

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ №5

дисциплина: Основы администрирования операционных систем

Студент: Накова Амина

Студ. билет № 1132232887

Группа: НПИбд-02-23

МОСКВА

2025 г.

Цель работы:

Целью данной работы является получение навыков управления системными службами операционной системы посредством systemd.

Выполнение работы:

Управление сервисами:

Для начала получим полномочия администратора **su -**. Затем проверим статус службы Very Secure FTP: **systemctl status vsftpd**. Вывод команды показывает, что сервис в настоящее время отключён, так как служба Very Secure FTP не установлена. Установим службу Very Secure FTP: **dnf -y install vsftpd** и запустим: **systemctl start vsftpd** (Рис. 1.1):

```

Unit vsftpd.service could not be found.
[root@ismakhorin ~]# dnf -y install vsftpd
Rocky Linux 9 - BaseOS                650 B/s | 3.6 kB    00:05
Rocky Linux 9 - AppStream              6.1 kB/s | 3.6 kB    00:00
Rocky Linux 9 - Extras                 5.8 kB/s | 2.9 kB    00:00
Dependencies resolved.
=====
Package                Architecture    Version          Repository        Size
=====
Installing:
vsftpd                 x86_64          3.0.3-49.el9     appstream         158 k
Transaction Summary
=====
Install 1 Package

Total download size: 158 k
Installed size: 348 k
Downloading Packages:
vsftpd-3.0.3-49.el9.x86_64.rpm        604 kB/s | 158 kB    00:00
-----
Total                                244 kB/s | 158 kB    00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing      :                                1/1
  Installing     : vsftpd-3.0.3-49.el9.x86_64    1/1
  Running scriptlet: vsftpd-3.0.3-49.el9.x86_64    1/1
  Verifying      : vsftpd-3.0.3-49.el9.x86_64    1/1

Installed:
vsftpd-3.0.3-49.el9.x86_64

Complete!

```

Рис. 1.1. Открытие режима работы суперпользователя, проверка статуса, установка и запуск службы Very Secure FTP.

Снова проверим статус службы Very Secure FTP: **systemctl status vsftpd**. Вывод команды показывает, что служба в настоящее время работает, но не будет активирована при перезапуске операционной системы (Рис. 1.2):

```

Process: 2705 ExecStart=/usr/sbin/vsftpd /etc/vsftpd/vsftpd.conf (code=exited, sta
Main PID: 2706 (vsftpd)
Tasks: 1 (limit: 12209)
Memory: 704.0K
CPU: 4ms
CGroup: /system.slice/vsftpd.service
└─2706 /usr/sbin/vsftpd /etc/vsftpd/vsftpd.conf

```

Рис. 1.2. Проверка статуса службы Very Secure FTP.

Добавим службу Very Secure FTP в автозапуск при загрузке операционной системы, используя команду **systemctl enable vsftpd**. Затем проверим статус службы и удалим службу из автозапуска, используя команду **systemctl disable vsftpd**, и снова проверим её статус (Рис. 1.3):

```
Starting Vsftpd ftp daemon...
Started Vsftpd ftp daemon.
```

Рис. 1.3. Добавление службы Very Secure FTP в автозапуск и проверка её статуса.

Далее выведем на экран символические ссылки, ответственные за запуск различных сервисов: **ls /etc/systemd/system/multi-user.target.wants**. Отображается, что ссылка на **vsftpd.service** не существует. Снова добавляем службу Very Secure FTP в автозапуск: **systemctl enable vsftpd** и выводим на экран символические ссылки, ответственные за запуск различных сервисов. Вывод команды показывает, что создана символическая ссылка для файла **/usr/lib/systemd/system/vsftpd.service** в каталоге **/etc/systemd/system/multi-user.target.wants**. Проверяем статус службы Very Secure FTP: **systemctl status vsftpd**. Теперь мы видим, что для файла юнита состояние изменено с **disabled** на **enabled** (Рис. 1.4):

```
atd.service          irqbalance.service  rsyslog.service
auditd.service       kdump.service       smartd.service
avahi-daemon.service libstoragemgmt.service sshd.service
chronyd.service      mcelog.service      sssd.service
crond.service        mdmonitor.service   vboxadd.service
cups.path            ModemManager.service vboxadd-service.service
cups.service         NetworkManager.service vmtoolsd.service
firewalld.service    remote-fs.target
```

Рис. 1.4. Вывод на экран символических ссылок, добавление службы Very Secure FTP в автозапуск, проверка статуса службы.

Выведем на экран список зависимостей юнита: **systemctl list-dependencies vsftpd** (Рис. 1.5) и список юнитов, которые зависят от данного юнита: **systemctl list-dependencies vsftpd --reverse** (Рис. 1.6):

```
vsftpd.service
● └─system.slice
● └─sysinit.target
●   └─dev-hugepages.mount
●   └─dev-mqueue.mount
●   └─dracut-shutdown.service
○   └─iscsi-onboot.service
●   └─kmod-static-nodes.service
○   └─ldconfig.service
●   └─lvm2-lvmpolld.socket
●   └─lvm2-monitor.service
○   └─multipathd.service
●   └─nis-domainname.service
●   └─plymouth-read-write.service
●   └─plymouth-start.service
●   └─proc-sys-fs-binfmt_misc.automount
○   └─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-firstboot.service
○   └─systemd-hwdb-update.service
○   └─systemd-journal-catalog-update.service
●   └─systemd-journal-flush.service
●   └─systemd-journald.service
○   └─systemd-machine-id-commit.service
●   └─systemd-modules-load.service
●   └─systemd-network-generator.service
●   └─systemd-random-seed.service
○   └─systemd-repart.service
●   └─systemd-sysctl.service
○   └─systemd-sysusers.service
●   └─systemd-tmpfiles-setup-dev.service
●   └─systemd-tmpfiles-setup.service
●   └─systemd-udev-trigger.service
●   └─systemd-udevd.service
○   └─systemd-update-done.service
●   └─systemd-update-utmp.service
●   └─cryptsetup.target
●   └─integritysetup.target
●   └─local-fs.target
●     └─.mount
●     └─boot.mount
```

lines 1-46

Рис. 1.5. Список зависимостей юнита.

```
vsftpd.service
● └─multi-user.target
● └─graphical.target
```

Рис. 1.6. Список юнитов, которые зависят от данного юнита.

Конфликты юнитов:

Получим полномочия администратора **su** – и установим iptables: **dnf -y install iptables*** (Рис. 2.1).

```

Package iptables-libs-1.8.7-28.el9.x86_64 is already installed.
Package iptables-nft-1.8.7-28.el9.x86_64 is already installed.
Dependencies resolved.
=====
Package                Architecture Version                Repository            Size
=====
Installing:
iptables-devel          x86_64             1.8.7-28.el9          appstream             19 k
iptables-nft-services   noarch             1.8.7-28.el9          appstream             22 k
iptables-utils          x86_64             1.8.7-28.el9          baseos                44 k

Transaction Summary
=====
Install 3 Packages

Total download size: 85 k
Installed size: 141 k
Downloading Packages:
(1/3): iptables-utils-1.8.7-28.el9.x86_64.rpm          119 kB/s | 44 kB      00:00
(2/3): iptables-devel-1.8.7-28.el9.x86_64.rpm          45 kB/s | 19 kB      00:00
(3/3): iptables-nft-services-1.8.7-28.el9.noarch.rpm    52 kB/s | 22 kB      00:00
-----
Total                                                67 kB/s | 85 kB      00:01
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing      :                                1/1
  Installing     : iptables-devel-1.8.7-28.el9.x86_64 1/3
  Installing     : iptables-nft-services-1.8.7-28.el9.noarch 2/3
  Running scriptlet: iptables-nft-services-1.8.7-28.el9.noarch 2/3
  Installing     : iptables-utils-1.8.7-28.el9.x86_64 3/3
  Running scriptlet: iptables-utils-1.8.7-28.el9.x86_64 3/3
  Verifying      : iptables-utils-1.8.7-28.el9.x86_64 1/3
  Verifying      : iptables-nft-services-1.8.7-28.el9.noarch 2/3
  Verifying      : iptables-devel-1.8.7-28.el9.x86_64 3/3

Installed:
  iptables-devel-1.8.7-28.el9.x86_64      iptables-nft-services-1.8.7-28.el9.noarch
  iptables-utils-1.8.7-28.el9.x86_64

Complete!

```

Рис. 2.1. Получение полномочий администратора и установка iptables.

Далее проверим статус firewalld и iptables: **systemctl status firewalld** и **systemctl status iptables** (Рис. 2.2).

```

lines 1-13/13 (END)
[8]+ Stopped                                systemctl status firewalld
[root@ismakhorin ~]# systemctl status iptables
o iptables.service - IPv4 firewall with iptables
   Loaded: loaded (/usr/lib/systemd/system/iptables.service; disabled; vendor preset>
   Active: inactive (dead)
lines 1-3/3 (END)

```

Рис. 2.2. Проверка статуса firewalld и iptables.

Попробуем запустить firewalld и iptables: **systemctl start firewalld** и **systemctl start iptables**. Мы видим, что при запуске одной службы вторая деактивируется или не запускается (Рис. 2.3).

Рис. 2.3. Попытка запуска firewalld и iptables.

Введем **cat /usr/lib/systemd/system/firewalld.service** и опишем настройки конфликтов для этого юнита при наличии, далее введём **cat /usr/lib/systemd/system/iptables.service** и опишем настройки конфликтов для этого юнита (Рис. 2.4).


```

[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 nftables.service
Documentation=man:firewalld(1)

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

[Install]
WantedBy=multi-user.target
Alias=dbus-org.fedoraproject.FirewallD1.service
[root@ismakhorin ~]# 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

```

Рис. 2.4. Настройки конфликтов для юнитов.

Выгрузим службу iptables (на всякий случай, чтобы убедиться, что данная служба не загружена в систему): **systemctl stop iptables** и загрузим службу firewalld: **systemctl start firewalld**. Далее заблокируем запуск iptables, введя: **systemctl mask iptables**. Видим, как создана символическая ссылка на /dev/null для /etc/systemd/system/iptables.service. Поскольку юнит-файлы в /etc/systemd имеют приоритет над файлами в /usr/lib/systemd, то это сделает невозможным случайный запуск сервиса iptables. Для проверки попробуем запустить iptables: **systemctl start iptables**. После попытки запуска появилось сообщение об

ошибке, указывающее, что служба замаскирована и по этой причине не может быть запущена. Теперь попробуем добавить iptables в автозапуск: **systemctl enable iptables**. Показывает, что сервис неактивен, а статус загрузки отображается как замаскированный (Рис. 2.5).

```
Failed to enable unit: Unit file /etc/systemd/system/iptables.service is masked.
```

Рис. 2.5. Выгрузка службы iptables, загрузка службы firewalld, блокировка запуска iptables, попытка запуска iptables и добавления iptables в автозапуск.

Изолируемые цели:

Получим полномочия администратора **su** – и перейдём в каталог **systemd**, найдём список всех целей, которые можно изолировать:

```
cd /usr/lib/systemd/system
```

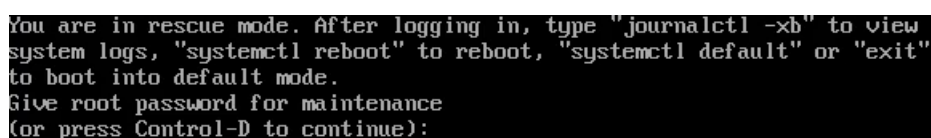
```
grep Isolate *.target
```

На следующем шаге переключим операционную систему в режим восстановления: **systemctl isolate rescue.target** (Рис. 3.1).

```
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
system-update.target:AllowIsolate=yes
```

Рис. 3.1. Открытие каталога. Нахождение списка всех целей, которые можно изолировать. Переключение операционной системы в режим восстановления.

Как только операционная система переключилась в режим восстановления вводим пароль root. После чего перезапустим операционную систему следующим образом: **systemctl isolate reboot.target** (Рис. 3.2).

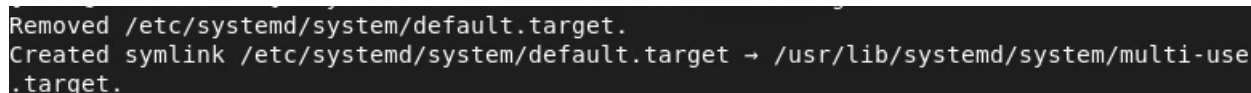


```
You are in rescue mode. After logging in, type "journalctl -xb" to view
system logs, "systemctl reboot" to reboot, "systemctl default" or "exit"
to boot into default mode.
Give root password for maintenance
(or press Control-D to continue):
```

Рис. 3.2. Перезапуск операционной системы.

Цель по умолчанию:

Получим полномочия администратора **su** – и выведем на экран цель, установленную по умолчанию: **systemctl get-default**. Для установки цели по умолчанию используется команда **systemctl set-default**. В нашем случае для запуска по умолчанию текстового режима введём **systemctl set-default multi-user.target**. После чего перезагрузим систему командой **reboot** (Рис. 4.1)



```
Removed /etc/systemd/system/default.target.
Created symlink /etc/systemd/system/default.target → /usr/lib/systemd/system/multi-use
.target.
```

Рис. 4.1. Получение полномочий администратора, установка запуска по умолчанию текстового режима, последующая перезагрузка системы.

Убедимся, что система загрузилась в текстовом режиме, после чего получим полномочия администратора (для начала зайдём в пользователя ismakhorin, а затем в режим администратора). Для запуска по умолчанию графического режима введём **systemctl set-default graphical.target** и вновь перегрузим систему командой **reboot** (Рис. 4.2). Убедимся, что система загрузилась в графическом режиме (Рис. 4.3)

```
Password:
[root@ismakhorin ~]# 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@ismakhorin ~]# reboot
```

Рис. 4.2. Загрузка системы в текстовом режиме, получение полномочий администратора, установка запуска по умолчанию графического режима и последующая перезагрузка системы.

Рис. 4.3. Загрузка системы в графическом режиме.

Ответы на контрольные вопросы:

1. Что такое юнит (unit)? Приведите примеры. **Unit** – объект, которым может управлять система.

2. Какая команда позволяет вам убедиться, что цель больше не входит в список автоматического запуска при загрузке системы? **systemctl is-enable “имя_юнита”** (пример: **systemctl is-enable vsftpd.service**).

3. Какую команду вы должны использовать для отображения всех сервисных юнитов, которые в настоящее время загружены? **system list-units.**

```
UNIT
proc-sys-fs-binfmt_misc.automount
sys-devices-pci0000:00-0000:00:01.1-ata2-host1-target1:0:0-1:0:0:0-block-sr0.>
sys-devices-pci0000:00-0000:00:03.0-net-enp0s3.device
sys-devices-pci0000:00-0000:00:05.0-sound-card0-controlC0.device
sys-devices-pci0000:00-0000:00:0d.0-ata3-host2-target2:0:0-2:0:0:0-block-sda->
sys-devices-pci0000:00-0000:00:0d.0-ata3-host2-target2:0:0-2:0:0:0-block-sda->
sys-devices-pci0000:00-0000:00:0d.0-ata3-host2-target2:0:0-2:0:0:0-block-sda->
sys-devices-platform-serial8250-tty-ttyS0.device
sys-devices-platform-serial8250-tty-ttyS1.device
sys-devices-platform-serial8250-tty-ttyS2.device
sys-devices-platform-serial8250-tty-ttyS3.device
sys-devices-virtual-block-dm\x2d0.device
sys-devices-virtual-block-dm\x2d1.device
sys-devices-virtual-misc-rfkill.device
sys-module-configfs.device
sys-module-fuse.device
sys-subsystem-net-devices-enp0s3.device
-.mount
boot.mount
dev-hugepages.mount
```

4. Как создать потребность (wants) в сервисе? **Нужно внести всю необходимую информацию в переменную “Wants”, которая находится в файле имя_сервиса.service.**

5. Как переключить текущее состояние на цель восстановления (rescue target)? **systemctl set-default rescue.target.**

```
# systemctl set-default rescue.target
```

6. Поясните причину получения сообщения о том, что цель не может быть изолирована. **Изолируя цель, мы запускаем эту цель со всеми её зависимостями. Не все цели могут быть изолированы (в случае, если цель является неотъемлемой частью system).**

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_юнита” --reverse (пример: systemctl list-dependencies firewalld.service --reverse).**

```
firewalld.service
● └─multi-user.target
●   └─graphical.target
```

Вывод:

В ходе выполнения лабораторной работы были получены навыки управления системными службами операционной системы посредством systemd.