**Федеральное агентство по образованию**

**ГОУ ВПО Нижегородский государственный технический университет им. Р.Е. Алексеева**

**Институт радиоэлектроники и информационных технологий**

**Кафедра “Вычислительные системы и**

**Технологии”**

**Эксплуатация современных операционных систем**

**Отчёт по лабораторной работе №1**

Студент группы 21-ПО

Малинок С.М. ­­­­­\_\_\_\_\_\_\_

Провел преподаватель кафедры ВСТ

Кабальцев Н.Д.\_\_\_\_\_\_\_

Г. Нижний Новгород 2024

1.Процес установки ОС Astra Linux.

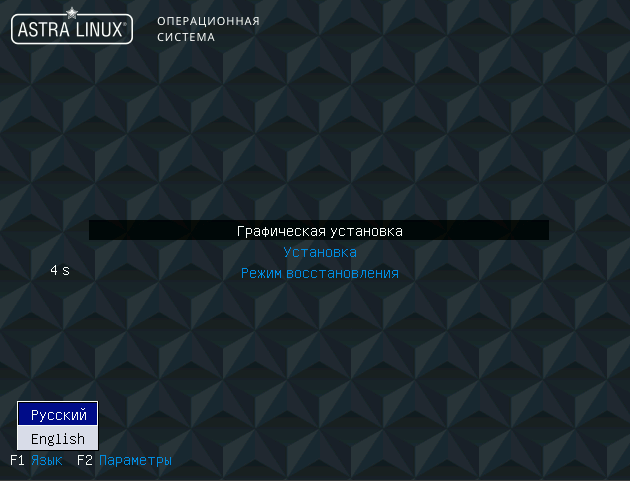
Расписать порядок установки Astra Linux:

* Язык установки и тип установки:

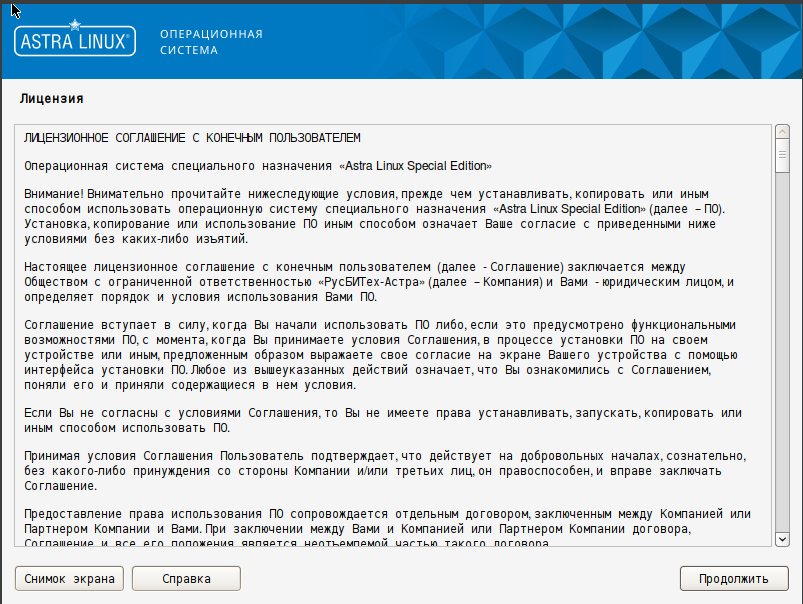
Язык установки – Русский

Тип установки – Графика

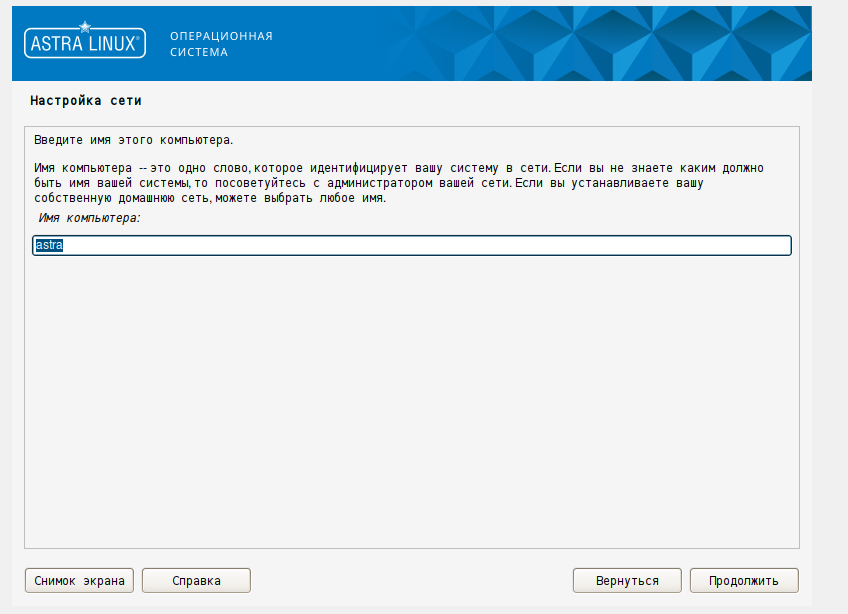
Скриншот:



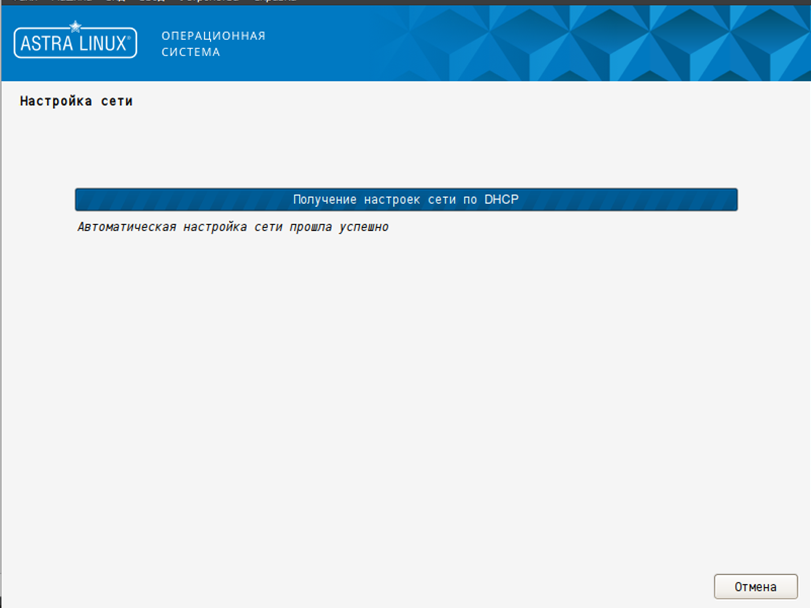
* Лицензионное соглашение:



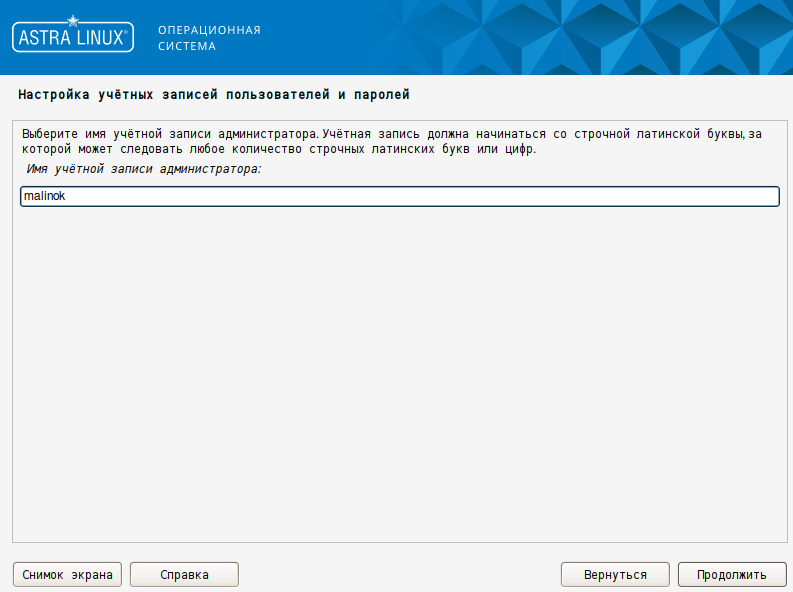
* Ввод имени хоста:

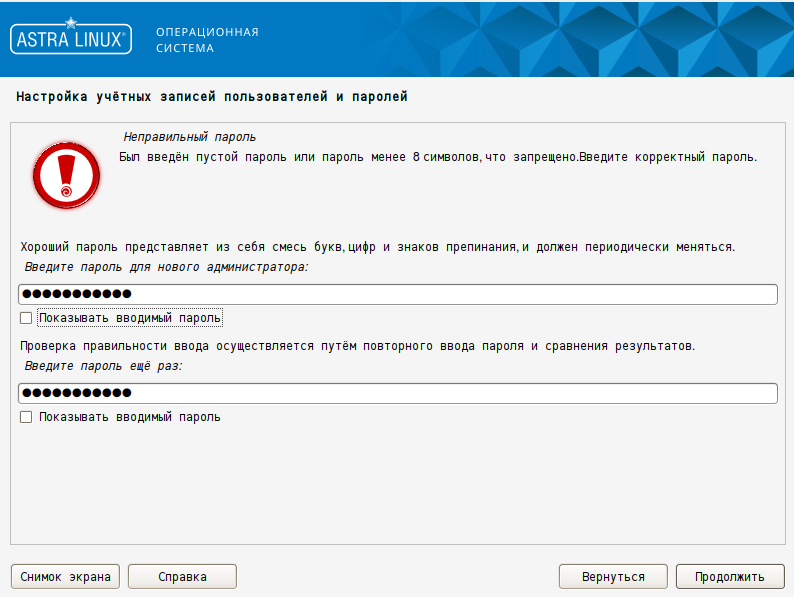


* Настройка сети (без участия пользователя)



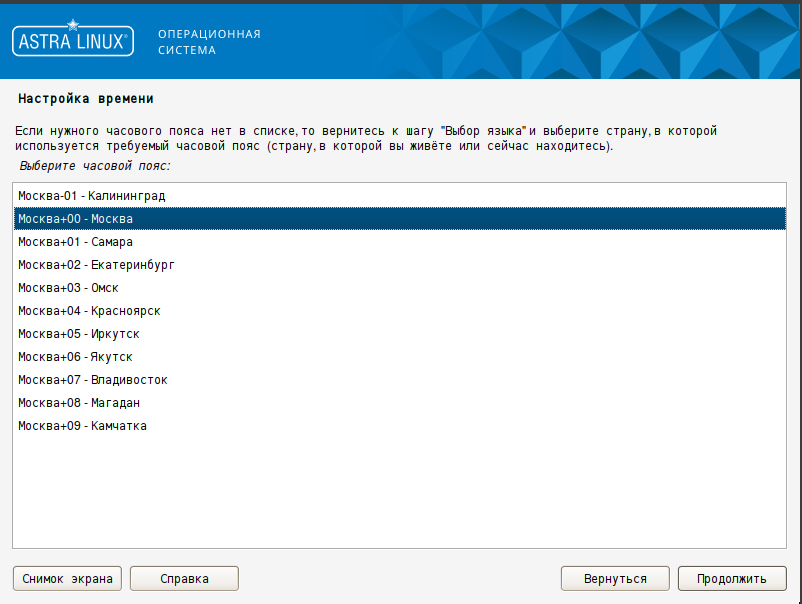
* Настройка учетной записи



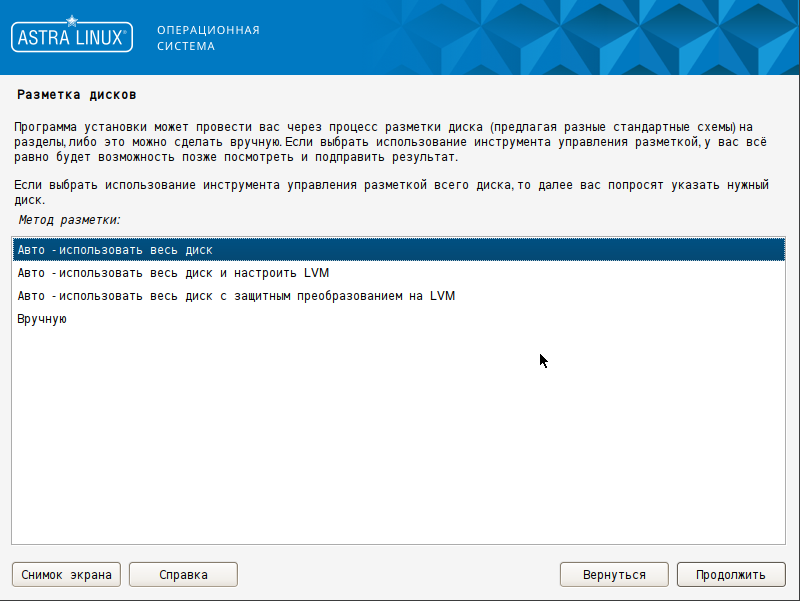


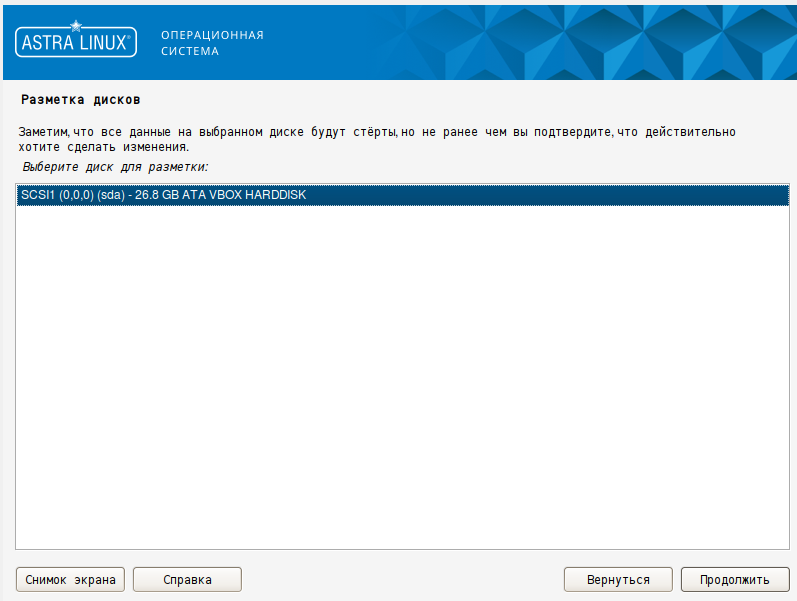
* Выбор часового пояса

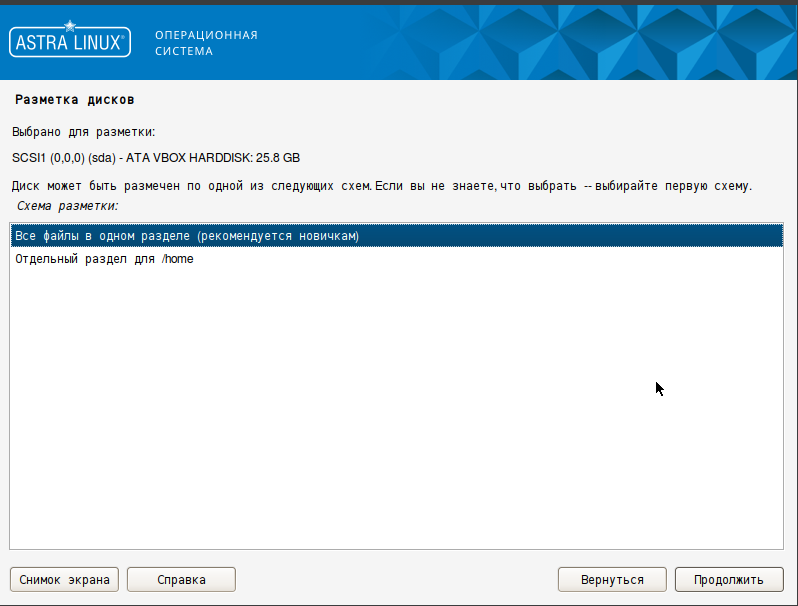
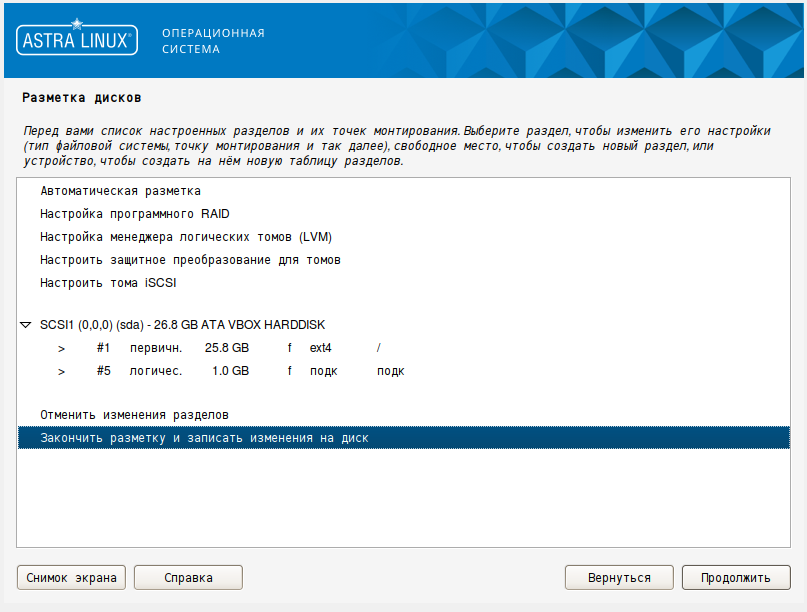
Часовой пояс: Москва



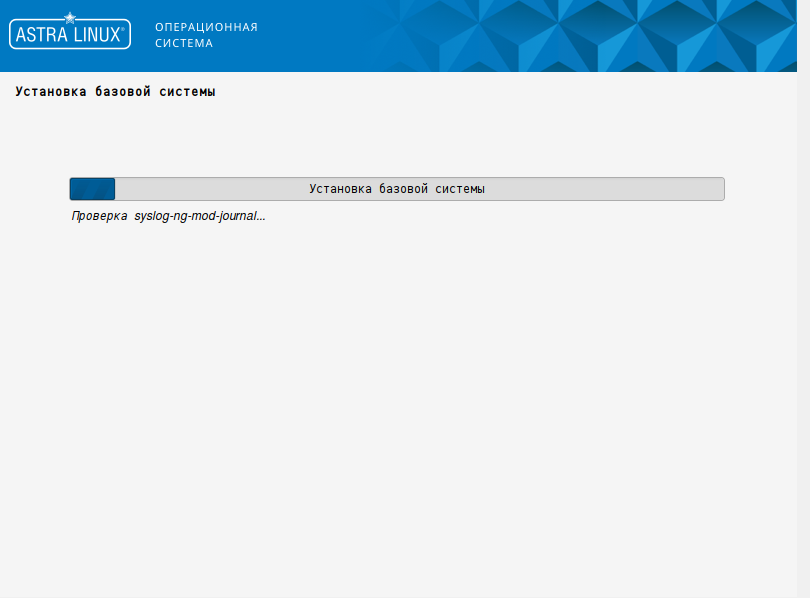
* Разметка дисков – Автоматическая

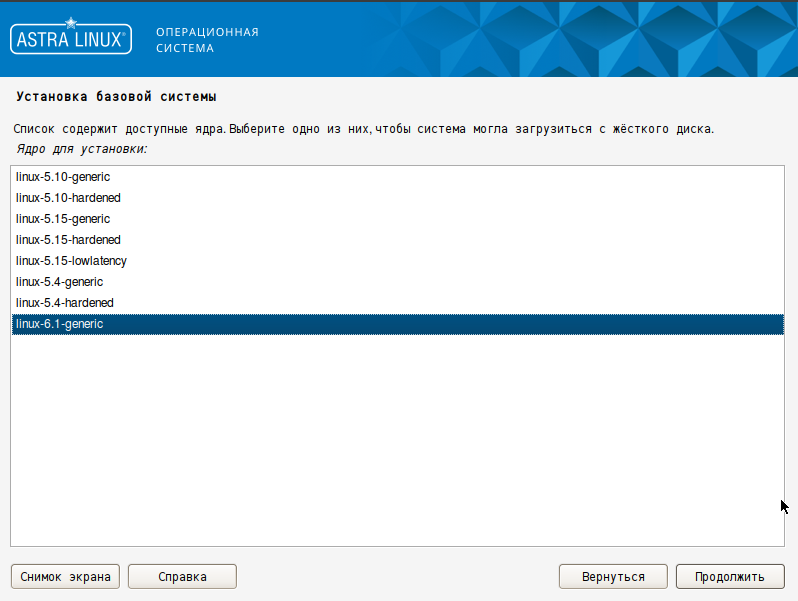


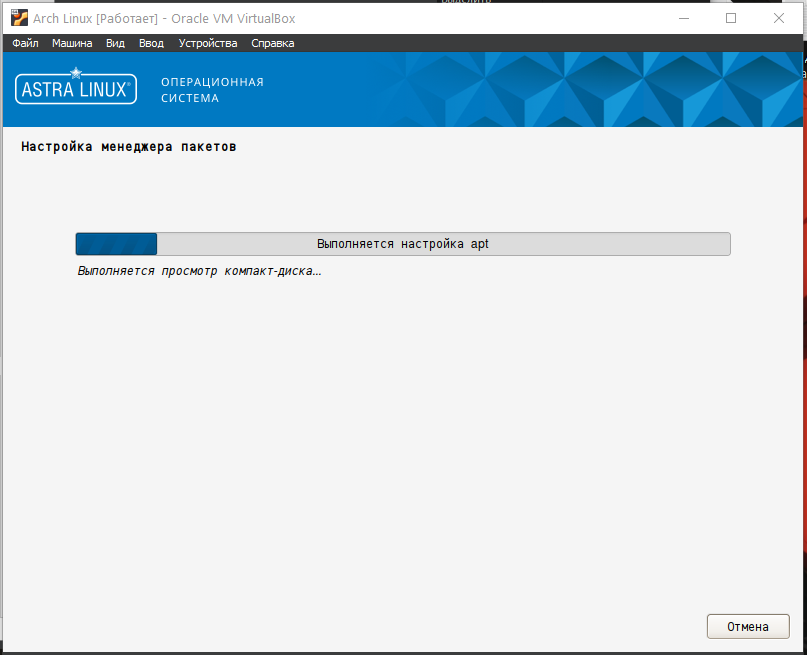




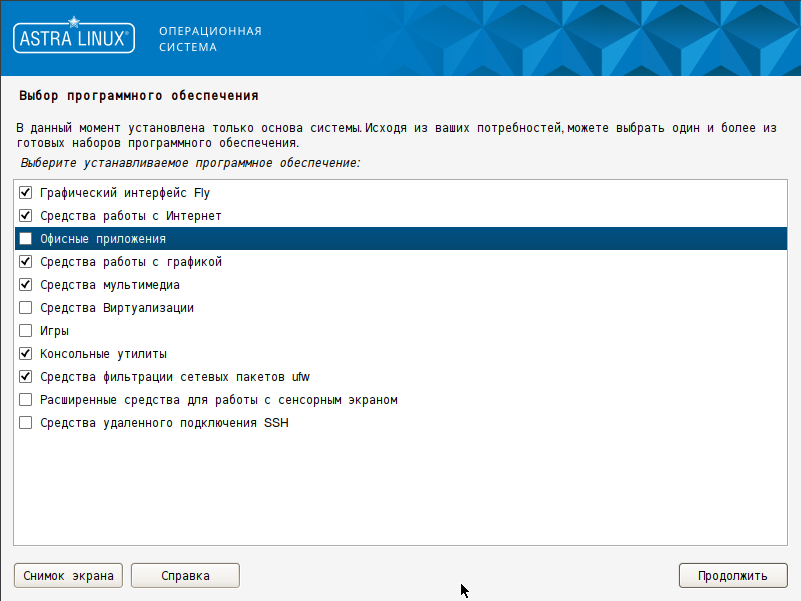
* Установка базового ПО (без участия пользователя)

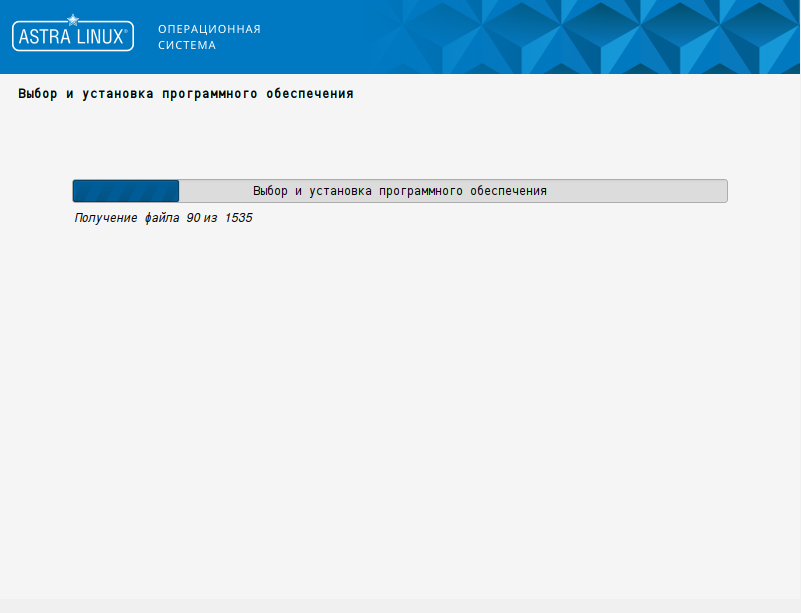




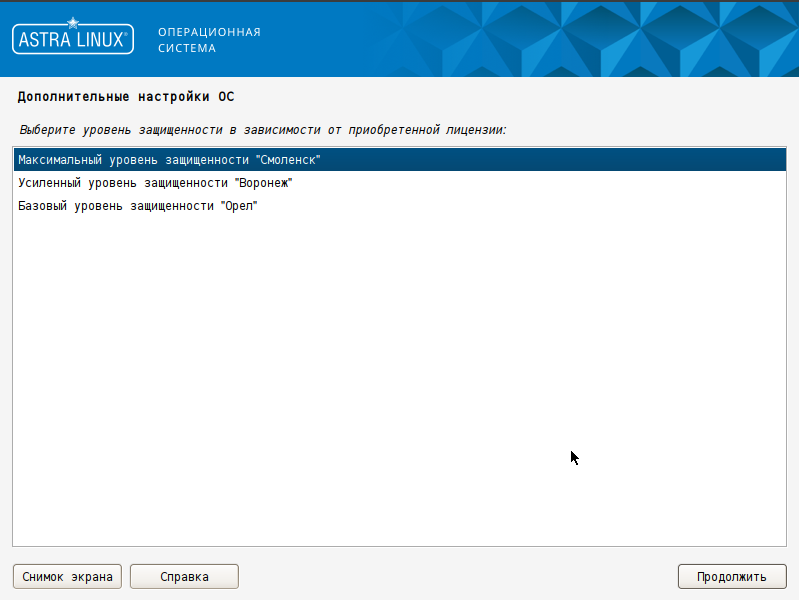


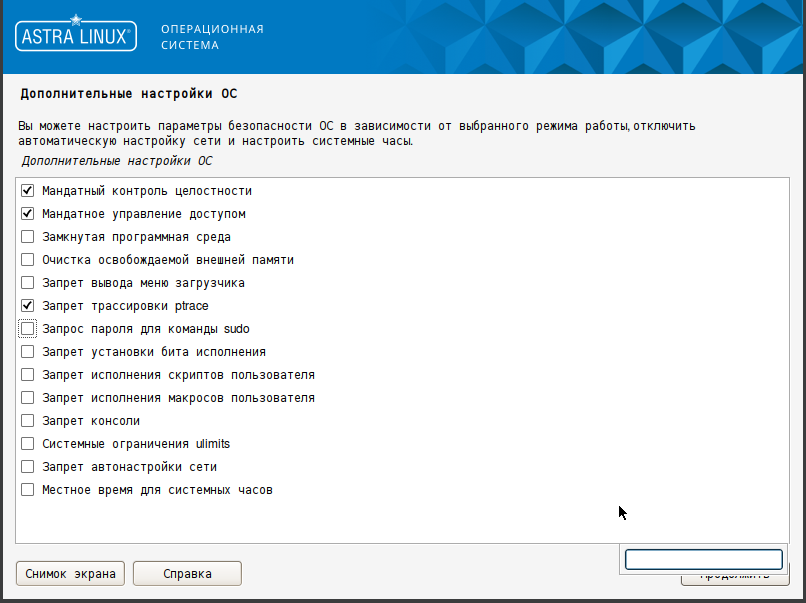
* Выбор дополнительного ПО



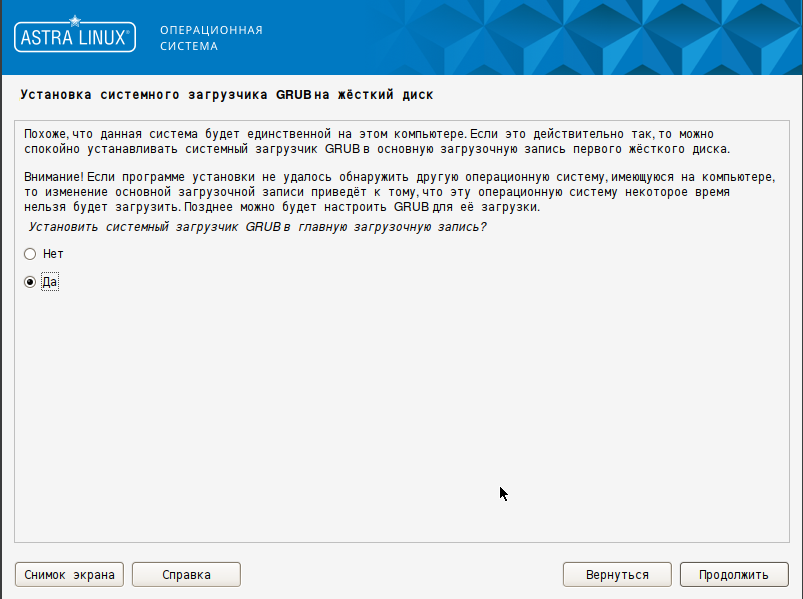


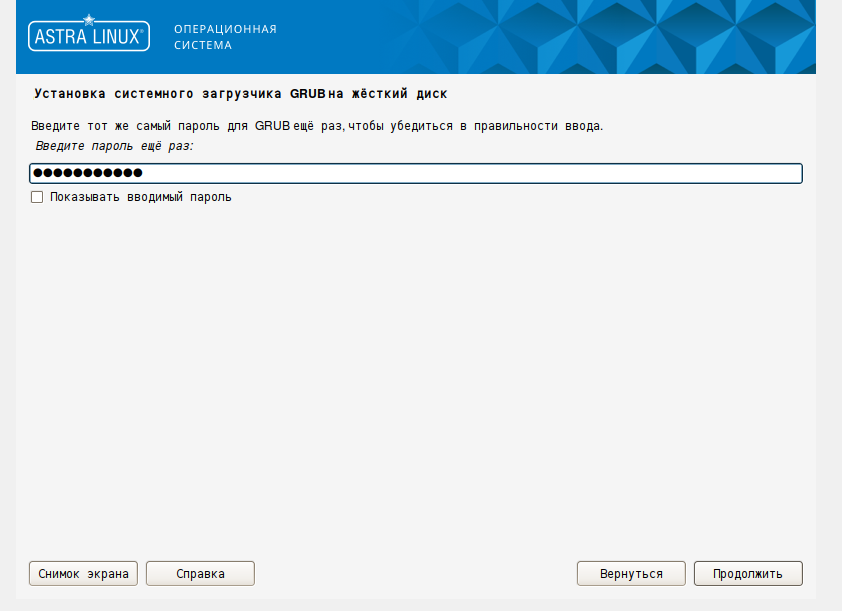
* Дополнительные настройки ОС





* Установка загрузчика системы







2. Процесс загрузки Linux.

Расписать этапы загрузки операционной системы

1. BIOS

BIOS (Basic Input/Output System) запускает компьютер и инициирует процесс загрузки операционной системы. Вот основные шаги, которые BIOS выполняет для запуска системы:

Проверка аппаратной части (POST):

BIOS начинает процесс с выполнения POST - тестирования аппаратного обеспечения (Power-On Self Test). Это включает проверку основных компонентов компьютера, таких как процессор, память (RAM), видеокарта, жесткий диск, клавиатура, и другие важные устройства. Если находятся какие-либо проблемы, BIOS может выдать звуковой сигнал или сообщение об ошибке.

Инициализация аппаратного обеспечения:

После успешного прохождения POST BIOS инициализирует аппаратное обеспечение, включая устройства ввода/вывода (например, клавиатура, мышь, порты), контроллеры устройств (например, контроллеры жестких дисков, контроллеры USB) и другие ключевые компоненты.

Поиск загрузочного устройства:

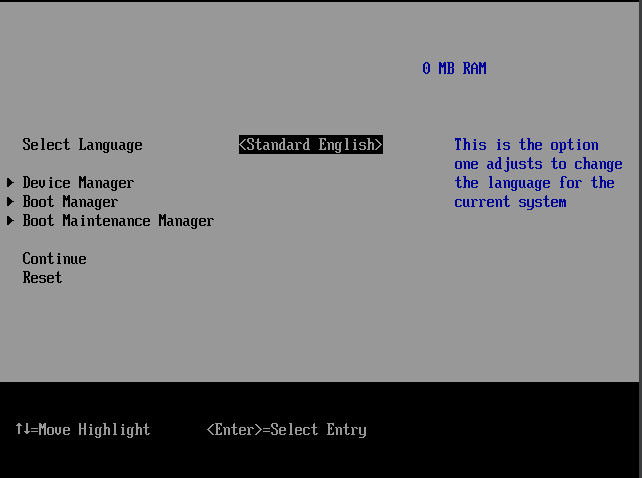
После завершения POST BIOS начинает поиск загрузочного устройства, на котором находится операционная система. Загрузочное устройство может быть жестким диском, SSD, USB-флешкой, CD/DVD-диском или сетевым устройством в зависимости от настроек BIOS.

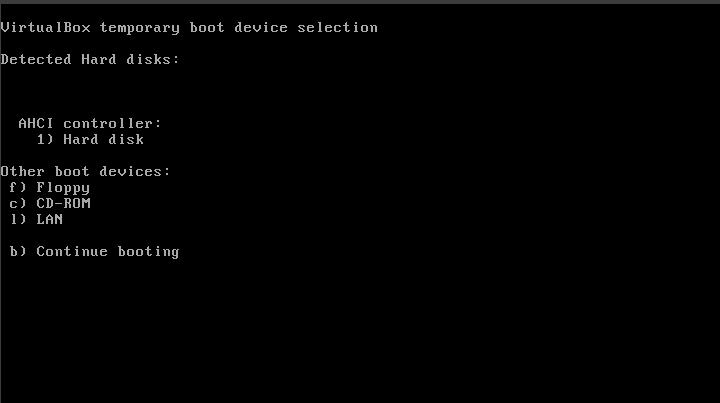
Загрузка загрузчика операционной системы:

Когда BIOS определяет загрузочное устройство, он загружает и запускает загрузчик операционной системы (например, GRUB или NTLDR для Linux и Windows соответственно). Загрузчик операционной системы загружает необходимые файлы операционной системы в оперативную память и передает управление операционной системе.

Загрузка операционной системы:

После передачи управления операционной системе, начинается процесс загрузки операционной системы. Этот процесс включает в себя инициализацию ядра операционной системы, запуск необходимых системных служб и загрузку драйверов устройств.

В результате выполнения этих шагов операционная система готова к использованию, и пользователь может начать работу с компьютером. BIOS играет ключевую роль в этом процессе, обеспечивая правильную и стабильную инициализацию аппаратного обеспечения и загрузку операционной системы.



1. MBR

MBR (Master Boot Record) - это первый сектор (сектор с номером 0) жёсткого диска. Он играет важную роль в процессе загрузки системы, особенно на системах с BIOS.

Расположение загрузчика:

MBR содержит небольшую программу, известную как загрузчик (bootloader). Этот загрузчик выполняет несколько важных задач, включая поиск и загрузку активной раздела (обычно первого раздела на диске) и передачу управления загруженной операционной системе.

Поиск активного раздела:

Загрузчик, хранящийся в MBR, ищет активный раздел на жёстком диске. Активный раздел - это раздел, помеченный как готовый для загрузки. Загрузчик MBR ищет таблицу разделов (обычно в MBR или GPT) и находит активный раздел.

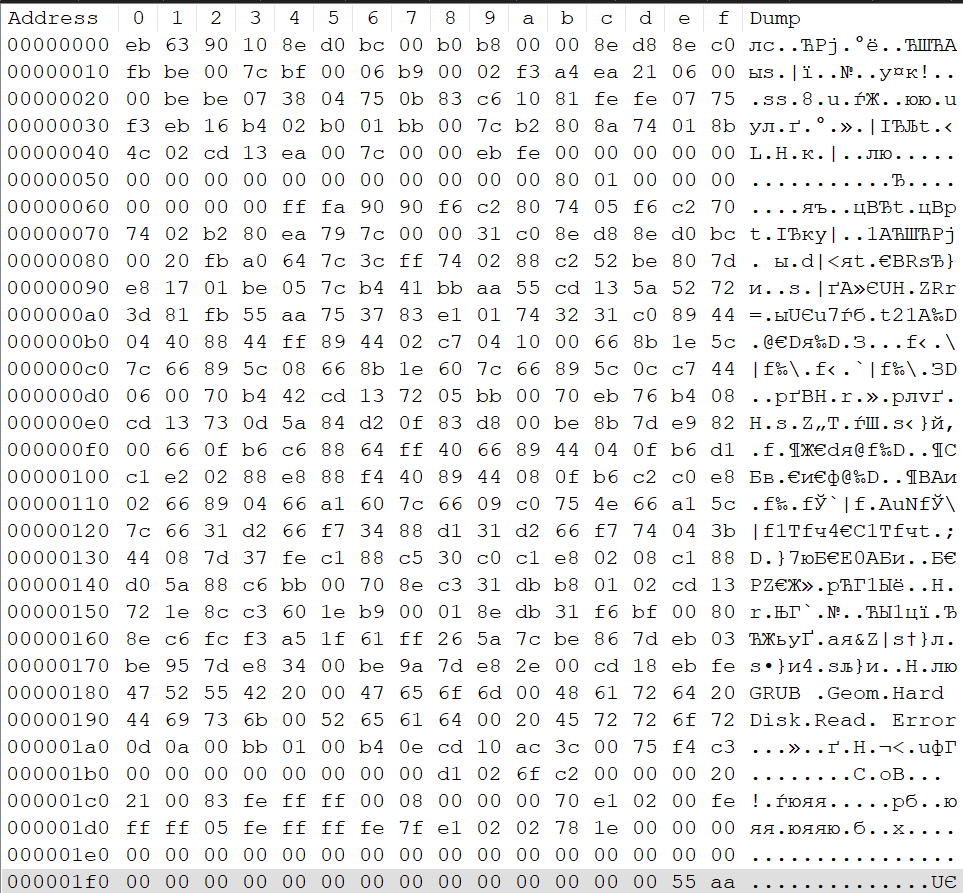
Загрузка загрузчика операционной системы:

Когда загрузчик MBR находит активный раздел, он загружает дополнительный загрузчик, который находится в начале активного раздела. Этот дополнительный загрузчик может быть частью операционной системы или специализированным загрузчиком, таким как GRUB для Linux или NTLDR для Windows.

Передача управления операционной системе:

После загрузки дополнительного загрузчика, управление передается ему. Загрузчик операционной системы продолжает процесс загрузки операционной системы, загружая необходимые файлы и инициализируя ядро операционной системы.

Таким образом, MBR является первым шагом в процессе загрузки операционной системы на компьютерах с BIOS. Он содержит код загрузчика, который инициирует загрузку операционной системы, обеспечивая передачу управления на активный раздел жёсткого диска, где находится дополнительный загрузчик операционной системы.Скриншот первых 512 Байт диска:



1. GRUB

GRUB (GRand Unified Bootloader) - это мощный загрузчик, используемый во многих дистрибутивах Linux и других операционных систем, поддерживающих загрузку с Linux. Он является одним из наиболее распространенных загрузчиков для систем с Linux и многими другими операционными системами.

Вот некоторые ключевые особенности GRUB:

Гибкость конфигурации:

GRUB предоставляет широкие возможности для настройки процесса загрузки. Его конфигурационный файл позволяет настраивать параметры загрузки операционной системы, выбирать различные ядра и опции загрузки, а также управлять загрузочными разделами.

Многосистемный загрузчик:

GRUB позволяет загружать несколько операционных систем с одного компьютера. Он автоматически обнаруживает другие установленные операционные системы и добавляет их в меню выбора загрузки.

Графический интерфейс:

Существует возможность настройки GRUB для отображения графического меню выбора загрузки, что делает процесс загрузки более удобным для пользователей.

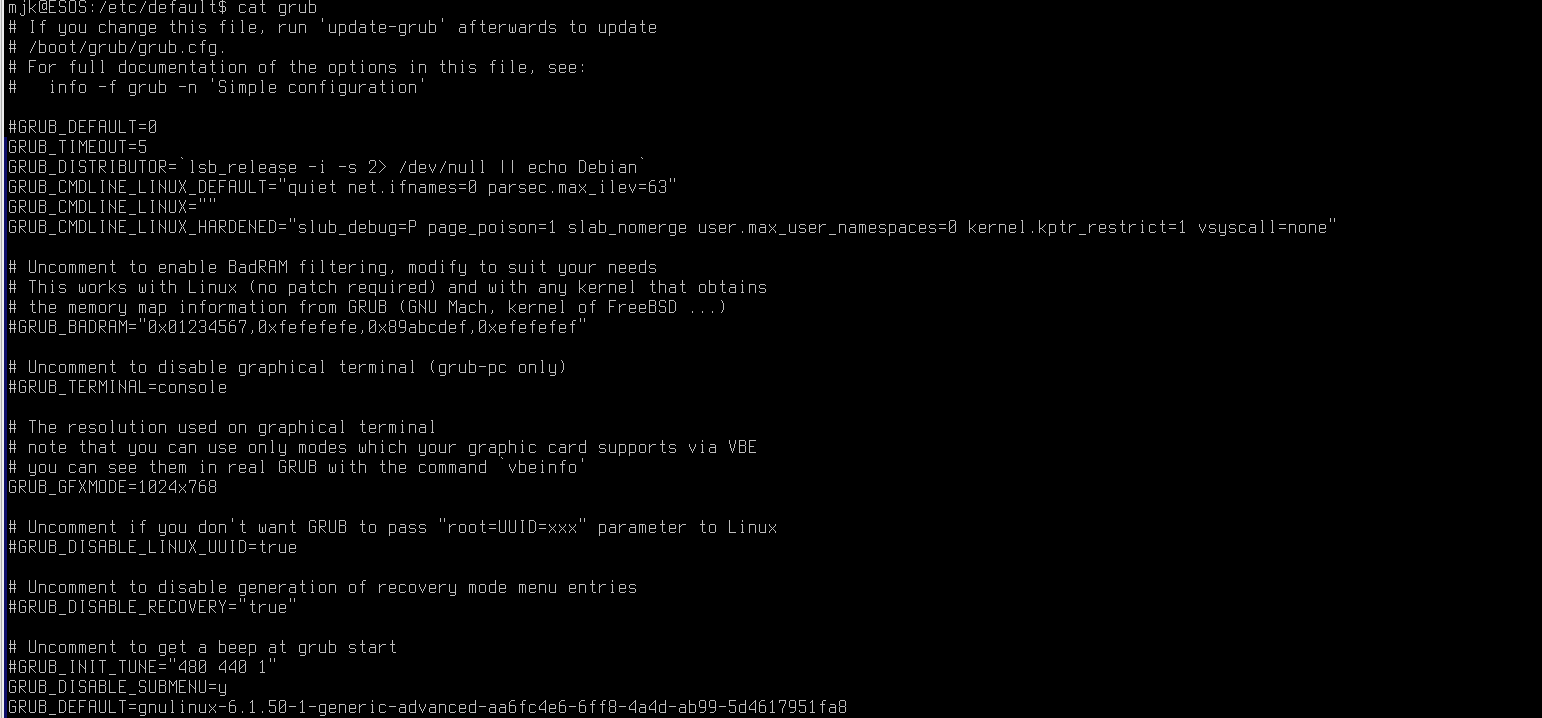
Поддержка множества файловых систем:

GRUB способен загружать операционные системы с различных типов файловых систем, таких как ext2, ext3, ext4, btrfs, NTFS, FAT и многих других.

Модульная архитектура:

GRUB имеет модульную архитектуру, что позволяет добавлять дополнительные функции и драйверы при необходимости, не усложняя сам загрузчик.

GRUB обычно устанавливается в MBR (Master Boot Record) диска и отвечает за начало процесса загрузки операционной системы. После запуска он отображает меню выбора, где пользователь может выбрать операционную систему для загрузки или настроить параметры загрузки..





1. Ядро или Kernel

Ядро операционной системы (Kernel) играет ключевую роль в процессе запуска операционной системы. Вот как оно участвует в этом процессе:

Инициализация железа:

После того, как загрузчик (например, GRUB) загрузил ядро операционной системы в оперативную память, первой задачей ядра является инициализация аппаратного обеспечения компьютера. Это включает в себя обнаружение и настройку процессора, памяти, устройств ввода/вывода и других устройств.

Создание первичного процесса (init):

Ядро создает первичный процесс, обычно называемый init. Init является первым процессом, который запускается ядром и является родительским процессом для всех остальных процессов в системе.

Запуск системных служб и драйверов:

После создания первичного процесса ядро начинает запускать различные системные службы и драйверы, необходимые для работы операционной системы. Это включает службы, управляющие сетью, файловой системой, управлением памятью, планировщиком задач и так далее.

Загрузка пользовательского пространства:

После инициализации системных служб и драйверов ядро начинает загружать пользовательское пространство операционной системы. Это включает в себя загрузку пользовательских программ, библиотек и других ресурсов, необходимых для работы пользовательского интерфейса.

Передача управления пользовательским процессам:

По мере завершения инициализации ядро передает управление пользовательским процессам, начиная с процесса init. Пользовательские процессы могут быть запущены автоматически при загрузке системы или в результате действий пользователя.

В итоге, ядро операционной системы является центральным компонентом, управляющим всеми аспектами работы операционной системы, начиная с инициализации аппаратного обеспечения и заканчивая выполнением пользовательских программ. Оно обеспечивает связь между аппаратным обеспечением компьютера и пользовательскими приложениями, обеспечивая стабильную и эффективную работу операционной системы.

1. Init

Процесс init (Initialization) является первым пользовательским процессом, который запускается в операционной системе после инициализации ядра (kernel). Вот как он работает в процессе запуска операционной системы:

Инициализация ядра:

После того как загрузчик (например, GRUB) загрузил ядро операционной системы в память, начинается процесс инициализации ядра.

Создание процесса init:

После инициализации ядра, ядро создает процесс init. Этот процесс имеет идентификатор PID 1 и является родительским процессом для большинства других процессов, запускаемых в системе.

Запуск системных служб:

После создания процесса init, он начинает запускать системные службы и процессы, необходимые для функционирования операционной системы. Это может включать службы, управляющие сетью, файловой системой, обменом данными и другими аспектами системы.

Чтение конфигурационных файлов:

Во многих дистрибутивах Linux, процесс init читает конфигурационные файлы (например, /etc/inittab или /etc/init) для определения, какие процессы и службы должны быть запущены при старте системы и в каком порядке.

Передача управления остальным процессам:

После того, как init завершил запуск системных служб и процессов, он передает управление остальным процессам и приложениям, которые могут быть запущены в системе. Это может включать в себя запуск графического интерфейса, менеджера входа пользователя и других пользовательских приложений.

Процесс init отвечает за поддержание стабильности работы системы и запуск всех необходимых компонентов, необходимых для корректной работы операционной системы. В современных системах init часто заменяется менеджерами инициализации, такими как systemd в большинстве дистрибутивов Linux, который предоставляет более мощные и гибкие средства управления процессами и службами.

1. Уровень выполнения программ (Runlevel)

Runlevel (уровень запуска) - это состояние операционной системы, определяющее набор работающих процессов и служб. В системах Unix и подобных им, таких как Linux, уровни запуска определяют, какие процессы и службы запускаются при старте системы.

Обычно в Unix-подобных системах существует несколько стандартных уровней запуска. Вот некоторые из них:

Runlevel 0:

Этот уровень обычно соответствует выключению системы. В этом состоянии останавливаются все процессы и службы.

Runlevel 1 или S (Single-user mode):

Он обычно используется для административных задач и обслуживания. В этом режиме запускается минимальный набор процессов и служб, а обычно не требуется аутентификации пользователя.

Runlevel 2:

Это обычно многопользовательский режим без поддержки сети.

Runlevel 3:

Многопользовательский режим с поддержкой сети. В этом режиме запускаются все необходимые службы для работы сети.

Runlevel 4:

Не используется во многих дистрибутивах Linux, оставлен для пользовательской конфигурации.

Runlevel 5:

Обычно это многопользовательский режим с графическим интерфейсом. В этом режиме запускаются графический сервер и менеджер входа в систему (например, GNOME, KDE).

Runlevel 6:

Этот уровень обычно соответствует перезагрузке системы. В этом состоянии останавливаются все процессы и службы, а затем система перезагружается.

Управление уровнями запуска в основном осуществляется с помощью файла конфигурации, такого как /etc/inittab в системах до systemd или с помощью инструментов, предоставляемых менеджером инициализации, например, systemctl в системах, использующих systemd. Все это позволяет администраторам систем гибко настраивать, какие сервисы запускаются при старте системы в зависимости от их потребностей.

3. Работа в терминале.

1 Расписать типы терминалов:

Аппаратный терминал

Аппаратный терминал - это устройство, которое обеспечивает ввод и вывод данных для компьютера. Он состоит из клавиатуры для ввода данных и монитора (или экрана) для вывода информации. В качестве коммуникационных портов раньше широко использовались COM-порты (RS-232), сейчас USB.

Принцип работы терминала:

Ввод данных:

Пользователь вводит данные с помощью клавиатуры терминала. Каждая клавиша, нажатая пользователем, генерирует электрический сигнал, который передается терминалу.

Передача данных компьютеру:

Электрические сигналы, сгенерированные нажатиями клавиш, поступают на интерфейс терминала, который затем передает их компьютеру через соответствующий порт (например, порт RS-232).

Обработка данных компьютером:

Компьютер получает данные от терминала через порт ввода-вывода. Эти данные могут быть обработаны различными программами, операционной системой или интерфейсом командной строки.

Вывод данных:

Компьютер отправляет данные на монитор терминала для отображения пользователю. Это может быть текстовая информация, графика или любой другой тип данных, который компьютер может обработать.

Отображение данных на экране:

Монитор терминала получает данные от компьютера и отображает их на экране. Это позволяет пользователю видеть информацию, которую он вводит с клавиатуры или которую выводит компьютер.

Виртуальный терминал

Виртуальный терминал - это эмулированный терминал, который работает внутри компьютерной операционной системы и позволяет пользователям взаимодействовать с системой через текстовый интерфейс командной строки.

Операционные системы, такие как Linux и Unix, предоставляют механизм виртуальных терминалов для управления несколькими текстовыми консолями с помощью клавиатуры и монитора.

Множество консолей: Виртуальные терминалы позволяют пользователям иметь несколько текстовых консолей, каждая из которых работает независимо. Это полезно, когда требуется одновременно выполнять несколько задач или вести работу в различных окружениях.

Переключение между консолями: Пользователь может переключаться между виртуальными терминалами, используя сочетания клавиш или команды операционной системы. Например, в Linux часто используются клавиши Alt+Fn (где Fn - номер виртуального терминала), чтобы переключаться между терминалами.

Многопользовательская поддержка: Виртуальные терминалы могут поддерживать множество пользователей, каждый из которых может взаимодействовать с системой через свой собственный текстовый интерфейс.

Отслеживание системных журналов: Виртуальные терминалы могут использоваться для отслеживания системных журналов, выполнения административных задач и управления системой в режиме реального времени.

Удаленный доступ: Виртуальные терминалы также могут быть использованы для удаленного доступа к системе по протоколам, таким как SSH (Secure Shell), что позволяет пользователям работать с системой удаленно через интернет.

В целом, виртуальные терминалы являются мощным инструментом для работы с операционной системой через текстовый интерфейс, особенно в ситуациях, когда требуется эффективное управление и мониторингом системы без использования графического интерфейса.

Псевдотерминал

Псевдотерминал (Pseudo Terminal, PTY) - это механизм, используемый в операционных системах Unix и подобных им для создания виртуальных терминалов и связи между процессами.

Создание пары терминалов:

Псевдотерминал создает пару устройств: мастер и его подчиненный. Мастер является интерфейсом, через который процесс может взаимодействовать с подчиненным терминалом.

Связь между процессами:

Мастер и подчиненный терминалы соединены друг с другом. Процесс, работающий с мастер терминалом, может передавать данные в подчиненный терминал, и наоборот.

Использование в командных оболочках и интерактивных приложениях:

Псевдотерминалы широко используются для обеспечения интерактивного взаимодействия с командными оболочками (например, bash, zsh) и другими интерактивными приложениями, которые требуют ввода данных пользователя.

Удаленный доступ:

PTY также используется для удаленного доступа к системе через протоколы, такие как SSH. При этом удаленное приложение или пользователь взаимодействует с псевдотерминалом на удаленной машине, как если бы он был физическим терминалом.

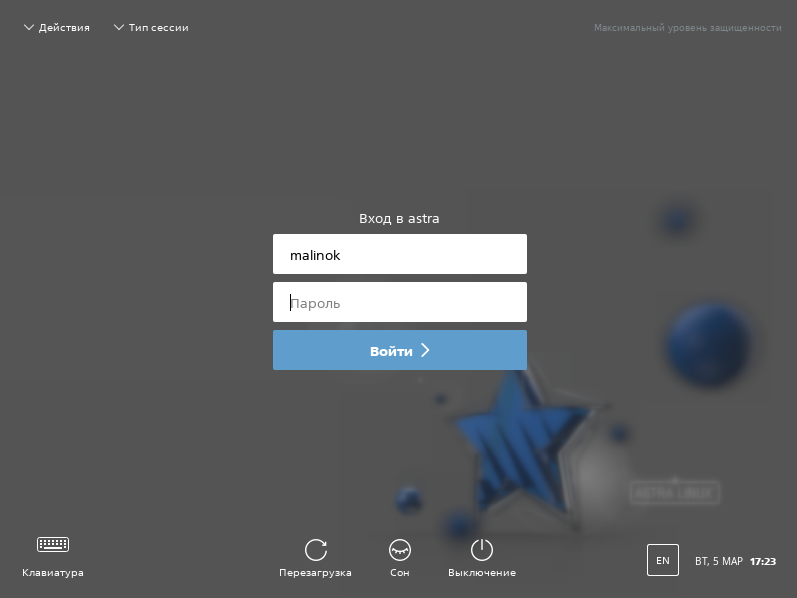
Имитация терминала:

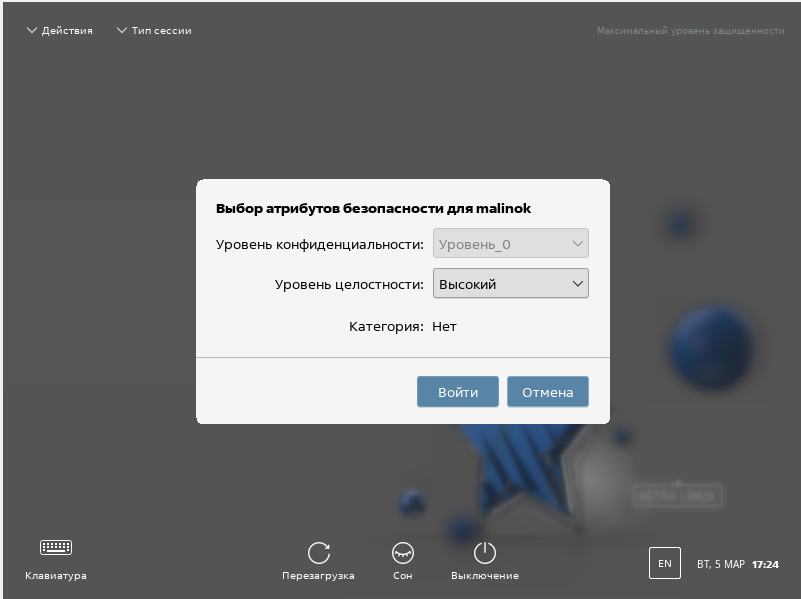
Псевдотерминалы предоставляют программам и процессам интерфейс, который похож на реальный терминал, что позволяет им работать с текстовыми и вводом-выводом, как если бы они работали с физическим устройством.

В целом, псевдотерминалы играют важную роль в создании интерактивных средств взаимодействия пользователя с системой в операционных системах Unix и подобных им. Они позволяют создавать виртуальные терминалы, обеспечивая гибкость и эффективность в управлении процессами и вводом-выводом данных.

1. Выполнить настройку терминала:

1 Осуществите вход в систему через графический экранный менеджер

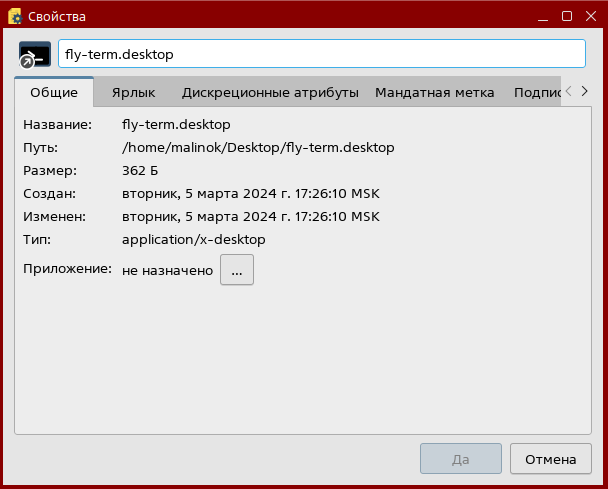


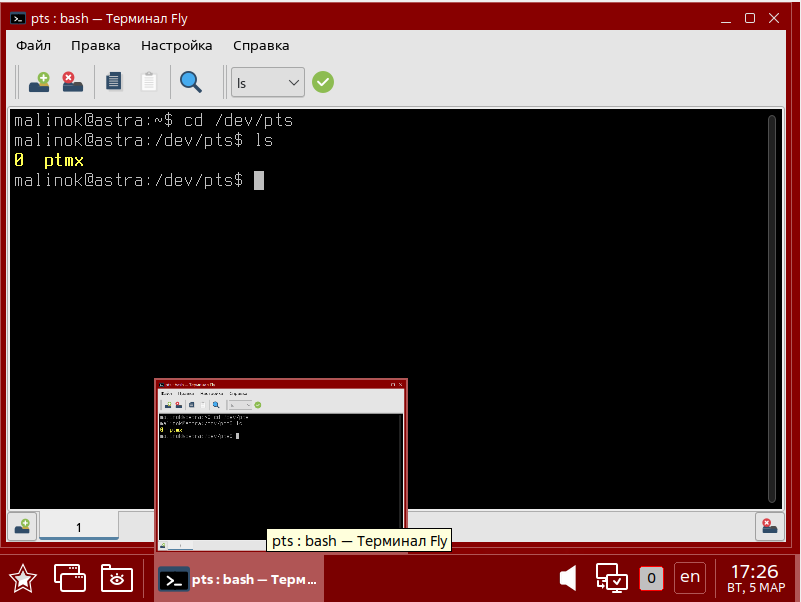


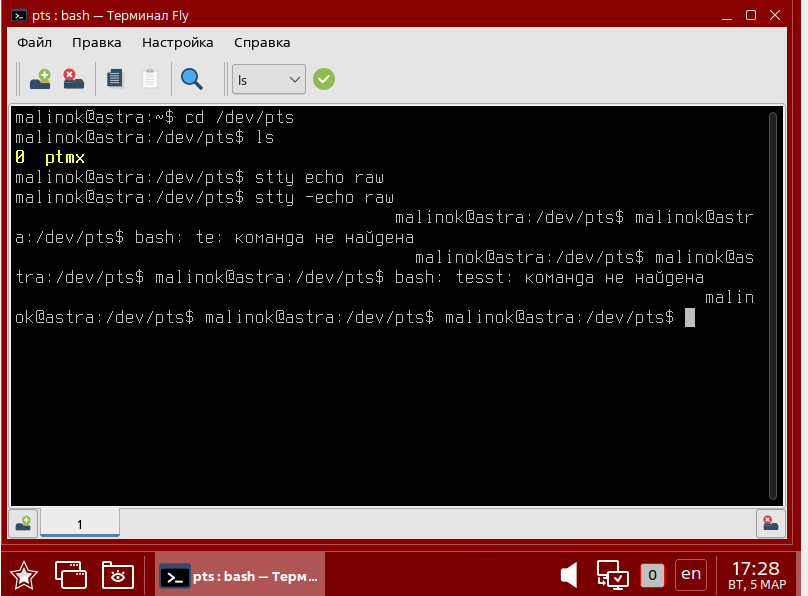
2 Запустите «Терминал Fly».

1. Узнайте название запущенного терминала. К какому типу относится данный терминал.

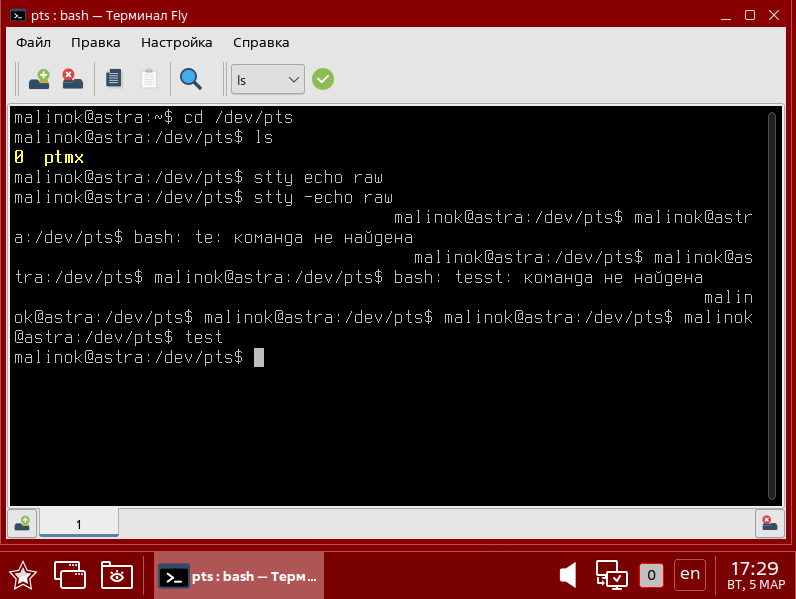
Псевдотериминал fly-term:



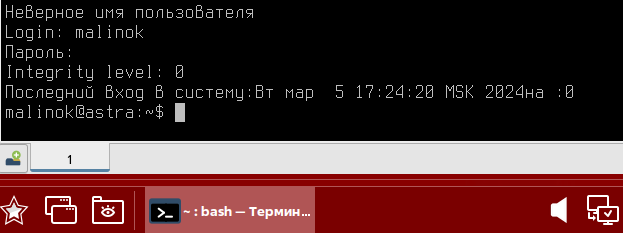
1. Посмотрите содержимое каталога /dev/pts. 
2. На второй вкладке наберите: stty -echo raw



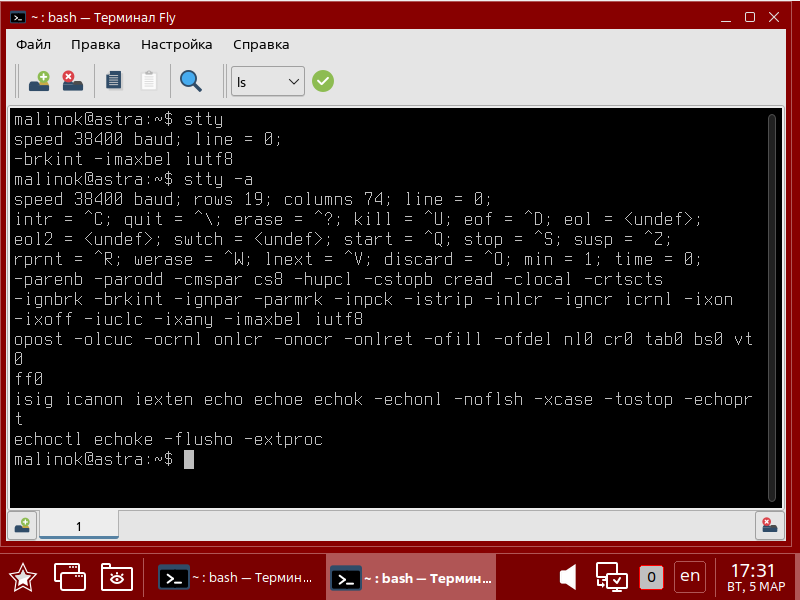
1. Восстановите нормальный вид терминала на второй вкладке



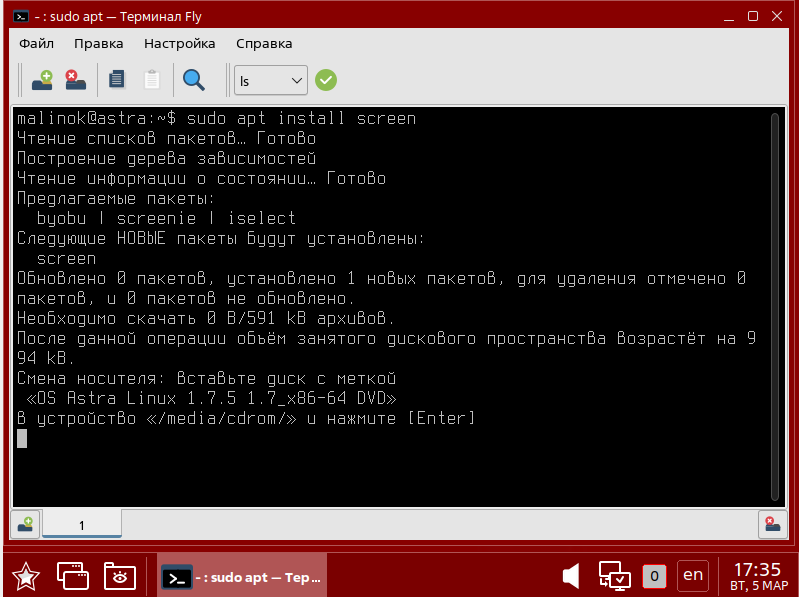
1. Перейдите на третий виртуальный терминал и войдите в систему.



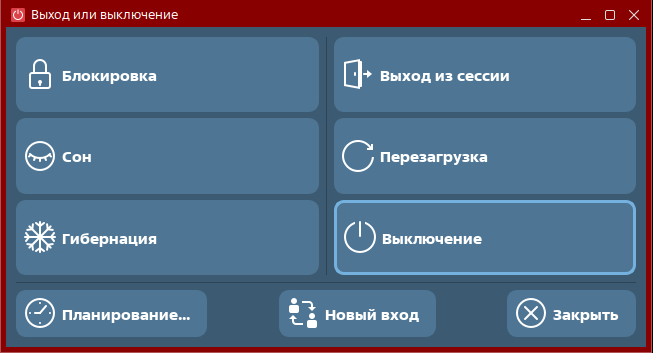
1. Узнайте количество строк и столбцов на текстовом терминале

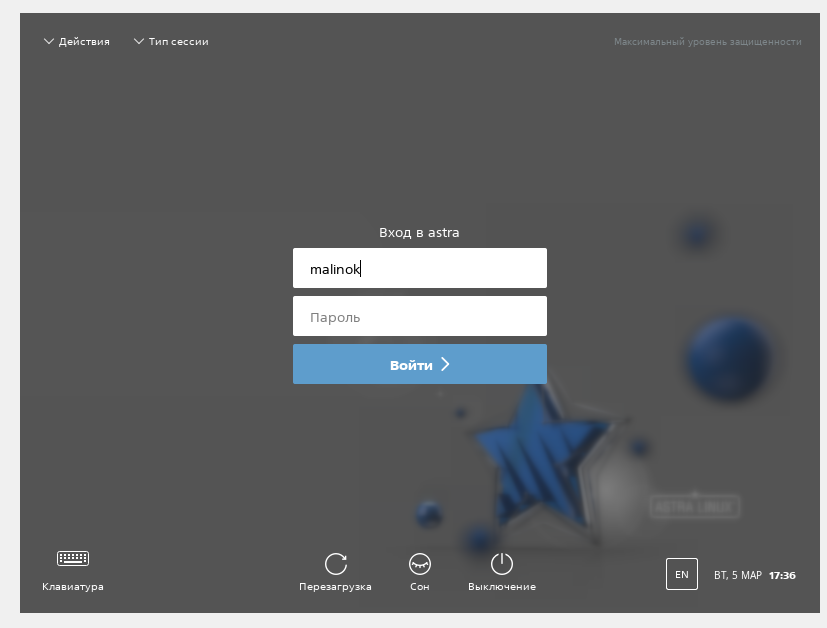


1. Утилита screen



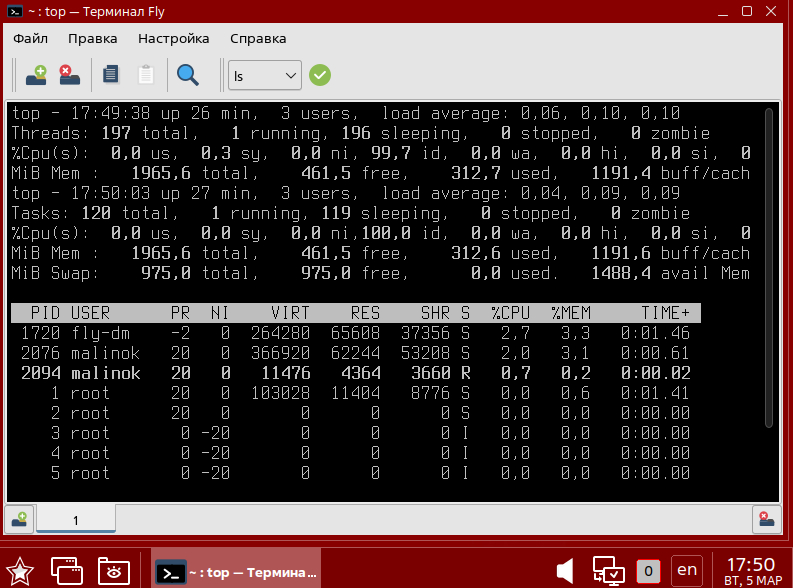
1. Вход в систему и выход из системы



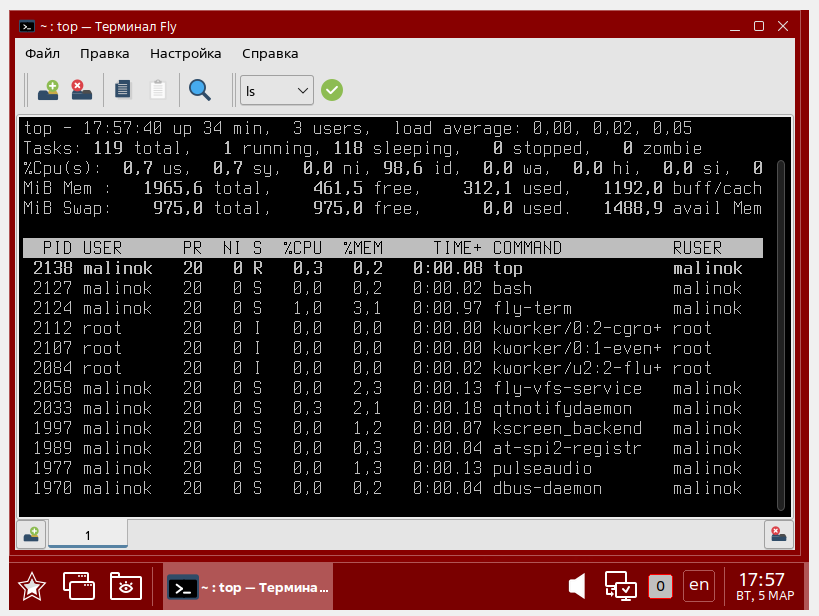


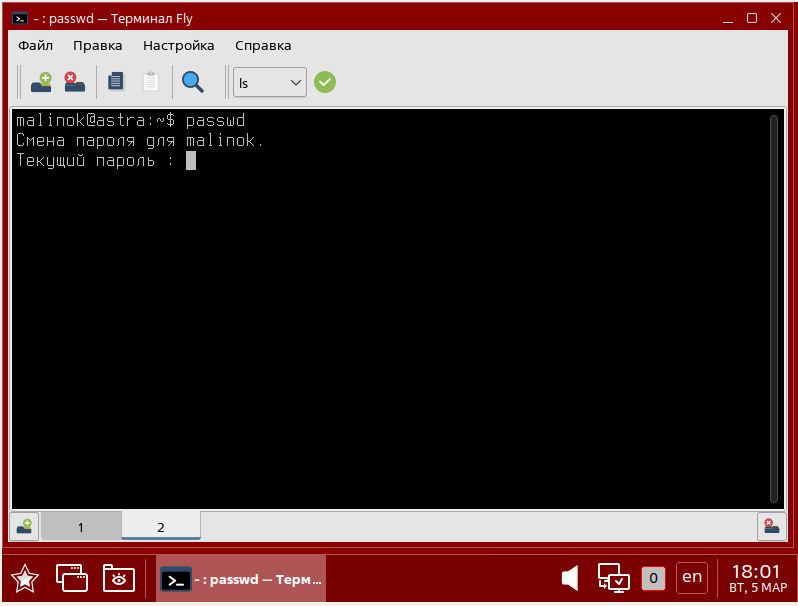
4. Процессы в Linux

1 Посчитайте количество процессов, имеющих несколько потоков выполнения.

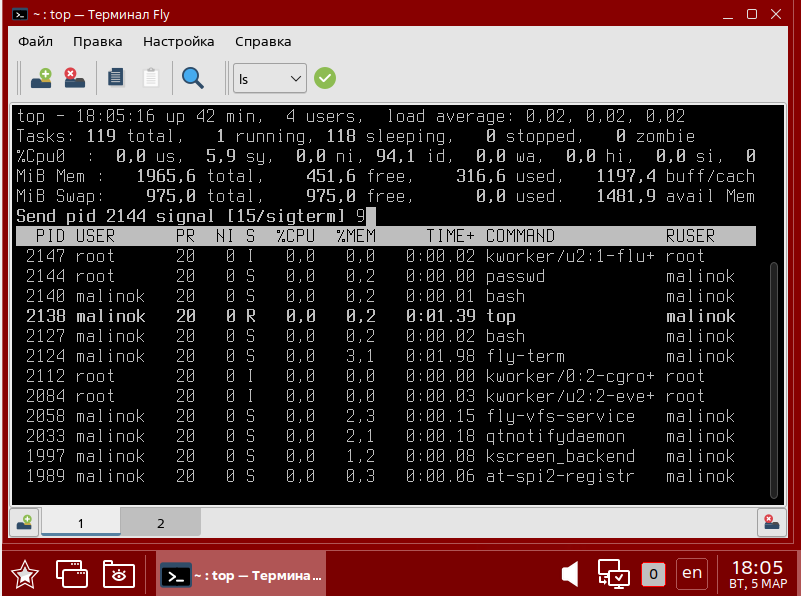
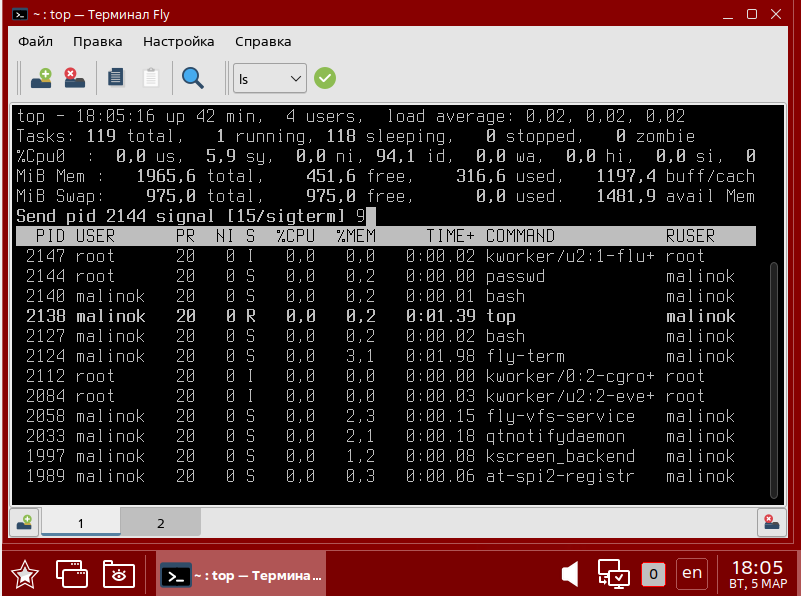
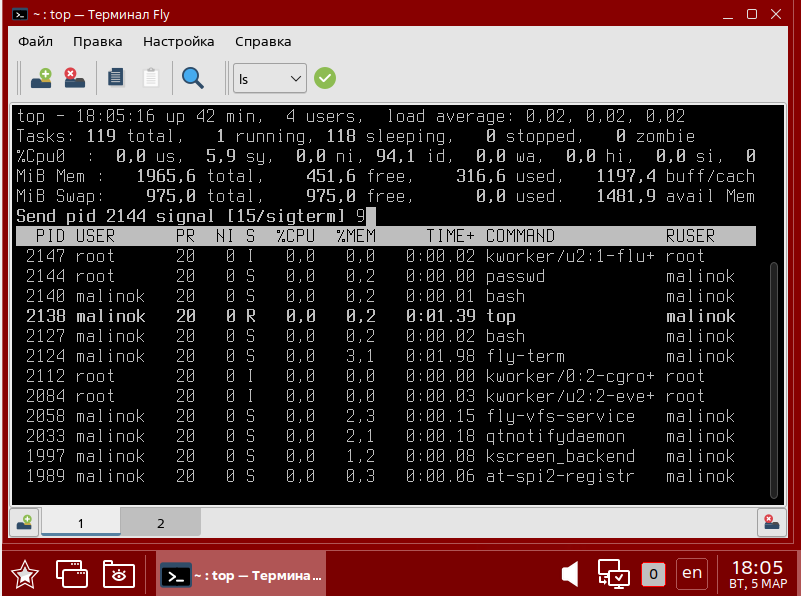


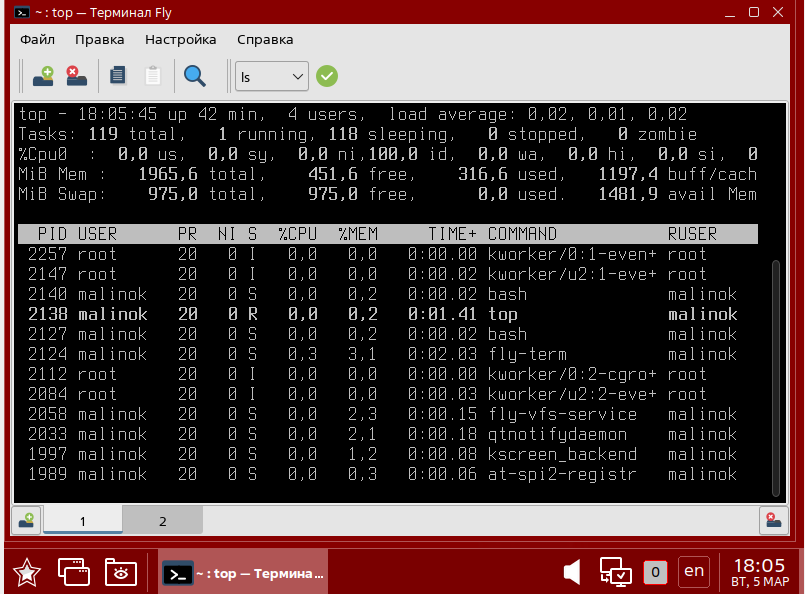
2 Запустите top и настройте вывод полей с информацией о процессе следующим образом:

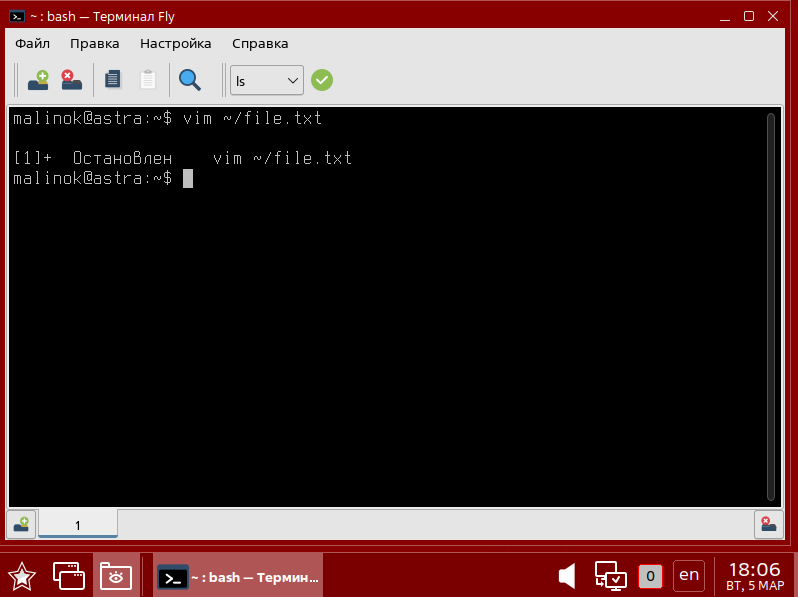
3 В другом терминальном окне выполните команду passwd и оставьте ее в состоянии запроса текущего пароля.



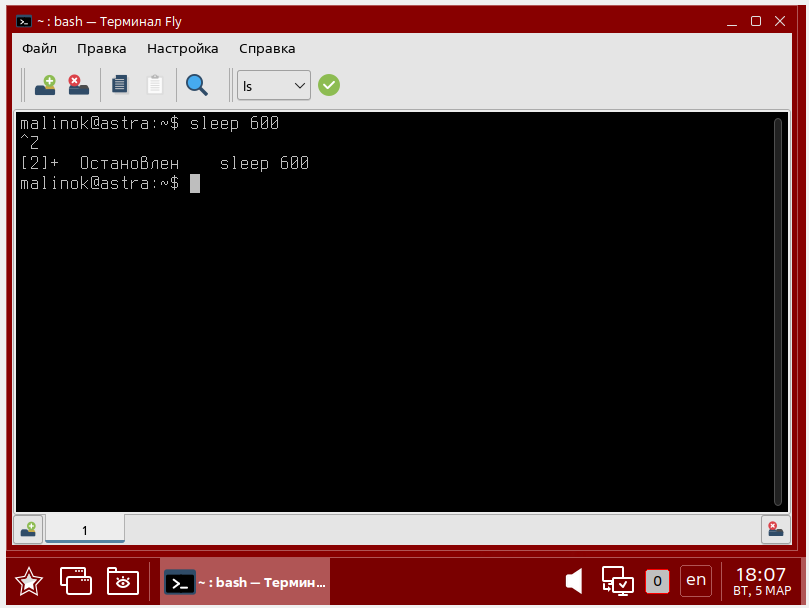
4 Перейдите в терминальное окно с top и выполните следующие действия:

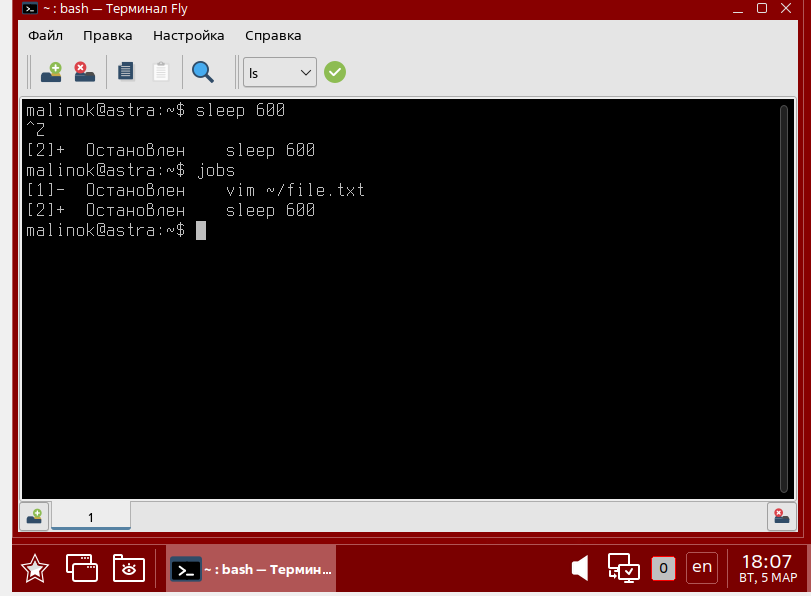


5 Выполните команду vim ~/file.txt и нажмите Ctrl-Z.

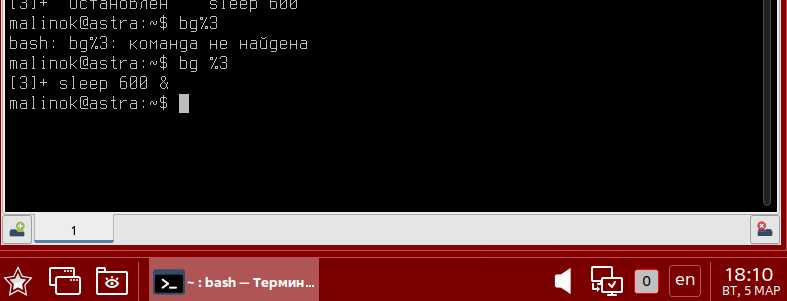


6 Выполните команду sleep 600, нажмите Ctrl-Z и выполните команду jobs.

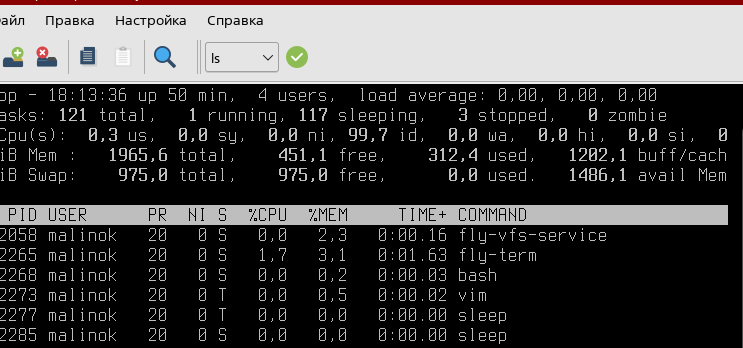




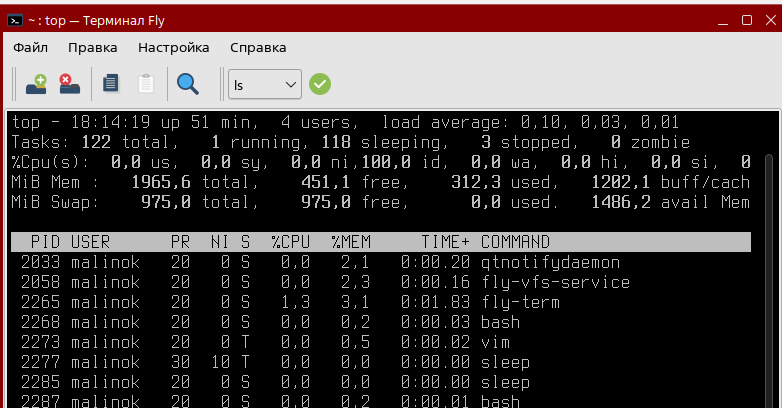
7 Последнее задание (sleep 600) сделайте фоновым.

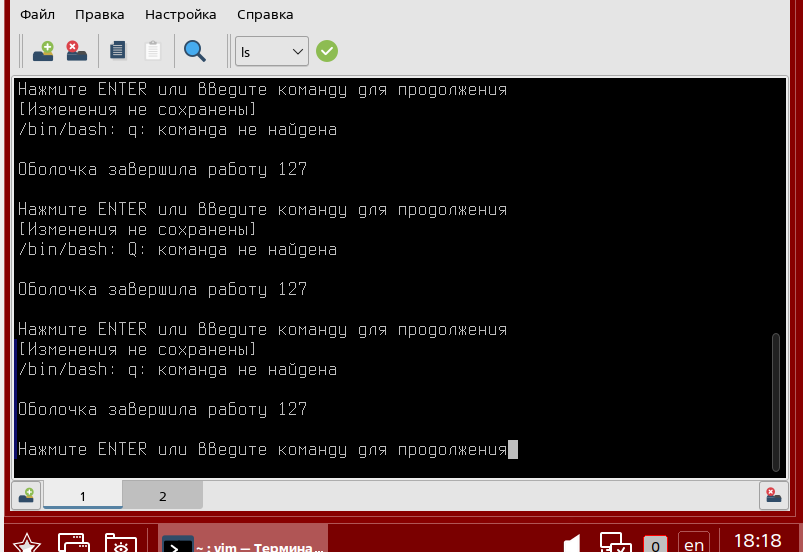


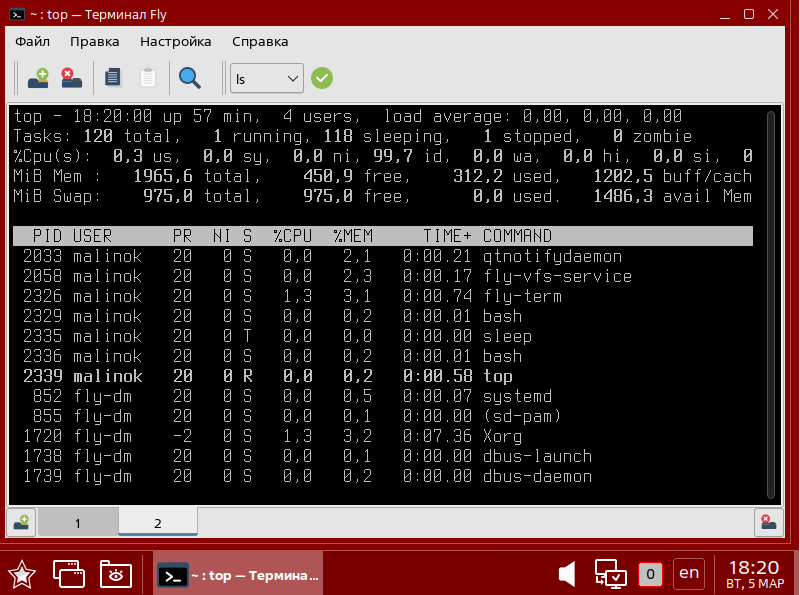
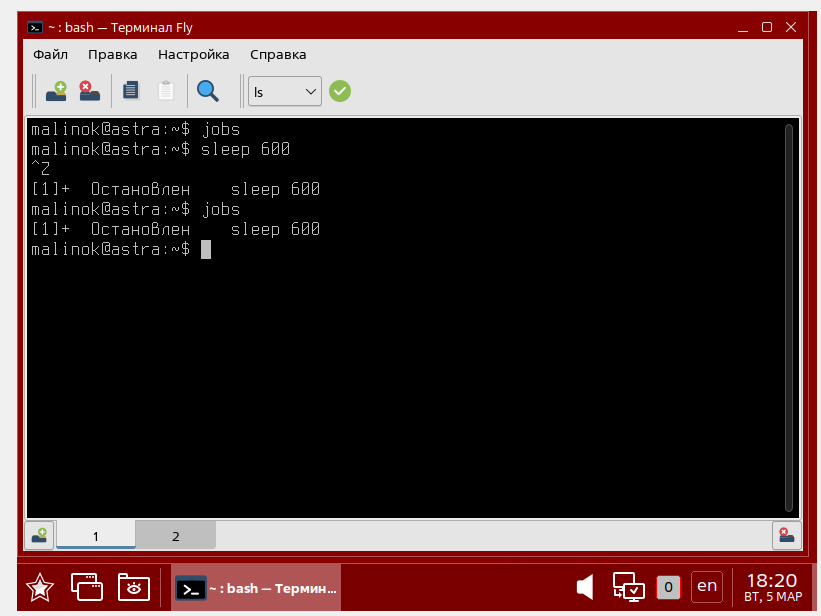
8 Измените число NICE у задания (sleep 600), сделав его равным 10.

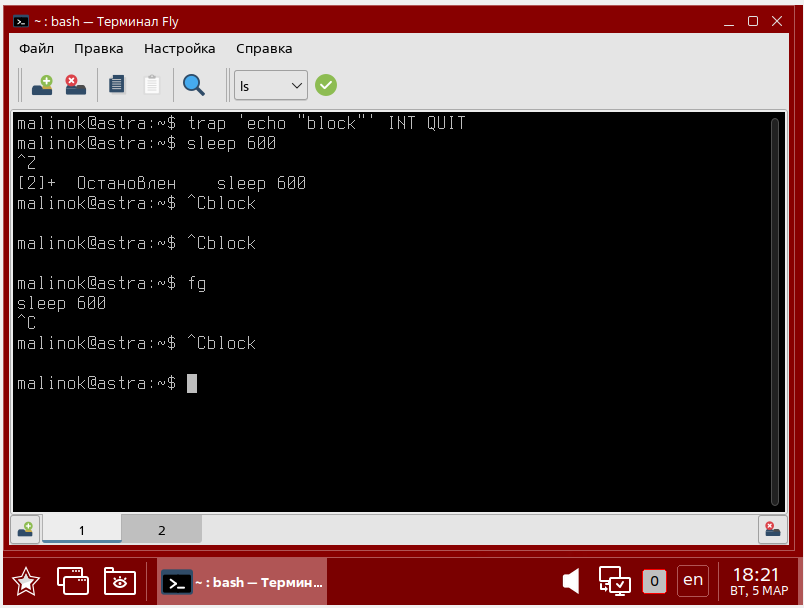


9 Проверьте, что число NICE у этого задания изменилось.



10 Сделайте задание vim ~/file.txt активным и выйдите из редактора. 

11 Отправьте сигнал 15 (SIGTERM) заданию sleep 600 и выполните команду jobs. 

12 Создайте перехватчик сигналов SIGINT и SIGQUIT внутри командного интерпретатора, который выводит сообщение «Сигнал заблокирован». (Используйте встроенную команду trap) 

5. Управление учетными записями пользователей и групп

1 С помощью команд useradd, groupadd, passwd создайте учетную запись user1 со следующими параметрами:

• UID - 1500;

• основная (первичная) группа user1 (GID 1500);

• дополнительная группа - video;

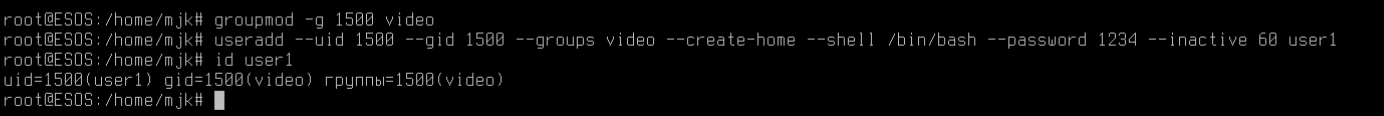
• домашний каталог должен быть создан;

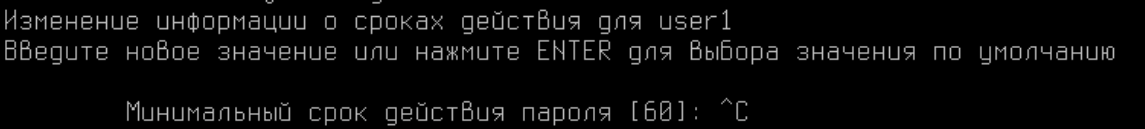
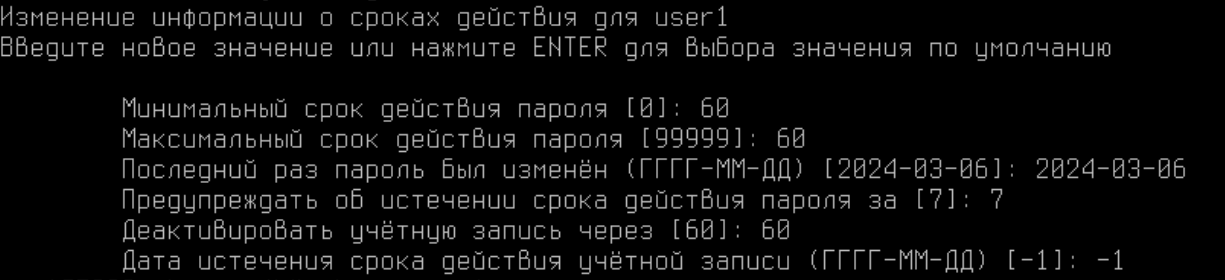
• входной командный интерпретатор - /bin/bash;

• задать пароль по своему усмотрению;

• время действия пароля - 60 дней;

• пользователь должен сменить пароль при первом входе в систему.



2 Проверьте, что атрибуты учетной записи и параметры пароля установлены верно (воспользуйтесь командами id и chage), зайдите в систему, используя созданную учетную запись пользователя. 

3 С помощью утилит adduser и addgroup создайте учетную запись user2 со следующими параметрами:

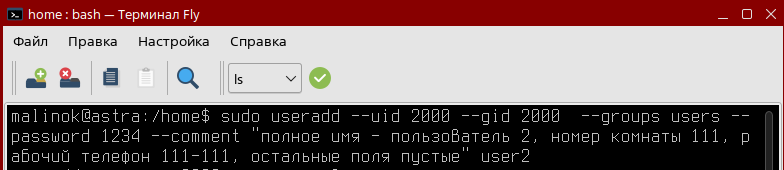
• UID - 2000;

• основная группа user2 (GID 2000);

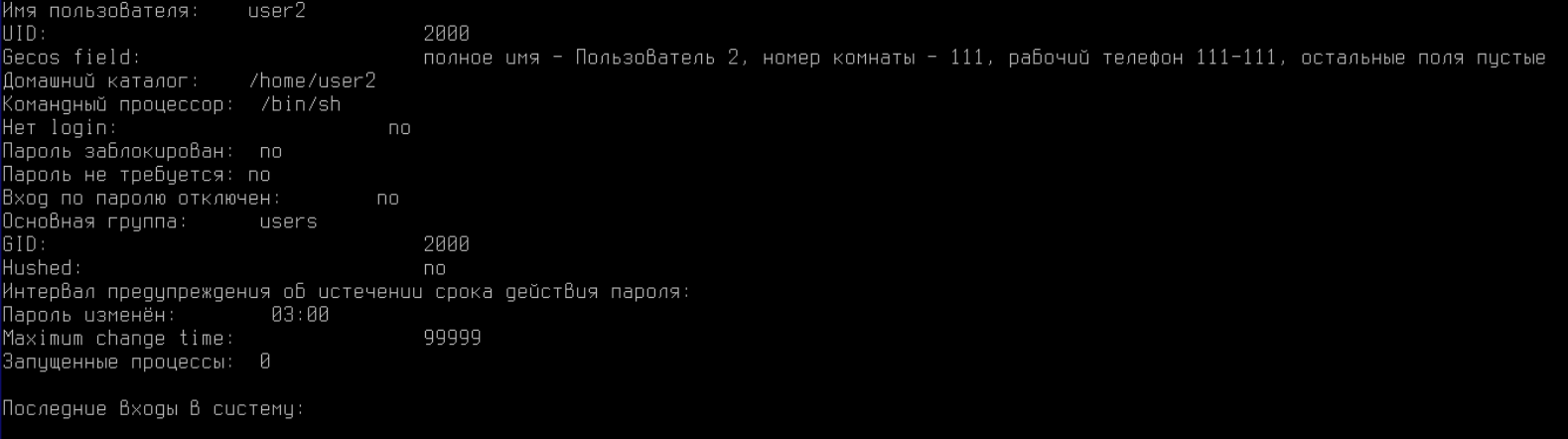
• дополнительная группа users;

• GECOS: полное имя- Пользователь 2, номер комнаты - 111, рабочий телефон 111-111, остальные поля пустые;

• задайте пароль по своему усмотрению.



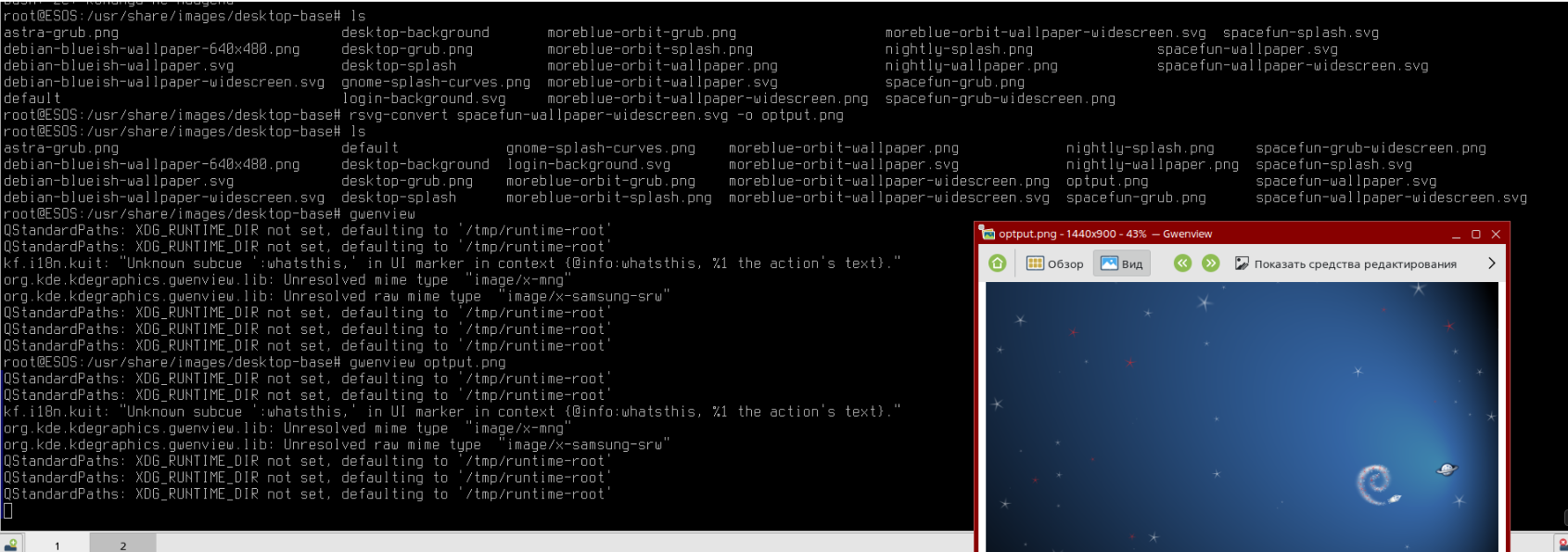
4 Проверьте, что учетная запись создана согласно требованиям из предыдущего пункта (используйте команду lslogins) и зайдите в систему под учетной записью user2.



5 Измените обои у новых пользователей:

• преобразуйте файл /usr/share/images/destop-base/spacefun-wallpaper-widescreen.svg в png формат. Используйте команду rsvg-convert. Сохраните файл в каталоге /usr/share/images под именем spacefun.png.

• Укажите в файле настройки темы имя файла с новыми обоими (параметр WallPaper), обои растянуть на весь экран (параметр WallPaperPos).



6 С помощью графической утилиты (fly-admin-smc) создайте учетную запись user3 со следующими параметрами:

• UID - 2500:

• основная группа user3 (GID 2500);

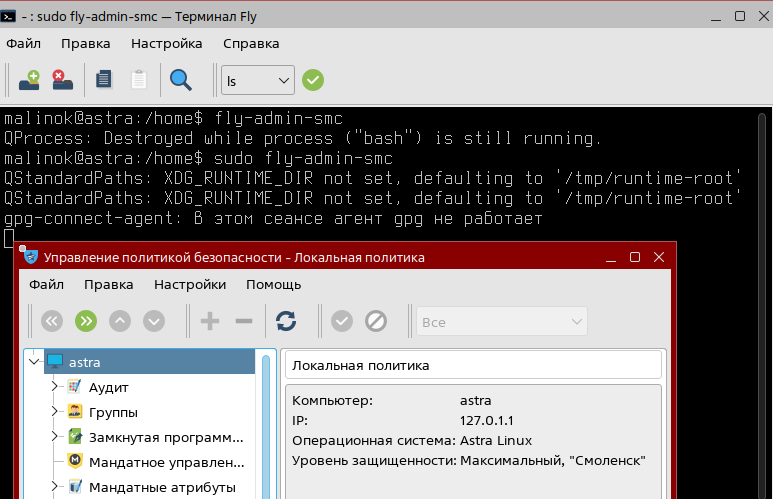
• дополнительные группы: users, cdrom;

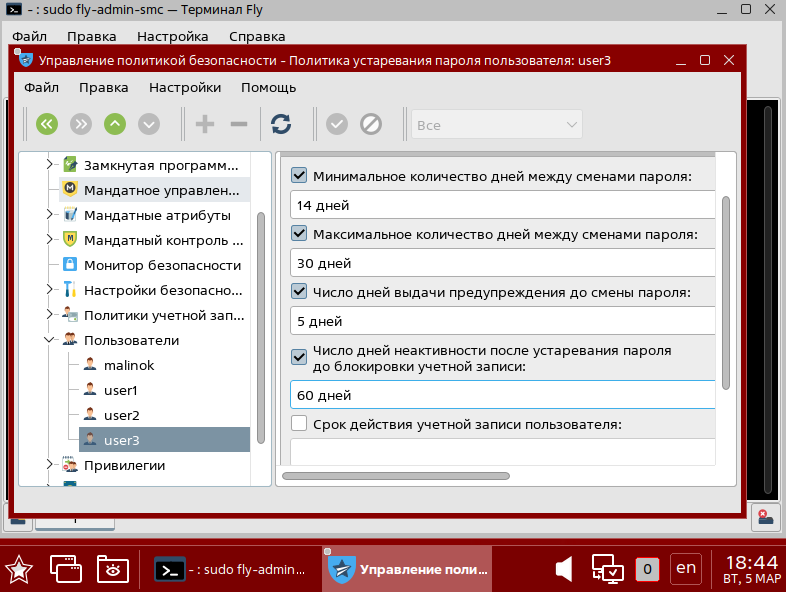
• задайте пароль по своему усмотрению;

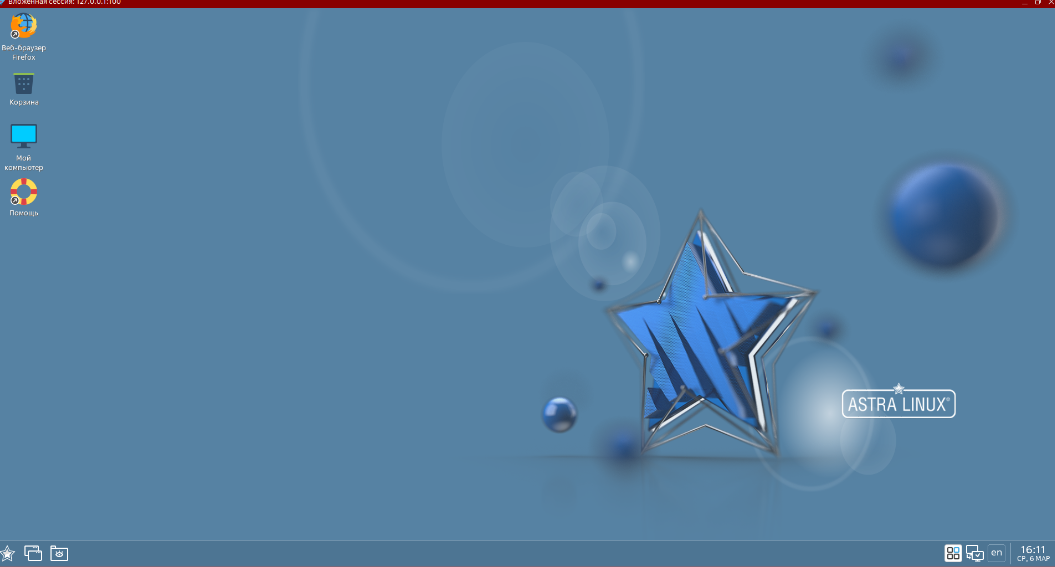
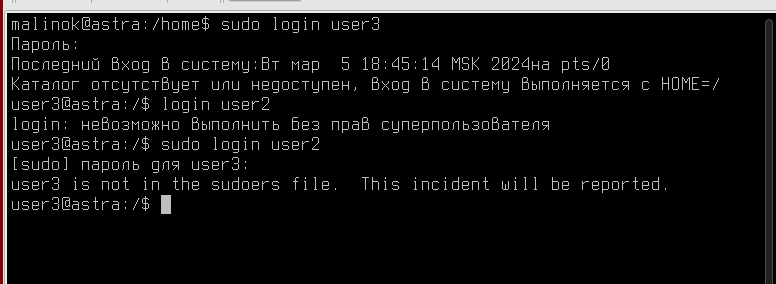
• время действия пароля - 30 дней;

• минимальное время между сменой пароля - 14 дней;

• время неактивности пользователя после окончания действия пароля – 60 дней.

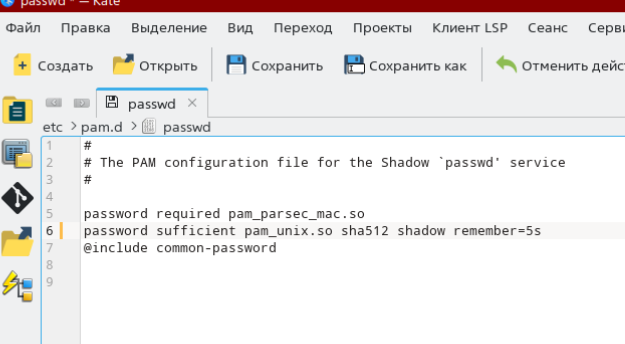




7 Проверьте, что параметры учетной записи user3 соответствуют заданию. Зайдите этим пользователем в графическое окружение и убедитесь, что обои - новые. 

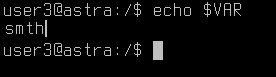
8 Настройте PAM так, чтобы запоминалось 5 последних паролей пользователей, не давая их использовать при очередной смене пароля. Проверьте, что нельзя использовать предыдущие пароли.

9 Когда passwd запускается от имени пользователя root, то можно задавать «плохие» пароли, несмотря на предупреждение команды passwd. Настройте PAM так, чтобы и пользователь root не мог задавать пароли из словаря. Проверьте, что пользователь root должен придерживаться тех же правил формирования пароля, что и обычные пользователи.

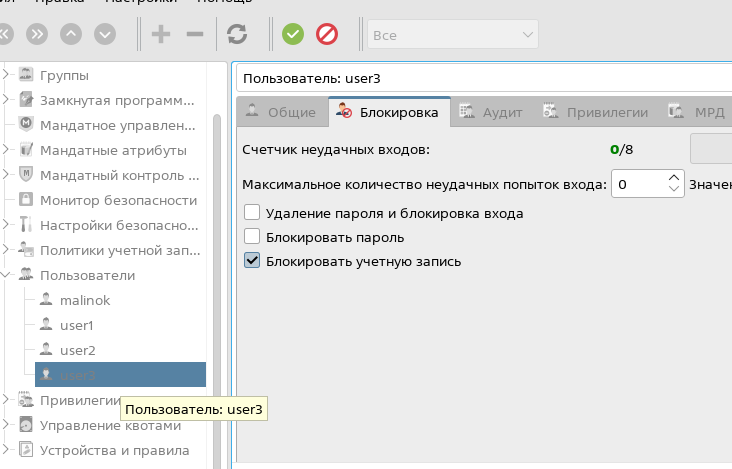


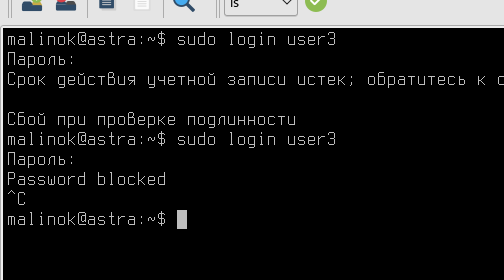
Примечание: изучите man-страницу по модулю pam\_cracklib.

10 Задайте любое значение переменной окружения VAR в файле /etc/environment.



11 Проверьте, что при входе в систему переменная VAR определена.

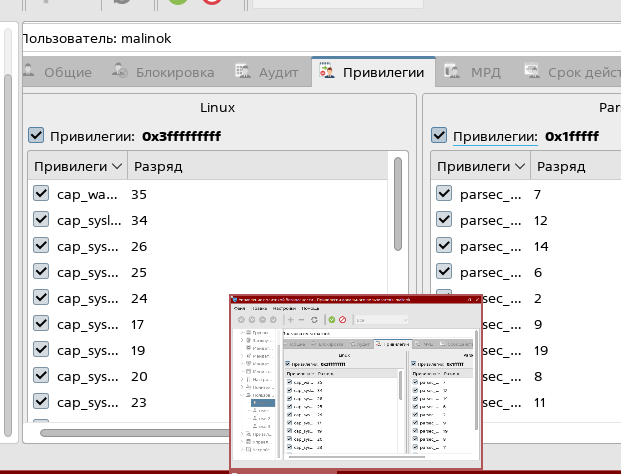
12 Заблокируйте учетную запись user3. 



6. Мандатной управление доступом

Примечание: задания можно выполнять как с помощью утилит командной строки, так и с помощью графических утилит.

1 Зайдите в систему администратором. Получите права root.



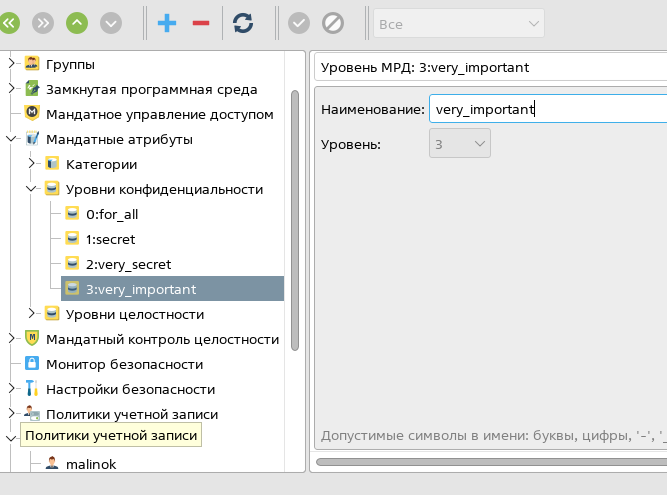
2 Переименуйте уровни конфиденциальности:

• 0 — for\_all

• 1 — secret

• 2 — very\_secret

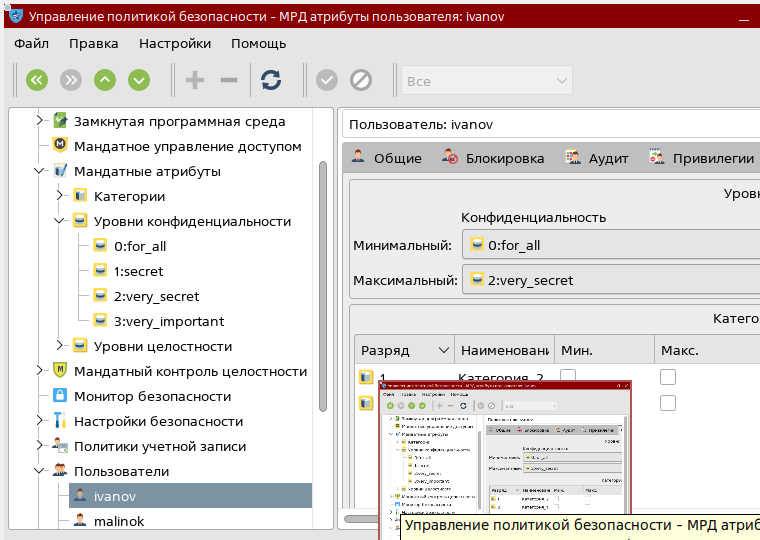
• 3 — very\_important



3 Создайте учетную запись для пользователя ivanov

• минимальный уровень конфиденциальности — for\_all

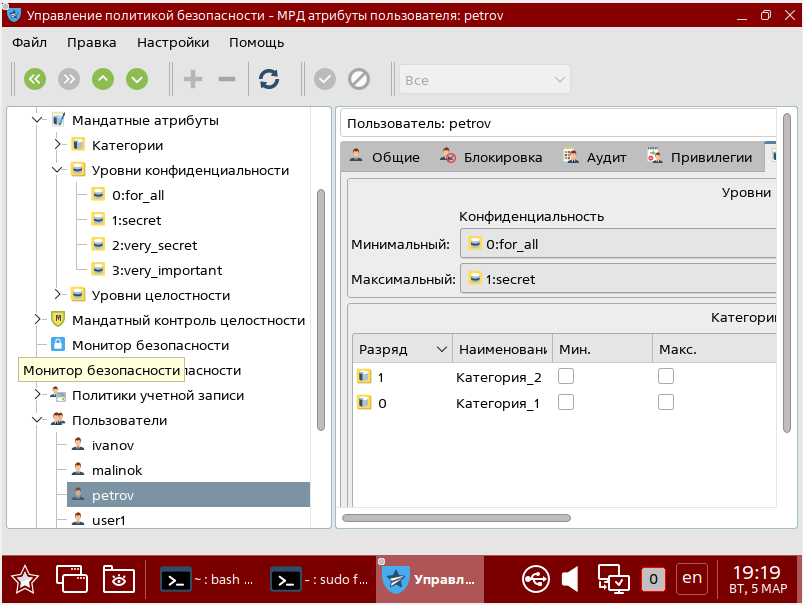
• максимальный уровень конфиденциальности — very\_secret



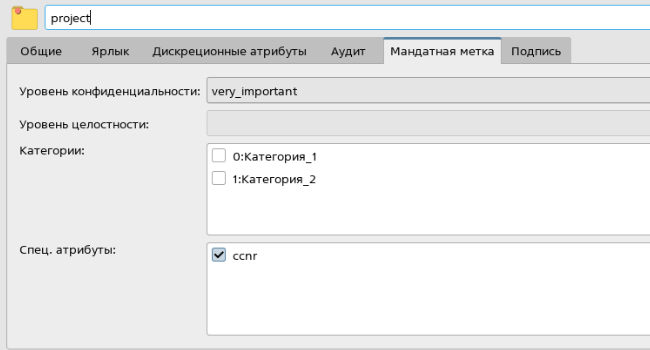
4 Создайте учетную запись для пользователя petrov

• минимальный уровень конфиденциальности — for\_all

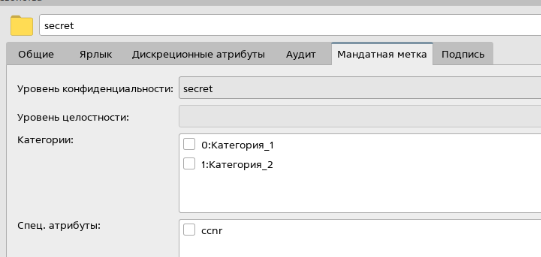
• максимальный уровень конфиденциальности — secret



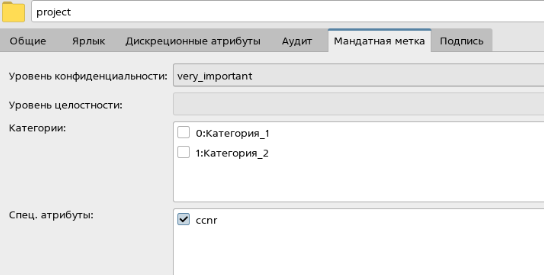
5 Создайте каталог /home/project. Установите на каталог уровень конфиденциальности very\_important и установите дополнительный атрибут ccnr.



6 Создайте каталог /home/project/secret. Установите на каталог уровень конфиденциальности secret.

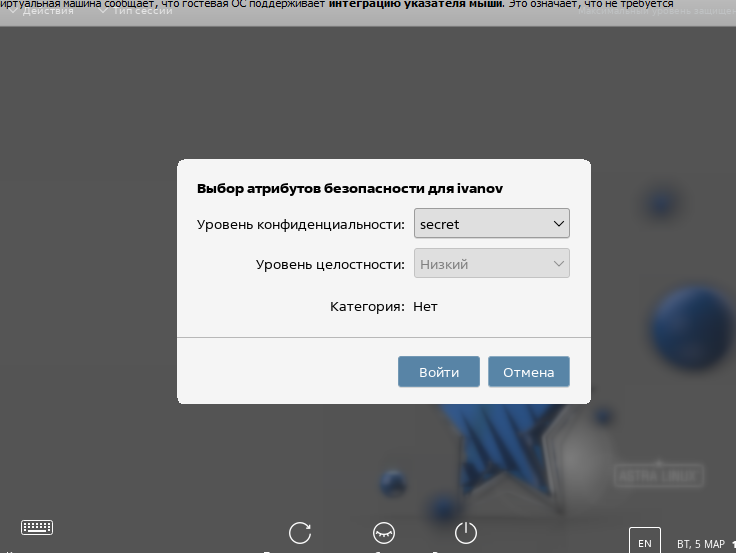


7 Создайте каталог /home/project/very\_secret. Установите на каталог уровень конфиденциальности very\_secret.

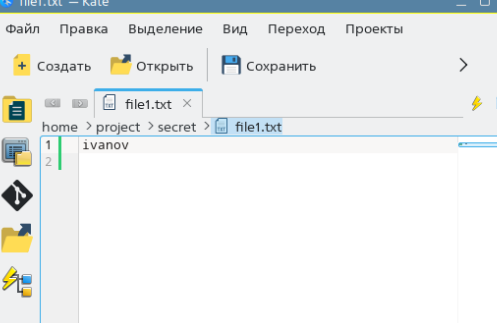


8 Установите файловые списки управления доступом (ACL) и файловые списки управления доступом по умолчанию (default ACL) на каталоги /home/project/secret и /home/project/very\_secret, позволяющие пользователям ivanov и petrov создавать и удалять файлы в этих каталогах и изменять содержимое созданных файлов.

9 Зайдите в систему под учетной записью ivanov с уровнем конфиденциальности secret.



10 Создайте файл file1.txt в каталоге /home/project/secret. В этот файл добавьте строку ivanov. Сохраните файл.

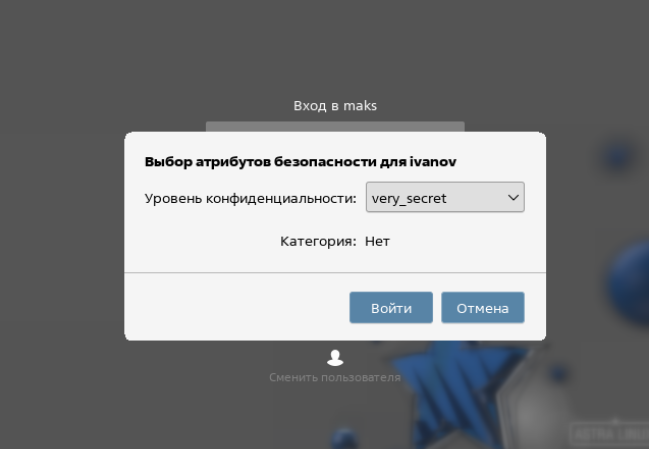


1. Удалось ли создать, изменить и сохранить файл file1.txt?

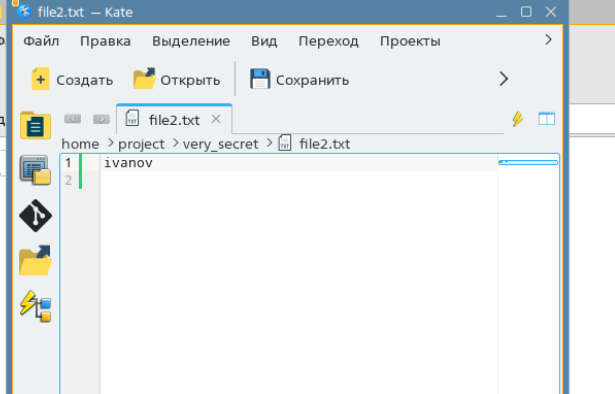
Да

1. Виден ли каталог /home/project/very\_secret?

Нет

13 Зайдите под учетной записью ivanov в систему с уровнем конфиденциальности very\_secret. 

14 Создайте файл file2.txt в каталоге /home/project/very\_secret. В этот файл добавьте строку ivanov. Сохраните файл.

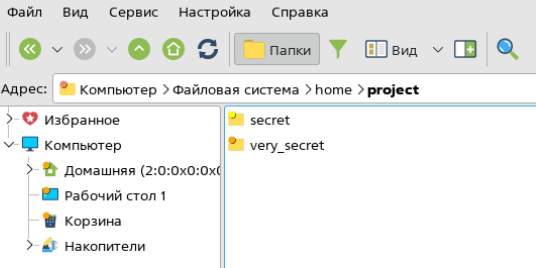


15 Удалось ли создать, и изменить и сохранить файл file2.txt?

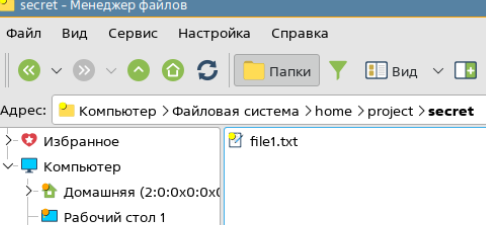
Да

16 Виден ли каталог /home/project/secret?

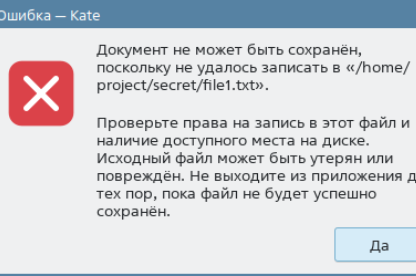
Да



17 Виден ли файл /home/project/secret/file1.txt?



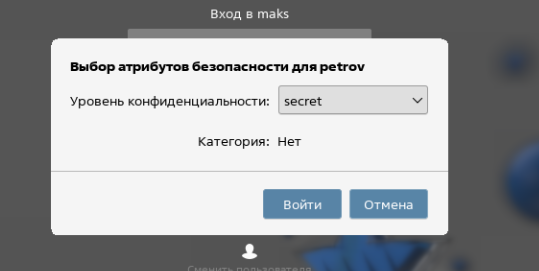
18 Добавьте в файл /home/project/secret/file1.txt строку ivanov2.



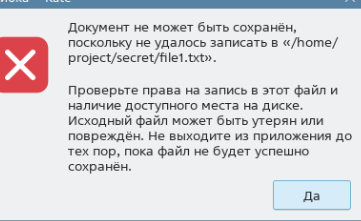
19 Удалось ли изменить содержимое этого файла?

нет

20 Зайдите в систему под учетной записью пользователем petrov с уровнем конфиденциальности secret.



21 Добавьте в файл /home/project/secret/file1.txt строку petrov.

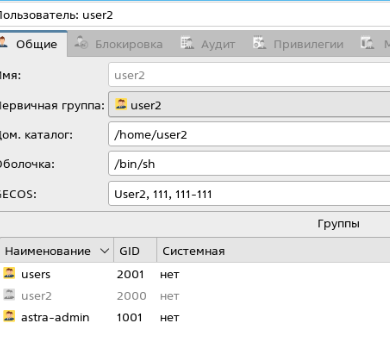


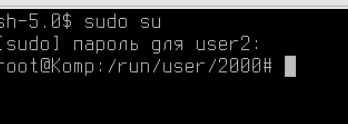
22 Удалось ли изменить содержимое этого файла?

нет

23 Можете ли Вы прочитать содержимое файла /home/project/very\_secret/file2.txt?

нет

24 Сделайте пользователя user2 администратором. Проверьте, что данный пользователь может выполнять команды от имени пользователя root.



Вывод:

В ходе выполнения лабораторной работы были изучены принципы работы ОС Astra Linux, получены навыки администрирования ОС, настройки и контроля мандатного доступа встроенным ПО.

Выполнена настройка терминала, произведена работа с командами терминала по взаимодействию с процессами, создание учётных записей, настройка уровней доступа и блокировка учетных записей.