

ДИСЦИПЛИНА НАСТРОЙКА СЕТЕВЫХ СЕРВИСОВ
ИНСТИТУТ РАДИОЭЛЕКТРОНИКИ И ИНФОРМАТИКИ
КАФЕДРА ТЕЛЕКОММУНИКАЦИЙ
ВИД УЧЕБНОГО МАТЕРИАЛА ПРАКТИКА
ПРЕПОДАВАТЕЛЬ ЗАБОТКИНА ЕКАТЕРИНА МИХАЙЛОВНА
СЕМЕСТР СЕДЬМОЙ, 2024/2025

Практическая работа № 7. Управление устройствами, модулями ядра и программным обеспечением. Система журналирования в Astra Linux

1. Изучение процесса загрузки системы

1.1. Перезагрузите систему и нажмите клавишу `e` для перехода в загрузчик GRUB. Введите username и пароль. Для выхода нажмите `Ctrl-X`.

Примечание: команда `linux` позволяет загрузить ядро и передать ему параметры, `initrd` загружает образ `initramfs` и передает управление ядру. Все изменения в данном режиме будут происходить только на текущую сессию.

Чтобы войти в GRUB shell, нужно нажать на клавишу `c`.

1.2. Выполните перезагрузку еще раз и войдите в GRUB shell. Посмотрите список команд и содержимое. Далее симулируйте загрузку ядра и образа.

`TAB`

`ls`

`linux /vmlinuz root=/dev/sda1 ro parsec.max_ilev=63 quiet
net.ifnames=0`

`initrd /initrd.img`

`boot`

1.3. Изучите содержимое конфигурационного файла загрузчика GRUB.

`sudo head /boot/grub/grub.cfg`

Примечание:

`/etc/default/grub` – установка переменных для GRUB.

/etc/grub.d содержит сценарии для формирования grub.cfg, они должны иметь право на выполнение.

Для редактирования предназначен файл 40_custom.

grub-mkconfig генерирует конфигурационный файл для GRUB.

1.4. Отобразите содержимое файла /etc/default/grub.

```
cat /etc/default/grub
```

Примечание:

GRUB_TIMEOUT – время ожидания загрузчика.

GRUB_DISTRIBUTOR – параметр, указывающий на название дистрибутива, которое будет отображаться в GRUB.

GRUB_CMDLINE_LINUX_DEFAULT – параметры, передаваемые ядру системы по умолчанию.

GRUB_CMDLINE_LINUX – дополнительные параметры ядра.

1.5. В файле /etc/default/grub измените время ожидания загрузчика, установите значение 10 секунд и включите параметр загрузки splash.

```
GRUB_TIMEOUT=10
```

```
GRUB_CMDLINE_LINUX_DEFAULT="parsec.max_ilev=63 splash  
net.ifnames=0"
```

1.6. Перечитайте конфигурационный файл загрузчика. Далее перезагрузите систему для просмотра изменений.

```
sudo update-grub2
```

```
sudo reboot
```

2. Управление службами с помощью systemd

2.1. Просмотрите все юниты, которые есть в системе.

```
systemctl -a
```

2.2. Просмотрите все юниты, которые относятся к типу target.

```
systemctl --type=target
```

2.3. Отобразите статус сетевой службы NetworkManager и проанализируйте содержимое вывода.

```
systemctl status NetworkManager
```

2.4. Отобразите содержимое юнита, с которым связана служба NetworkManager.

```
cat /lib/systemd/system/NetworkManager.service
```

3. Юниты systemd типа target

Таргеты позволяют объединить различные юниты в группу. Например, можно создать target который будет запускать сразу несколько служб.

В отличие от сервисов, этот тип юнитов имеют только 2 блока – [Unit] и [Install]. Отдельного блока [target] не существует.

3.1. Объедините службы ssh и networking.

```
sudo nano /etc/systemd/system/ssh-networking.target
```

```
[Unit]
```

```
Description=ssh networking target
```

```
Requires=ssh.service
```

```
Requires=networking.service
```

```
[Install]
```

```
WantedBy=multi-user.target
```

Примечание:

Description – описание юнита.

Requires – в этом параметре указывается юнит, который должен запуститься при запуске таргета.

WantedBy – на каком загрузочном таргете запускать этот юнит.

3.2. Проверьте работу созданного таргета. Для этого выключите обе службы и убедитесь, что они выключены, а также обновите информацию обо всех юнитах.

```
sudo systemctl stop networking.service
```

```
sudo apt install ssh
sudo systemctl stop ssh.service
systemctl status networking.service
systemctl status ssh.service
sudo systemctl daemon-reload
```

3.3. Далее запустите созданный `ssh-networking.target`. И проверьте статус таргета и связанных служб.

```
sudo systemctl start ssh-networking.target
systemctl status ssh-networking.target
systemctl status networking.service
systemctl status ssh.service
```

3.4. Добавьте созданный таргет в автозагрузку.

```
sudo systemctl enable ssh-networking.target
```

Примечание:

Таким образом, вы можете объединить запуск нескольких служб, например, 1С и службу баз данных postgres или веб приложения: `apache2`, `php` и `mariadb`.

4. Загрузочные таргеты

В Linux есть так называемые режимы запуска, в первой системе инициализации они назывались “Уровни запуска” (`runlevel`). В `systemd` они называются загрузочными таргетами, или `boot targets`.

На каждом уровне – запускаются или останавливаются определённые службы. В отличие от Windows, в Linux можно переходить из одного уровня в другой не перезагружаясь. Хотя перезагрузка – это один из уровней запуска. И даже выключение – это тоже определённый уровень запуска.

Можно настроить определенный уровень запуска, как уровень по умолчанию. Именно в этом режиме будет загружаться ваша система.

4.1. Посмотрите список существующих загрузочных таргетов.

```
ls -l /lib/systemd/system/runlevel*.target
```

Примечание:

В этом примере видна связь загрузочных таргетов systemd и уровней запуска (runlevel). То есть каждый уровень запуска – это символическая ссылка на определенный таргет.

4.2. Перезагрузите систему с использованием таргета reboot.

```
sudo systemctl isolate reboot
```

или

```
sudo init 6
```

4.3. Просмотрите, на что ссылается таргет по умолчанию.

```
ls -l /lib/systemd/system/default.target
```

или

```
systemctl get-default
```

4.4. Отобразите содержание юнита для default.target и проанализируйте содержимое.

```
cat /lib/systemd/system/default.target
```

Примечание:

`Requires=multi-user.target` – при запуске этого юнита будет запущен юнит `multi-user.target`.

`Wants=display-manager.service` – этот юнит ожидает запуска службы `display-manager.service`, но не требует ее запуска.

`Conflicts=rescue.service rescue.target` – параметр `Conflicts` означает, что если указанный юнит выполняется, то этот не будет запущен. А режим `rescue.target` – это режим восстановления, чем-то похож на Безопасный режим в Windows. Если запущен режим восстановления (`rescue.target`), то нельзя запускать графический режим (`graphical.target`).

`After` – по очередности юнит должен запускаться после указанных юнитов, но не требует их запуска.

`AllowIsolate` – если этот юнит можно использовать в команде `systemctl isolate <название таргета>`, то здесь нужно поставить `yes`. То есть с помощью этого параметра мы указываем, что это именно `boot target`.

5. Управление модулями ядра

В Linux названия всех модулей заканчиваются расширением `.ko`, и обычно они загружаются автоматически при обнаружении оборудования во время загрузки системы. Однако системный администратор может управлять модулями с помощью специальных команд.

5.1. Посмотрите все загруженные модули ядра. Какой драйвер загружен для сетевого интерфейса? В качестве примера отобразите информацию про драйвер `e1000`.

```
lsmod  
  
sudo dmesg | grep eth0  
  
sudo modinfo e1000
```

5.2. Отобразите список доступных сетевых карт.

```
lspci | grep -i 'net'
```

5.3. Просмотрите информацию о драйвере `e100` и загрузите его.

```
sudo modinfo e100  
  
sudo modprobe e100
```

5.4. Перейдите в редактирование файла `/etc/modules` и добавьте информацию о том, что будет использоваться новый драйвер.

```
alias eth0 e100
```

5.6. Обновите файл `initramfs` после внесенных изменений.

```
sudo update-initramfs -u
```

5.5. Посмотрите параметры ядра. Далее отобразите значение параметра `ip_forward`. Включите поддержку маршрутизации между двумя сетевыми интерфейсами.

```
sudo sysctl -a | less  
  
cat /proc/sys/net/ipv4/ip_forward  
  
sudo -i  
  
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Примечание: пересылка пакетом между двумя сетевыми интерфейсами отключена.

Для того, чтобы поправить параметр в конфигурационном файле, необходимо внести изменения в `/etc/sysctl.conf`.

6. Управление устройствами

Подсистема `udev` предназначена для настройки и управления устройствами, которые были идентифицированы ядром ОС.

6.1. Отобразите информацию о разделе диска `/dev/sda1`. Определите полный путь до устройства и уникальный идентификатор.

```
sudo udevadm info /dev/sda1
```

6.2. Отобразите информацию о всех родительских устройствах для `/dev/sda1`.

```
sudo udevadm info -a /dev/sda1
```

6.3. Ознакомьтесь с графическим интерфейсом для менеджера устройств.

6.4. Узнайте, какие устройства подключены к шине PCI, а также параметры CPU и оперативной памяти.

```
lspci
```

```
lscpu
```

или

```
cat /proc/cpuinfo
```

```
cat /proc/meminfo
```

7. Управление программным обеспечением

7.1. Посмотрите, какие пакеты установлены в системе и посчитайте их количество.

```
dpkg -l
```

```
dpkg -l | wc -l
```

или

```
sudo aptitude
```

7.2. Узнайте информацию о пакете командного интерпретатора. Найдите все файлы, которые связаны с этим пакетом.

```
dpkg -l bash
```

```
dpkg -L bash
```

Примечание: `ii` – пакет установлен, `iA` – пакет установлен как зависимый.

7.3. Отобразите информацию о репозиториях, доступных системе.

```
cat /etc/apt/sources.list
```

Примечание: по умолчанию репозиторием является сам диск CD/DVD.

7.4. Установите пакеты `ssh`, `isc-dhcp-server`, `bind9`, `apache2`.

```
sudo apt install ssh isc-dhcp-server bind9 apache2
```

8. Службы журналирования

Система журналирования `journald` входит в состав `systemd`. Конфигурационный файл `/etc/systemd/journal.conf`.

Параметр `Storage` регулирует, где хранится журнал:

– `volatile` – в `/run/log/journal` (временная память);

– `persistent` – в `/var/log/journal` (постоянная память);

– `auto` – если есть каталог `/var/log/journal`, то в нем, если нет, то в `/run/log/journal` (по умолчанию).

Основные параметры утилиты `journalctl`:

```
journalctl --since "дата" --until "дата". Формат даты: YYYY-MM-DD  
HH:MM:SS;
```

```
journalctl -f – просматривать в реальном времени (аналог tail -f имя_журнала);
```

```
journalctl -u имя_юнита – сообщения от юнита;
```

```
journalctl -k – сообщения от ядра;
```

```
journalctl -p приоритет – фильтрация по приоритету (уровню критичности  
сообщений);
```

```
journalctl -n число – последние n сообщений;
```

```
journalctl имя_поля=значение – выводит сообщения, в котором указанное поле  
принимает заданное значение. Часто используемые имена полей: _UID, _PID;
```

`journalctl -N` – выводит список полей, используемых в сообщениях, хранимых в журнале;

`journalctl -F поле` – выводит возможные значения для указанного поля;

`journalctl -o verbose` – вывод сообщений в детализированном виде (кроме сообщения выводятся поля со значениями);

`systemd-cat` – передача сообщений в систему журналирования (может использоваться для тестирования или в скриптах). Данные из стандартного потока ввода перенаправляются в систему журналирования.

Параметр `-p` задает уровень критичности сообщения.

8.1. Отобразите сообщения, которые поступают в реальном времени при перезапуске службы `NetworkManager`. Для демонстрации сообщений в реальном времени отключите и включите сетевой интерфейс.

```
sudo journalctl -f
```

8.2. Просмотрите сообщения об ошибках, связанные с пользователем с `UID=1000`.

```
sudo journalctl _UID=1000 -p err
```

8.3. Выведите сообщения журнала для юнита `data1.mount` за определенный период времени.

```
journalctl --until "1 second ago" -u data1.mount
```

8.4. Просмотрите журнал предыдущей загрузки.

```
journalctl -b -1
```

Примечание: по умолчанию `journald` перезаписывает свои журналы логов при каждой перезагрузке, и вызов `journalctl` выведет журнал логов начиная с текущей загрузки системы.

Когда в конфигурационном файле `/etc/systemd/journald.conf` параметр `Storage=auto` и каталога `/var/log/journal/` не существует, журнал будет записываться в `/run/log/journal` без сохранения между перезагрузками.

Если `/var/log/journal/` существует, журналы будут сохраняться в нем, на постоянной основе.

8.5. Создайте каталог для хранения журналов, установите необходимые атрибуты и перезапустите службу. Для демонстрации изменений можно перезагрузить систему и еще раз ввести команду `journalctl -b -1`

```
sudo mkdir /var/log/journal
```

```
sudo systemd-tmpfiles --create --prefix /var/log/journal
```

```
sudo systemctl restart systemd-journald
```

8.6. Сделайте так, чтобы `journalctl` показывал сообщения без прерывания строк.

По умолчанию `journalctl` отсекает части строк, которые не вписываются в экран по ширине, хотя иногда перенос строк может оказаться более предпочтительным.

Управление этой возможностью производится посредством переменной окружения `SYSTEMD_LESS`, в которой содержатся опции, передаваемые в `less` (программу постраничного просмотра, используемую по умолчанию). По умолчанию переменная имеет значение `FRSXMK`, если убрать опцию `S`, строки не будут обрезаться.

```
sudo SYSTEMD_LESS=FRXMK journalctl
```