

Отчёт по лабораторной работе №5

Управление системными службами

Руслан Алиев

Содержание

1	Цель работы	5
2	Выполнение	6
2.1	Управление сервисами	6
2.2	Конфликты юнитов	9
2.3	Изолируемые цели	13
2.4	Цель по умолчанию	15
3	Заключение	16
4	Контрольные вопросы	17

Список иллюстраций

2.1	Проверка статуса службы vsftpd до установки	6
2.2	Запуск и проверка статуса vsftpd	7
2.3	Добавление службы vsftpd в автозапуск	7
2.4	Отключение автозапуска службы vsftpd	8
2.5	Символические ссылки сервисов в multi-user.target	8
2.6	Обратные зависимости vsftpd	9
2.7	Установка iptables	10
2.8	Статус firewalld и iptables	10
2.9	Конфликт запуска firewalld и iptables	11
2.10	Конфликты в firewalld.service	12
2.11	Содержимое iptables.service	12
2.12	Маскирование iptables	13
2.13	Список изолируемых целей	14
2.14	Перевод системы в rescue.target	14
2.15	Текущая цель по умолчанию	15
2.16	Возврат графического режима по умолчанию	15

Список таблиц

1 Цель работы

Получить навыки управления системными службами операционной системы посредством systemd.

2 Выполнение

2.1 Управление сервисами

1. Сначала был получен доступ администратора с помощью команды **su -**.

Проверка статуса службы **vsftpd** показала, что она отсутствует в системе, так как пакет ещё не установлен.

```
raliev@raliev:~$ su
Password:
root@raliev:/home/raliev# systemctl status vsftpd
Unit vsftpd.service could not be found.
root@raliev:/home/raliev#
root@raliev:/home/raliev# dnf -y install vsftpd
Rocky Linux 10 - BaseOS                               16 MB/s | 19 MB   00:01
Rocky Linux 10 - AppStream                             4.0 MB/s | 2.1 MB 00:00
Rocky Linux 10 - Extras                               3.0 kB/s | 5.4 kB 00:01
Dependencies resolved.
=====
Package            Architecture    Version           Repository        Size
=====
Installing:
vsftpd             x86_64         3.0.5-9.el10     appstream         170 k
Transaction Summary
=====
Install 1 Package

Total download size: 170 k
Installed size: 348 k
Downloading Packages:
vsftpd-3.0.5-9.el10.x86_64.rpm                        4.4 MB/s | 170 kB 00:00
-----
Total                                                    442 kB/s | 170 kB 00:00
Rocky Linux 10 - AppStream                             1.6 MB/s | 1.6 kB 00:00
```

Рис. 2.1: Проверка статуса службы vsftpd до установки

2. Для установки службы **Very Secure FTP (vsftpd)** использовалась команда **dnf -y install vsftpd**.

После загрузки и распаковки пакета служба стала доступна в системе.

3. Для запуска службы применялась команда **systemctl start vsftpd**.

Повторная проверка статуса показала, что сервис находится в состоянии *active (running)*, но при этом автозапуск отключён (*disabled*).

```
root@raliev:/home/raliev# systemctl start vsftpd
root@raliev:/home/raliev# systemctl status vsftpd
● vsftpd.service - Vsftpd ftp daemon
   Loaded: loaded (/usr/lib/systemd/system/vsftpd.service; disabled; preset: disabled)
   Active: active (running) since Fri 2025-10-03 16:18:03 MSK; 9s ago
  Invocation: 4331515c40d64a3b8d0567a7118e1b04
     Process: 5048 ExecStart=/usr/sbin/vsftpd /etc/vsftpd/vsftpd.conf (code=exited, status=0/SUCCESS)
    Main PID: 5052 (vsftpd)
       Tasks: 1 (limit: 24779)
      Memory: 752K (peak: 1.2M)
         CPU: 2ms
        CGroup: /system.slice/vsftpd.service
                └─5052 /usr/sbin/vsftpd /etc/vsftpd/vsftpd.conf

Oct 03 16:18:03 raliev.localdomain systemd[1]: Starting vsftpd.service - Vsftpd ftp daemon...
Oct 03 16:18:03 raliev.localdomain systemd[1]: Started vsftpd.service - Vsftpd ftp daemon.
root@raliev:/home/raliev#
```

Рис. 2.2: Запуск и проверка статуса vsftpd

4. Для добавления службы в автозапуск была выполнена команда **systemctl enable vsftpd**.

Статус изменился: теперь сервис не только работает, но и имеет состояние *enabled*, что гарантирует запуск при старте ОС.

```
root@raliev:/home/raliev# systemctl enable vsftpd
Created symlink '/etc/systemd/system/multi-user.target.wants/vsftpd.service' → '/usr/lib/systemd/system/vsftpd.service'.
root@raliev:/home/raliev# systemctl status vsftpd
● vsftpd.service - Vsftpd ftp daemon
   Loaded: loaded (/usr/lib/systemd/system/vsftpd.service; enabled; preset: disabled)
   Active: active (running) since Fri 2025-10-03 16:18:03 MSK; 3min 59s ago
  Invocation: 4331515c40d64a3b8d0567a7118e1b04
     Main PID: 5052 (vsftpd)
       Tasks: 1 (limit: 24779)
      Memory: 752K (peak: 1.2M)
         CPU: 2ms
        CGroup: /system.slice/vsftpd.service
                └─5052 /usr/sbin/vsftpd /etc/vsftpd/vsftpd.conf

Oct 03 16:18:03 raliev.localdomain systemd[1]: Starting vsftpd.service - Vsftpd ftp daemon...
Oct 03 16:18:03 raliev.localdomain systemd[1]: Started vsftpd.service - Vsftpd ftp daemon.
root@raliev:/home/raliev#
```

Рис. 2.3: Добавление службы vsftpd в автозапуск

5. Для проверки отключения автозапуска была использована команда **systemctl disable vsftpd**.

Статус показал, что служба продолжает работать (*active*), но её автоматический запуск снова выключен (*disabled*).

```
-----
root@raliev:/home/raliev# systemctl disable vsftpd
Removed '/etc/systemd/system/multi-user.target.wants/vsftpd.service'.
root@raliev:/home/raliev#
root@raliev:/home/raliev# systemctl status vsftpd
● vsftpd.service - Vsftpd ftp daemon
   Loaded: loaded (/usr/lib/systemd/system/vsftpd.service; disabled; preset: disabled)
   Active: active (running) since Fri 2025-10-03 16:18:03 MSK; 4min 51s ago
  Invocation: 4331515c40d64a3b8d0567a7118e1b04
     Main PID: 5052 (vsftpd)
        Tasks: 1 (limit: 24779)
       Memory: 752K (peak: 1.2M)
          CPU: 2ms
         CGroup: /system.slice/vsftpd.service
                 └─5052 /usr/sbin/vsftpd /etc/vsftpd/vsftpd.conf

Oct 03 16:18:03 raliev.localdomain systemd[1]: Starting vsftpd.service - Vsftpd ftp daemon...
Oct 03 16:18:03 raliev.localdomain systemd[1]: Started vsftpd.service - Vsftpd ftp daemon.
root@raliev:/home/raliev#
```

Рис. 2.4: Отключение автозапуска службы vsftpd

6. Для просмотра списка сервисов, запускаемых при переходе в состояние **multi-user.target**, применялась команда **ls /etc/systemd/system/multi-user.target.wants/**.

В выводе отсутствовала ссылка на **vsftpd.service**.

После повторного включения автозапуска (**systemctl enable vsftpd**) символическая ссылка появилась, что подтверждает корректную настройку.

```
root@raliev:/home/raliev#
root@raliev:/home/raliev# ls /etc/systemd/system/multi-user.target.wants/
atd.service      cups.service      ModemManager.service  sssd.service
auditd.service   firewallld.service  NetworkManager.service  tuned.service
audit-rules.service  irqbalance.service  remote-cryptsetup.target  vboxadd.service
avahi-daemon.service  kdump.service      remote-fs.target        vboxadd-service.service
chronyd.service    libstoragemgmt.service  rsyslog.service        vmttoolsd.service
crond.service      mcelog.service      smartd.service
cups.path          mdmonitor.service  sshd.service
root@raliev:/home/raliev# systemctl enable vsftpd
Created symlink '/etc/systemd/system/multi-user.target.wants/vsftpd.service' → '/usr/lib/systemd/system/vsftpd.service'.
root@raliev:/home/raliev# ls /etc/systemd/system/multi-user.target.wants/
atd.service      cups.service      ModemManager.service  sssd.service
auditd.service   firewallld.service  NetworkManager.service  tuned.service
audit-rules.service  irqbalance.service  remote-cryptsetup.target  vboxadd.service
avahi-daemon.service  kdump.service      remote-fs.target        vboxadd-service.service
chronyd.service    libstoragemgmt.service  rsyslog.service        vmttoolsd.service
crond.service      mcelog.service      smartd.service        vsftpd.service
cups.path          mdmonitor.service  sshd.service
root@raliev:/home/raliev#
```

Рис. 2.5: Символические ссылки сервисов в multi-user.target

7. Для анализа зависимостей юнита использовалась команда **systemctl list-dependencies vsftpd**.

Она показала, что служба привязана к **multi-user.target** и, как следствие, к **graphical.target**.

8. Для отображения обратных зависимостей была применена команда **systemctl list-dependencies vsftpd --reverse**.

В результате видно, что запуск vsftpd осуществляется в рамках **multi-user.target** и **graphical.target**.

```
○ --selinux-autorelabel-mark.service
● --sys-fs-fuse-connections.mount
● --sys-kernel-config.mount
● --sys-kernel-debug.mount
● --sys-kernel-tracing.mount
○ --systemd-ask-password-console.path
○ --systemd-binfmt.service
○ --systemd-boot-random-seed.service
○ --systemd-confext.service
○ --systemd-firstboot.service
○ --systemd-hibernate-clear.service
○ --systemd-hwdb-update.service
○ --systemd-journal-catalog-update.service
● --systemd-journal-flush.service
root@raliev:/home/raliev#
root@raliev:/home/raliev# systemctl list-dependencies vsftpd --reverse
vsftpd.service
● └multi-user.target
● └graphical.target
root@raliev:/home/raliev#
```

Рис. 2.6: Обратные зависимости vsftpd

2.2 Конфликты юнитов

1. Сначала были получены полномочия администратора. Затем выполнена установка пакета **iptables** с помощью команды **dnf -y install iptables***.

В систему были установлены дополнительные пакеты: **iptables-devel**, **iptables-nft-services**, **iptables-utils**, а также обновлены библиотеки **iptables-libs** и **iptables-nft**.

```

root@raliev:/home/raliev# dnf -y install iptables\*
Last metadata expiration check: 0:09:13 ago on Fri 03 Oct 2025 04:16:54 PM MSK.
Package iptables-libs-1.8.11-8.el10_0.x86_64 is already installed.
Package iptables-nft-1.8.11-8.el10_0.x86_64 is already installed.
Dependencies resolved.
=====
Package                        Architecture      Version           Repository        Size
=====
Installing:
iptables-devel                x86_64            1.8.11-9.el10_0   appstream         17 k
iptables-nft-services         noarch            1.8.11-9.el10_0   appstream         24 k
iptables-utils                x86_64            1.8.11-9.el10_0   appstream         42 k
Upgrading:
iptables-libs                 x86_64            1.8.11-9.el10_0   baseos            408 k
iptables-nft                  x86_64            1.8.11-9.el10_0   appstream         189 k

Transaction Summary
=====
Install  3 Packages
Upgrade  2 Packages

Total download size: 680 k

```

Рис. 2.7: Установка iptables

2. Проверка состояния служб **firewalld** и **iptables** показала:

- **firewalld.service** работает (*active, running*) и включён в автозапуск (*enabled*),
- **iptables.service** неактивен (*inactive, dead*) и автозапуск для него отключён (*disabled*).

```

root@raliev:/home/raliev# systemctl status firewalld
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; preset: enabled)
   Active: active (running) since Fri 2025-10-03 16:04:49 MSK; 21min ago
     Invocation: 7a5dc2e467d64610928811d111bf51c8
       Docs: man:firewalld(1)
    Main PID: 965 (firewalld)
       Tasks: 2 (limit: 24779)
      Memory: 48.8M (peak: 50.9M)
         CPU: 269ms
    CGroup: /system.slice/firewalld.service
            └─965 /usr/bin/python3 -sP /usr/sbin/firewalld --nofork --nopid

Oct 03 16:04:48 raliev.localdomain systemd[1]: Starting firewalld.service - firewalld - dynamic firewall
Oct 03 16:04:49 raliev.localdomain systemd[1]: Started firewalld.service - firewalld - dynamic firewall
root@raliev:/home/raliev# systemctl status iptables
○ iptables.service - IPv4 firewall with iptables
   Loaded: loaded (/usr/lib/systemd/system/iptables.service; disabled; preset: disabled)
   Active: inactive (dead)
root@raliev:/home/raliev#

```

Рис. 2.8: Статус firewalld и iptables

3. При запуске обеих служб наблюдается конфликт:

- после старта **firewalld** запуск **iptables** останавливается,
- после старта **iptables** останавливается **firewalld**.

Это подтверждает, что данные сервисы не могут функционировать одновременно.

```

root@raliev:/home/raliev# systemctl status firewalld
○ firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; preset: enabled)
   Active: inactive (dead) since Fri 2025-10-03 16:28:09 MSK; 4s ago
     Duration: 23min 19.410s
  Invocation: 7a5dc2e467d64610928811d11bf51c8
         Docs: man:firewalld(1)
   Process: 965 ExecStart=/usr/sbin/firewalld --nofork --nopid $FIREWALLD_ARGS (code=exited, status=0/SUCCESS)
    Main PID: 965 (code=exited, status=0/SUCCESS)
      Mem peak: 50.9M
         CPU: 283ms

Oct 03 16:04:48 raliev.localdomain systemd[1]: Starting firewalld.service - firewalld - dynamic firewall daemon:
Oct 03 16:04:49 raliev.localdomain systemd[1]: Started firewalld.service - firewalld - dynamic firewall daemon:
Oct 03 16:28:09 raliev.localdomain systemd[1]: Stopping firewalld.service - firewalld - dynamic firewall daemon:
Oct 03 16:28:09 raliev.localdomain systemd[1]: firewalld.service: Deactivated successfully.
Oct 03 16:28:09 raliev.localdomain systemd[1]: Stopped firewalld.service - firewalld - dynamic firewall daemon:
root@raliev:/home/raliev# systemctl status iptables
● iptables.service - IPv4 firewall with iptables
   Loaded: loaded (/usr/lib/systemd/system/iptables.service; disabled; preset: disabled)
   Active: active (exited) since Fri 2025-10-03 16:28:09 MSK; 7s ago
     Invocation: 01cf42fda7424b5290d5aaffe78afbef
   Process: 7095 ExecStart=/usr/libexec/iptables/iptables.init start (code=exited, status=0/SUCCESS)
    Main PID: 7095 (code=exited, status=0/SUCCESS)
      Mem peak: 1.5M
         CPU: 7ms

Oct 03 16:28:09 raliev.localdomain systemd[1]: Starting iptables.service - IPv4 firewall with iptables:
Oct 03 16:28:09 raliev.localdomain iptables.init[7095]: iptables: Applying firewall rules: [ OK ]
Oct 03 16:28:09 raliev.localdomain systemd[1]: Finished iptables.service - IPv4 firewall with iptables.
root@raliev:/home/raliev#

```

Рис. 2.9: Конфликт запуска firewalld и iptables

4. Содержимое файла юнита **firewalld.service** показывает, что данный сервис имеет явное указание на конфликт с **iptables.service**, **ip6tables.service**, **ebtables.service** и **ipset.service**.

Это означает, что при старте **firewalld** данные сервисы будут автоматически деактивированы.

```

root@raliev:/home/raliev#
root@raliev:/home/raliev# cat /usr/lib/systemd/system/firewalld.service
[Unit]
Description=firewalld - dynamic firewall daemon
Before=network-pre.target
Wants=network-pre.target
After=dbus.service
After=polkit.service
Conflicts=iptables.service ip6tables.service ebtables.service ipset.service
Documentation=man:firewalld(1)

[Service]
EnvironmentFile=-/etc/sysconfig/firewalld
ExecStart=/usr/sbin/firewalld --nofork --nopid $FIREWALLD_ARGS
ExecReload=/bin/kill -HUP $MAINPID
# supress to log debug and error output also to /var/log/messages
StandardOutput=null
StandardError=null
Type=dbus
BusName=org.fedoraproject.FirewallD1
KillMode=mixed

```

Рис. 2.10: Конфликты в firewalld.service

5. Содержимое файла юнита **iptables.service** показывает отсутствие явных конфликтов.

Однако он настроен как сервис типа *oneshot* с возможностью перезапуска правил, и при старте firewalld его работа блокируется.

```

root@raliev:/home/raliev#
root@raliev:/home/raliev# cat /usr/lib/systemd/system/iptables.service
[Unit]
Description=IPv4 firewall with iptables
AssertPathExists=/etc/sysconfig/iptables
Before=network-pre.target
Wants=network-pre.target

[Service]
Type=oneshot
RemainAfterExit=yes
ExecStart=/usr/libexec/iptables/iptables.init start
ExecReload=/usr/libexec/iptables/iptables.init reload
ExecStop=/usr/libexec/iptables/iptables.init stop
Environment=BOOTUP=serial
Environment=CONSOLETYPE=serial

[Install]
WantedBy=multi-user.target
root@raliev:/home/raliev#

```

Рис. 2.11: Содержимое iptables.service

6. Для исключения конфликтов была выгружена служба **iptables** (**systemctl stop iptables**) и запущен **firewalld**.

Затем для предотвращения случайного запуска службы `iptables` применена команда **`systemctl mask iptables`**, которая создала символическую ссылку на `/dev/null`.

```
root@raliev:/home/raliev#  
root@raliev:/home/raliev# systemctl stop iptables  
root@raliev:/home/raliev# systemctl start firewalld  
root@raliev:/home/raliev# systemctl mask iptables  
Created symlink '/etc/systemd/system/iptables.service' → '/dev/null'.  
root@raliev:/home/raliev# systemctl start iptables  
Failed to start iptables.service: Unit iptables.service is masked.  
root@raliev:/home/raliev# systemctl enable iptables  
Failed to enable unit: Unit /etc/systemd/system/iptables.service is masked  
root@raliev:/home/raliev#
```

Рис. 2.12: Маскирование `iptables`

7. Попытка запустить или добавить **`iptables`** в автозагрузку показала сообщение об ошибке:

- запуск невозможен, так как сервис замаскирован,
- добавление в автозагрузку также не выполняется по той же причине.

Это гарантирует, что `iptables` не будет случайно активирован и не вызовет конфликт с `firewalld`.

2.3 Изолируемые цели

1. Сначала были получены полномочия администратора.

Для поиска целей, которые могут быть изолированы, использовалась команда **`grep Isolate *.target`** в каталоге `/usr/lib/systemd/system`.

В результате отобразился список целей, содержащих строку `AllowIsolate=yes`, например: `multi-user.target`, `graphical.target`, `rescue.target`, `reboot.target` и другие.

```

root@raliev:/home/raliev#
root@raliev:/home/raliev# cd /usr/lib/systemd/system
root@raliev:/usr/lib/systemd/system# grep Isolate *.target
ctrl-alt-del.target:AllowIsolate=yes
default.target:AllowIsolate=yes
emergency.target:AllowIsolate=yes
exit.target:AllowIsolate=yes
graphical.target:AllowIsolate=yes
halt.target:AllowIsolate=yes
initrd-switch-root.target:AllowIsolate=yes
initrd.target:AllowIsolate=yes
kexec.target:AllowIsolate=yes
multi-user.target:AllowIsolate=yes
poweroff.target:AllowIsolate=yes
reboot.target:AllowIsolate=yes
rescue.target:AllowIsolate=yes
runlevel0.target:AllowIsolate=yes
runlevel1.target:AllowIsolate=yes
runlevel2.target:AllowIsolate=yes
runlevel3.target:AllowIsolate=yes
runlevel4.target:AllowIsolate=yes
runlevel5.target:AllowIsolate=yes
runlevel6.target:AllowIsolate=yes
soft-reboot.target:AllowIsolate=yes
system-update.target:AllowIsolate=yes
root@raliev:/usr/lib/systemd/system#

```

Рис. 2.13: Список изолируемых целей

- Операционная система была переведена в режим восстановления с помощью команды **systemctl isolate rescue.target**.

После этого для входа потребовалось ввести пароль суперпользователя.

```

You are in rescue mode. After logging in, type "journalctl -xb" to view
system logs, "systemctl reboot" to reboot, or "exit"
to continue bootup.
Give root password for maintenance
(or press Control-D to continue):
root@raliev:~# systemctl isolate reboot.target

```

Рис. 2.14: Перевод системы в rescue.target

- Для перезапуска системы была выполнена команда **systemctl isolate reboot.target**, что инициировало процесс перезагрузки ОС.

2.4 Цель по умолчанию

1. После получения полномочий администратора была проверена текущая цель по умолчанию с помощью команды **systemctl get-default**.

По умолчанию была установлена графическая цель (**graphical.target**).

```
raliev@raliev:~$ su
Password:
root@raliev:/home/raliev# systemctl get-default
graphical.target
root@raliev:/home/raliev# systemctl set-default multi-user.target
Removed '/etc/systemd/system/default.target'.
Created symlink '/etc/systemd/system/default.target' → '/usr/lib/systemd/system/multi-user.target'.
root@raliev:/home/raliev#
```

Рис. 2.15: Текущая цель по умолчанию

2. Для изменения цели по умолчанию и запуска системы в текстовом режиме использовалась команда **systemctl set-default multi-user.target**.

В каталоге `/etc/systemd/system/` была создана символическая ссылка на соответствующий юнит. После перезагрузки система загрузилась в режиме без графического интерфейса.

3. Для возврата загрузки в графическом режиме использовалась команда **systemctl set-default graphical.target**.

После перезагрузки система вновь загружалась с графическим интерфейсом.

```
Rocky Linux 10.0 (Red Quartz)
Kernel 6.12.0-55.12.1.el10_0.x86_64 on x86_64

Web console: https://raliev.localdomain:9090/ or https://10.0.2.15:9090/

raliev login: root
Password:
Last login: Fri Oct 3 16:33:56 on pts/0
root@raliev:~# systemctl set-default graphical.target
Removed '/etc/systemd/system/default.target'.
Created symlink '/etc/systemd/system/default.target' → '/usr/lib/systemd/system/graphical.target'.
root@raliev:~#
```

Рис. 2.16: Возврат графического режима по умолчанию

3 Заключение

В ходе работы были изучены основные приёмы управления службами и целями в **systemd**.

Было рассмотрено, как запускать и останавливать сервисы, включать и отключать их из автозагрузки, а также анализировать зависимости между юнитами.

Особое внимание уделялось работе с изолируемыми целями (runlevels в терминах SystemV init) и настройке цели по умолчанию при загрузке системы.

Кроме того, были исследованы конфликты юнитов (например, между **firewalld** и **iptables**) и способы их разрешения.

4 Контрольные вопросы

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

Юнит — это объект управления systemd, который описывает сервис, сокет, точку монтирования, устройство, цель и т.д.

Примеры: `sshd.service`, `network.target`, `tmp.mount`, `rescue.target`.

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

`systemctl disable имя_цели`

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

`systemctl list-units -type=service`

4. Как создать потребность (wants) в сервисе?

`systemctl enable имя_сервиса`

(при этом создаётся символическая ссылка в каталог `*.wants/` на соответствующий юнит).

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

`systemctl isolate rescue.target`

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

Цель не может быть изолирована, если в её юнит-файле отсутствует параметр `AllowIsolate=yes`.

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?**

`systemctl list-dependencies имя_сервиса --reverse`