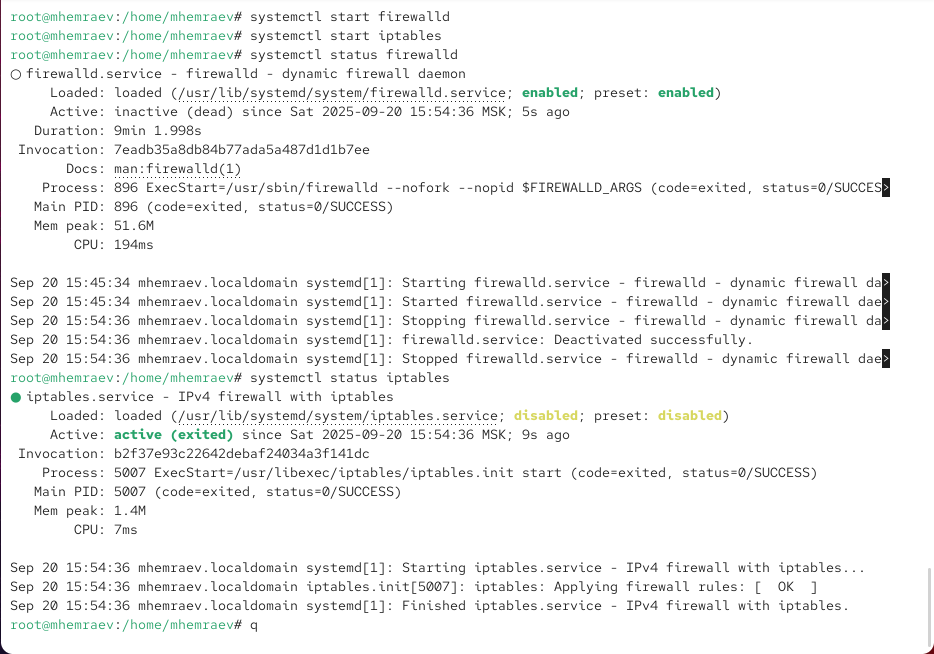
1. Получил полномочия администратора и установил пакет **iptables** вместе с дополнительными модулями. Установка завершилась успешно.

* 
* Рис. 5: Установка пакета iptables

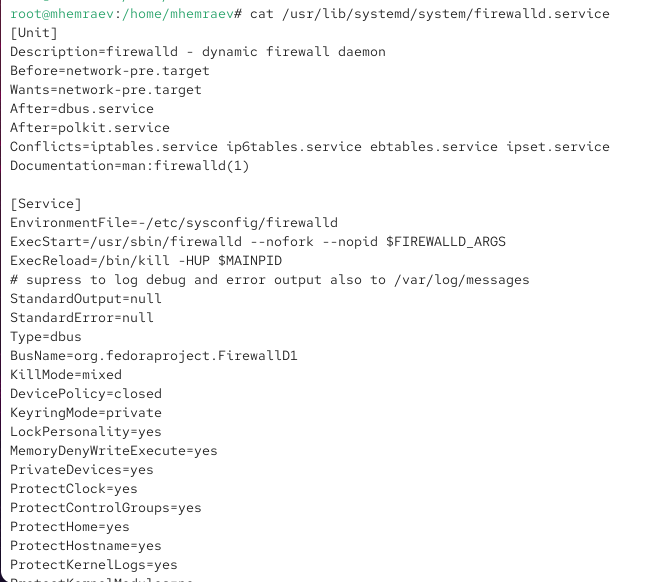
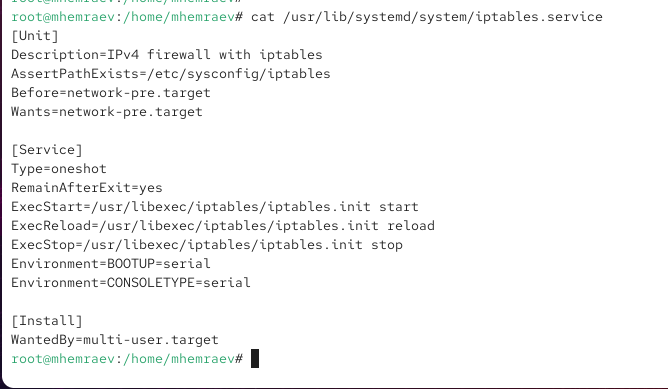
1. Проверил статус служб **firewalld** и **iptables**.  
   Вывод показал, что **firewalld** работает и активирован в автозагрузке, а **iptables** установлен, но неактивен.

* 
* Рис. 6: Проверка статуса firewalld и iptables

1. Попробовал запустить службы. При запуске **firewalld** сервис **iptables** оставался отключённым. При старте **iptables** — напротив, **firewalld** останавливался. Это подтверждает конфликт между двумя сервисами.

* 
* Рис. 7: Запуск и остановка конфликтующих сервисов

1. Ознакомился с содержимым юнитов.
   * В файле **firewalld.service** обнаружена строка *Conflicts=iptables.service*, указывающая на несовместимость с iptables.
   * В файле **iptables.service** отсутствует явное упоминание конфликтов, однако его запуск невозможен при активном firewalld.

*   
  

1. Остановил службу **iptables** и снова запустил **firewalld**, чтобы убедиться, что выбран именно firewalld в качестве активного межсетевого экрана.
2. Замаскировал сервис **iptables**, применив команду *systemctl mask*. В результате была создана символическая ссылка на /dev/null, что делает невозможным случайный запуск этой службы.
3. Попытка запуска и добавления **iptables** в автозагрузку завершилась сообщениями об ошибке. Система указала, что юнит замаскирован и не может быть активирован.

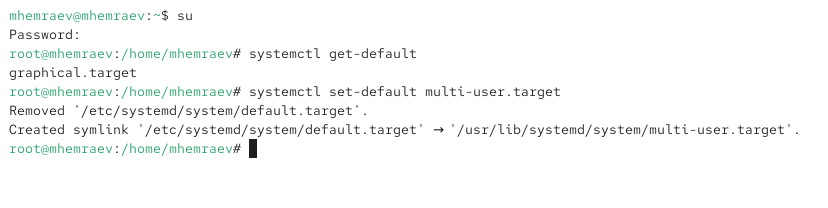
* 
* Рис. 8: Замаскированный сервис iptables

## 2.3 Изолируемые цели

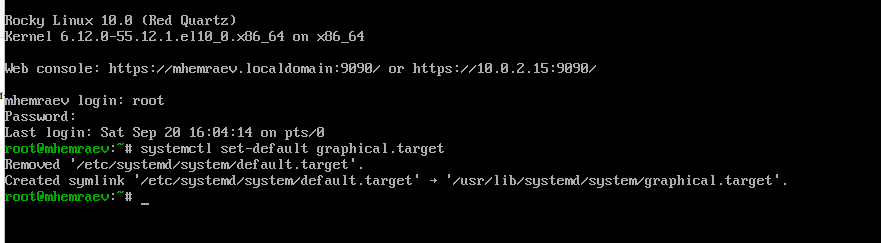
1. Получил полномочия администратора и перешёл в каталог с системными юнитами.  
   С помощью поиска по строке *AllowIsolate* определил список целей, которые можно изолировать.

* 
* Рис. 9: Список изолируемых целей

1. Проверил текущую цель по умолчанию. Система была настроена на запуск в графическом режиме (**graphical.target**).  
   Затем изменил цель по умолчанию на **multi-user.target**, что соответствует текстовому режиму.

* 
* Рис. 10: Изменение цели по умолчанию на multi-user.target

1. После перезагрузки система запустилась в текстовом режиме. Далее снова изменил цель по умолчанию на **graphical.target** и перезагрузил систему.  
   При следующем старте ОС загрузилась в графическом интерфейсе, что подтвердило корректность настроек.

* 
* Рис. 11: Возврат к графической цели по умолчанию

# 3 Контрольные вопросы

### 3.0.1 1. Что такое юнит (unit)? Приведите примеры.

* Юнит — это объект в системе systemd, описывающий ресурс или службу.
* Каждый юнит определяется конфигурационным файлом с расширением .service, .socket, .target, .mount и др.
* Примеры: sshd.service, multi-user.target, network.target.

### 3.0.2 2. Какая команда позволяет убедиться, что цель больше не входит в список автоматического запуска при загрузке системы?

* Для удаления юнита из автозапуска используется команда: systemctl disable .
* Проверить можно через: systemctl status или просмотром каталога /etc/systemd/system/multi-user.target.wants/.

### 3.0.3 3. Какую команду вы должны использовать для отображения всех сервисных юнитов, которые в настоящее время загружены?

* Команда: systemctl list-units –type=service.

### 3.0.4 4. Как создать потребность (wants) в сервисе?

* Потребность создаётся через включение юнита в автозагрузку: systemctl enable .
* При этом в каталог /etc/systemd/system/.wants/ создаётся символическая ссылка на соответствующий юнит.

### 3.0.5 5. Как переключить текущее состояние на цель восстановления (rescue target)?

* Команда: systemctl isolate rescue.target.

### 3.0.6 6. Поясните причину получения сообщения о том, что цель не может быть изолирована.

* Причина: у цели отсутствует параметр AllowIsolate=yes в конфигурации юнита.
* Только цели с этим параметром могут быть изолированы.

### 3.0.7 7. Вы хотите отключить службу systemd, но, прежде чем сделать это, вы хотите узнать, какие другие юниты зависят от этой службы. Какую команду вы бы использовали?

* Для отображения зависимостей в обратном порядке используется команда: systemctl list-dependencies –reverse.

# 4 Заключение

В ходе работы были изучены изолируемые цели и механизмы управления режимами загрузки в Linux. Получены практические навыки изменения цели по умолчанию, переключения системы в различные режимы (текстовый и графический), а также работы с командами systemctl для проверки статуса и настройки юнитов. Дополнительно рассмотрены конфликты между сервисами и способы их разрешения с помощью маскирования и управления зависимостями.