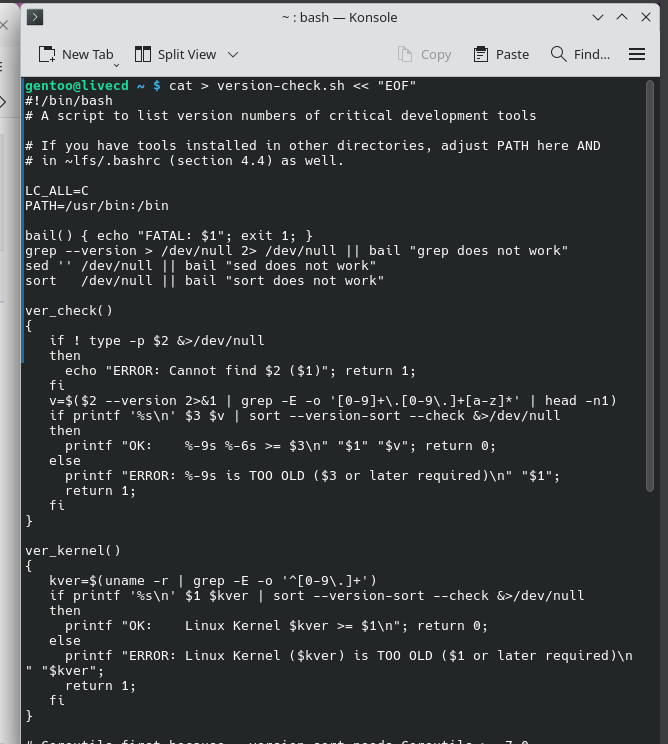
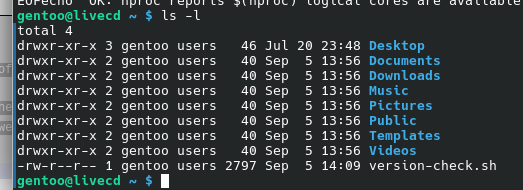
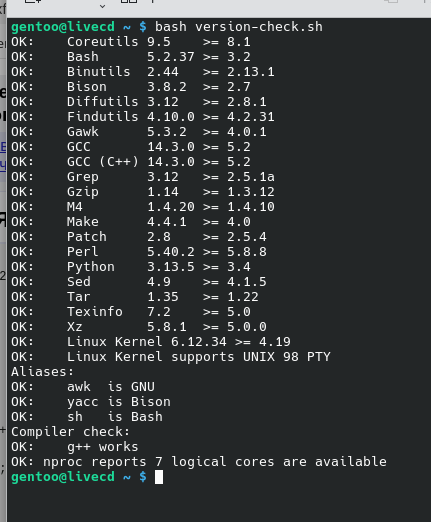
Я установила ОС Gentoo. Для начала нужно проверить, установлены ли у меня пакеты, которые требуются для сборки.



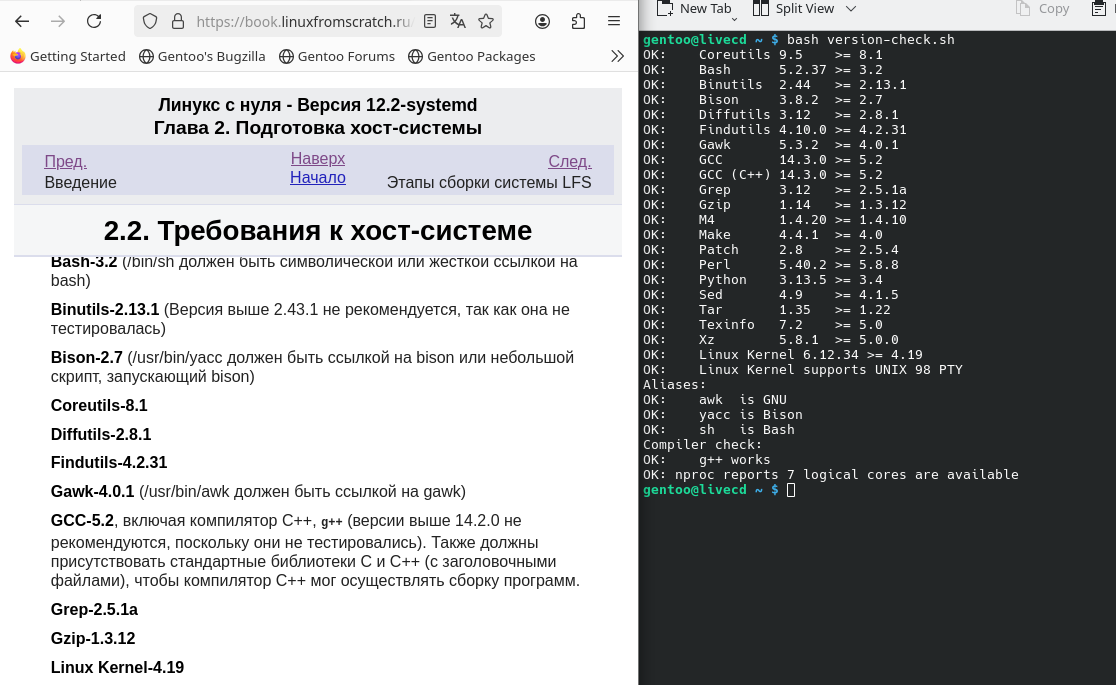
Данная команда выводит список файлов, которые находятся в текущем разделе.



Используем команду bash с названием файла, чтобы проверить наличие всех необходимых пакетов и возможность компилировать программы.

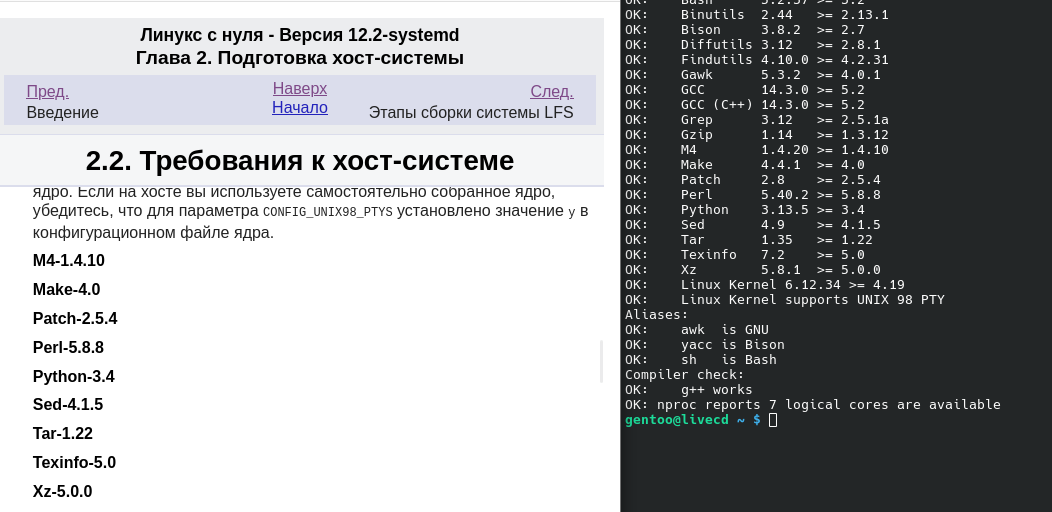


Теперь нужно проверить, совпадают ли версии с теми, что нам нужны. Как видим, версии идентичные.



Обращаю внимание, что версия ядра 4.19 (в книге этому уделен отдельный абзац), а также ядро хоста поддерживает псевдотерминал UNIX 98(PTY).

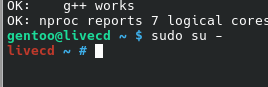
Проверяем версии других пакетов.



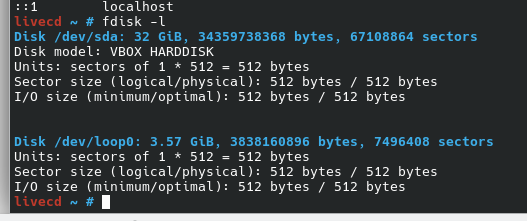
Все совпало.

Следующий шаг – создать переменную, которая будет root’ом для нас. То есть, переменная окружения.

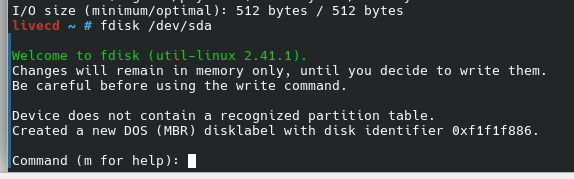
Для начала заходим с рута.



Мы стали root-пользователем того livecd, с которого загрузились. С помощью этого мы можем теперь писать команды самой системе. Далее нужно обратиться к справке с помощью команды “fdisk -l”.

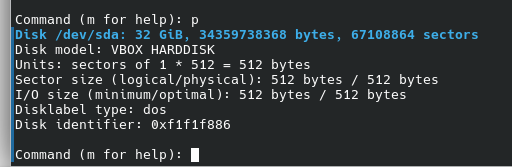


Прописав флаг -l, можно вывести диски списком. Из-за особенностей сборки на виртуальной машине, второй выведенный loop-диск нам не нужен, мы заинтересованы в sda, который обозначен как VBOX HARDDISK.

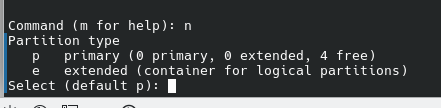


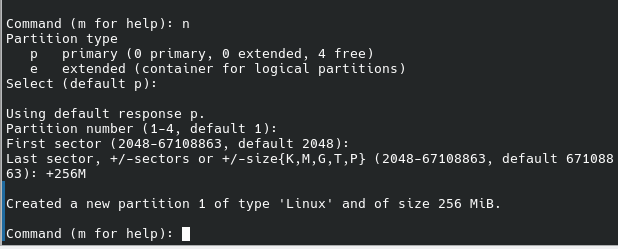
Здесь главное не ошибиться в названии диска, иначе дальше могут быть проблемы из-за невнимательности. Теперь мы находимся в этом диске.

Команда «р» используется для просмотра конфигурации диска.

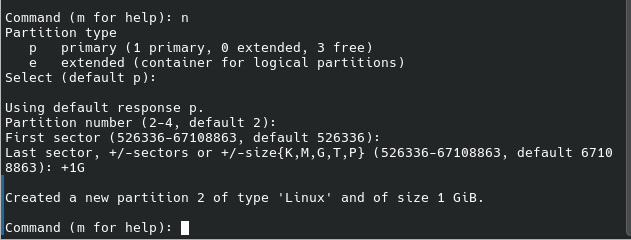


Теперь создаем раздел командой «n». Будем создавать три раздела для экономии времени в будущем.





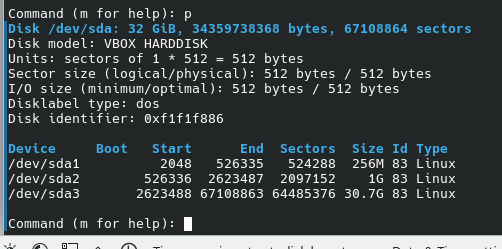
Мы создали первый стандартный (default) раздел на 256 мегабайт под boot.



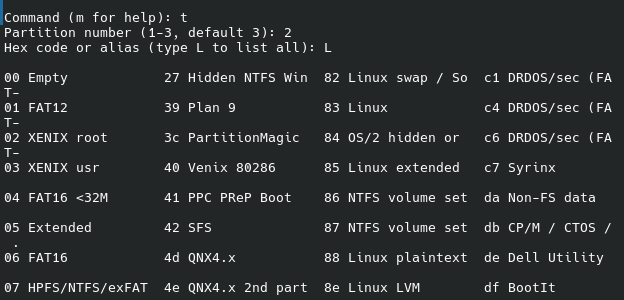
И второй раздел на 1 гигабайт под swap.

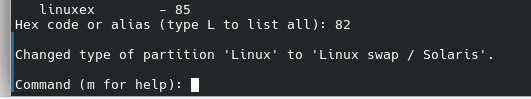
Теперь приступим к разделу root. Повторяем все действия, только в последнем пункте не выделяем определенную память, а берем всю оставшуюся (click enter for default). Вышло 30.7Гб. Это будет раздел под всю систему линукс.

Теперь можно посмотреть все наши разделы.

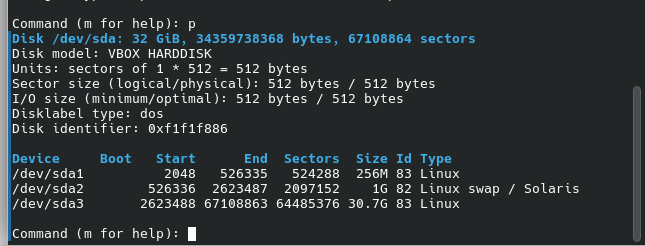


Нужно только заменить тип одного раздела, второго по счету, с типа Linux на тип swap. Команда «t» для смены типа. Команда «L» для того, чтобы посмотреть айди типа раздела, на который мы хотим поменять. 82 для swap Linux.

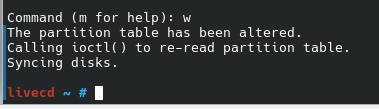




Чтобы убедиться в смене типа раздела, снова вызываем команду «p».

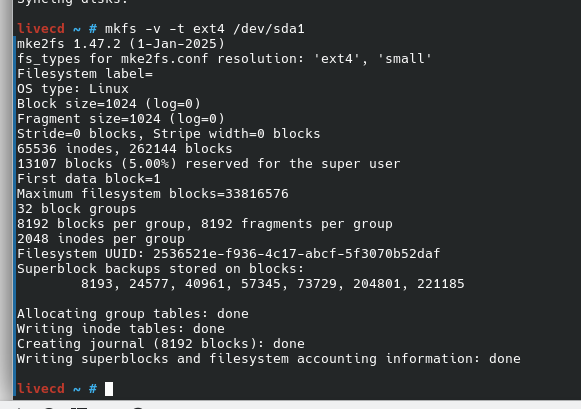


Записываем изменения командой «w».

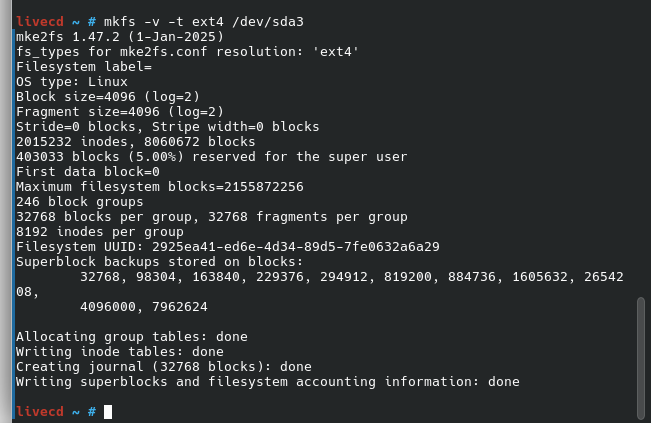


Далее создаем файловую систему.

Сначала используем команду, что прописана в книге.



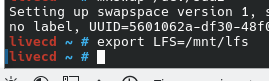
Также и для третьего раздела.



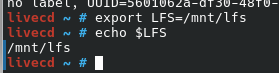
Так как был создан еще и новый раздел подкачки, используем другую команду ко второму разделу.



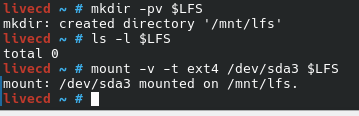
Наконец, создаем переменную LFS.



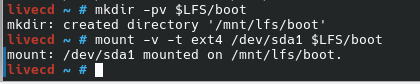
На всякий случай, проверим ее настройку.



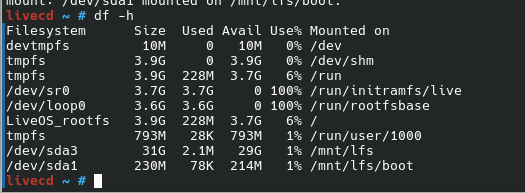
Далее монтируем раздел. Создаем точку монтирования и проверяем её наличие. Это будет корневой раздел будущей системы.



Затем монтируем загрузочный раздел.

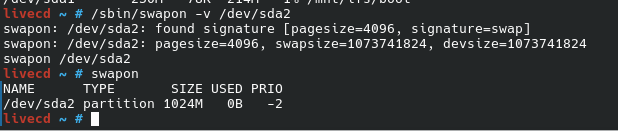


Проверяем все, что сделали.



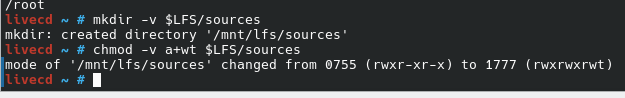
Обращаем внимание только на последние две строчки, остальное не особо важно.

Так как мы сделали раздел подкачки, нужно убедиться, что он также включен.



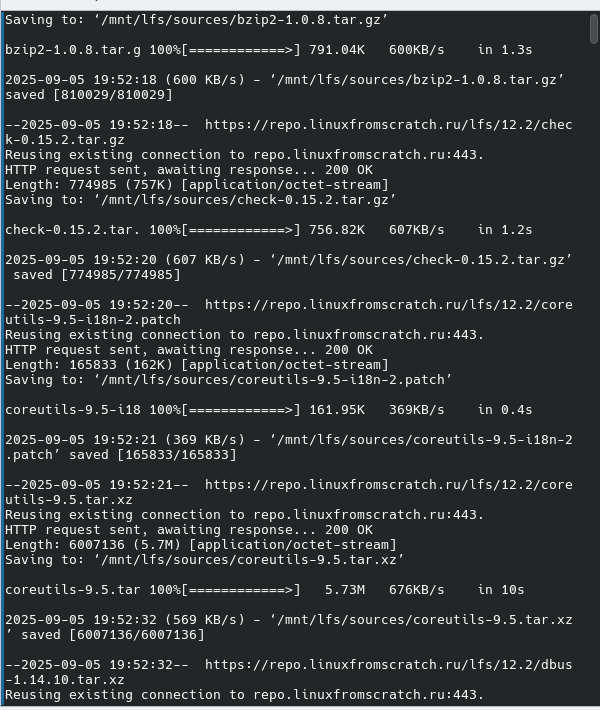
Пакеты и патчи

Для начала нужно создать место для хранения архивов и патчей. Для этого будет служить директорий sources. Сразу же сделаем из него директорий типа «sticky bit». Это означает, что даже если несколько пользователей имеют право на запись в каталог, только владелец файла может удалить файл в подобном каталоге.

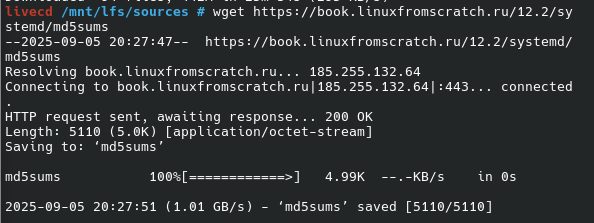


Далее нужно скачать все необходимые пакеты и патчи. Обращаю внимание, что пакеты именно для systemd сборки. Более новой и быстрой инит системы.

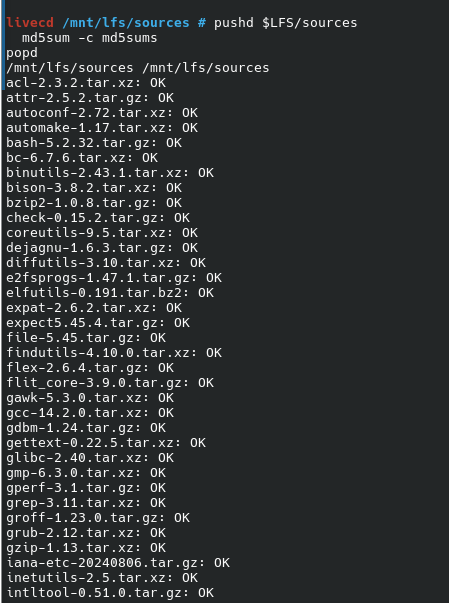




Чтобы проверить, правильно ли установлены все пакеты, скачиваем md5sum ссылкой прямиком из книги.

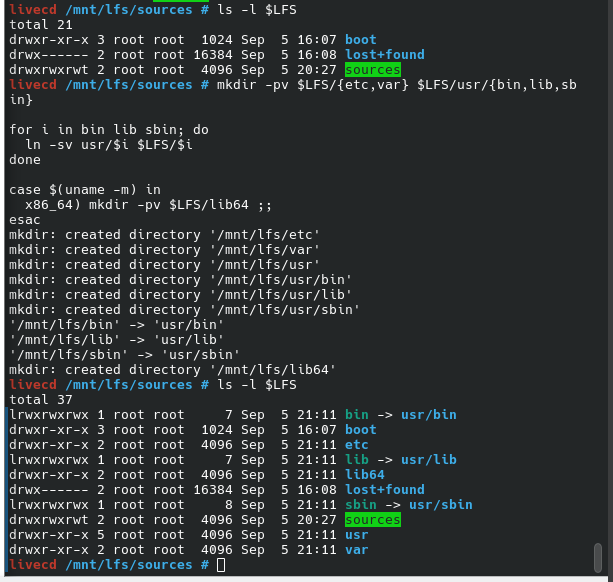


И проверяем командой, указанной в книге.



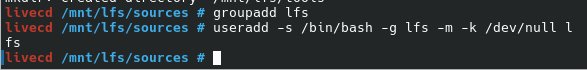
Присваиваем эти файлы имени root.

Далее создадим необходимую иерархию каталогов.

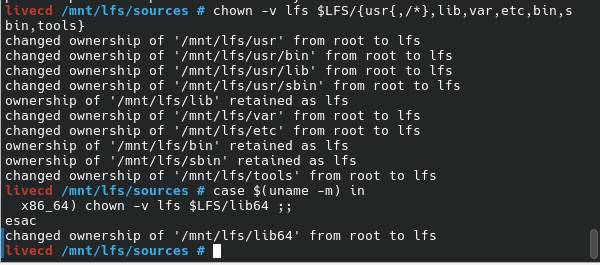


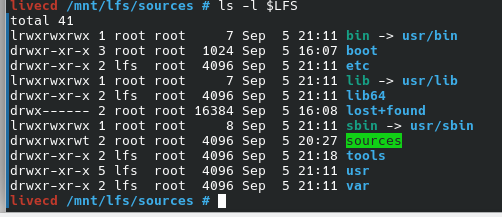
И создадим отдельный каталог под кросс-компилятор.

После этого работать под пользователем root становится опасно, поэтому следует создать непривилегированного пользователя.



Назначаем пользователя lfs владельцем, тем самым предоставляя ему доступ к каталогам папки $LFS.

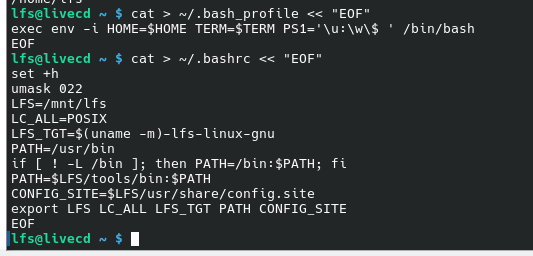




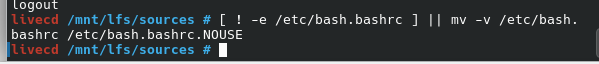
И меняем пользователя на lfs.



Далее создадим два новых файла запуска для оболочки bash.



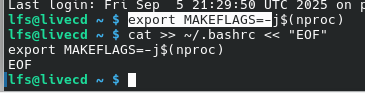
На всякий случай проверим, что пользовательская среда lfs чиста.



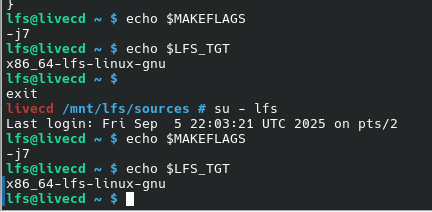
Никакого вывода не последовало, хотя установлен флаг -v, значит, никаких проблем нет.

Далее устанавливаем количество ядер для быстроты. Чтобы не указывать это каждый раз, пропишем это количество прямо в файле .bashrc.

Переменная nproc – количество доступных ядер. Это число будет автоматически подставлено. Чтобы просто узнать количество доступных ядер, нужно ввести просто nproc. В моем случае, выводом станет цифра 7.



Далее проверим, все ли хорошо. Для этого обратимся к переменной MAKEFLAGS, вместо того, чтобы искать определенную строку в файле .bash. Также, для демонстрации правильно использованной команды export, сменим пользователя на root, а затем снова вернемся в lfs.



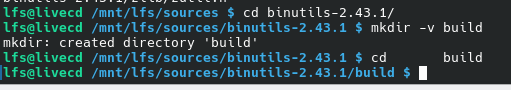
Перед установкой пакетов проверим, что переменная среды LFS настроена правильно.

Binutils-2.43.1 – Pass 1

Достаем пакет Binutils-2.43.1.



Для работы с ним нужно создать отдельный директорий.



Для установки пакета пропишем:



Так как я прописала команду time, то после установки можно наблюдать затраченное время на пакет для отсчета установки последующих пакетов (SBU). Время составило почти 1.2 минуты.

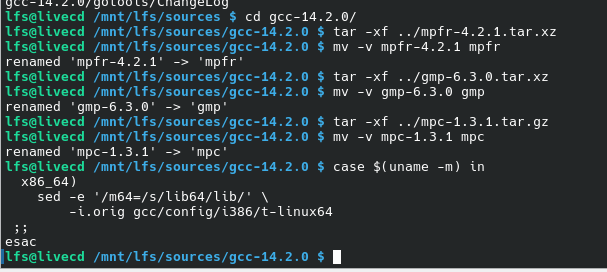


Возвращаемся в директорию sources и удаляем ненужные файлы, оставшиеся после установки binutils.

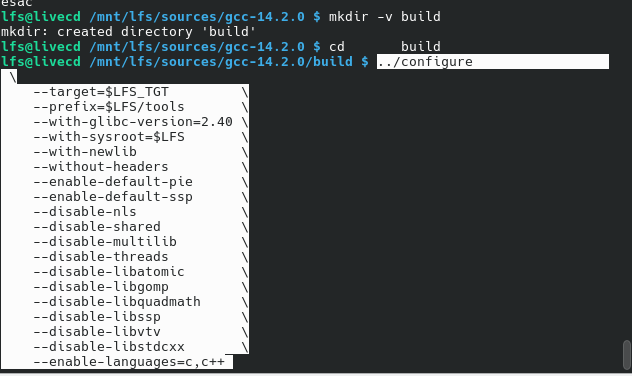


Устанавливаем пакет GCC-14.2.0. Начнем с распаковки архива gcc и других важных пакетов.

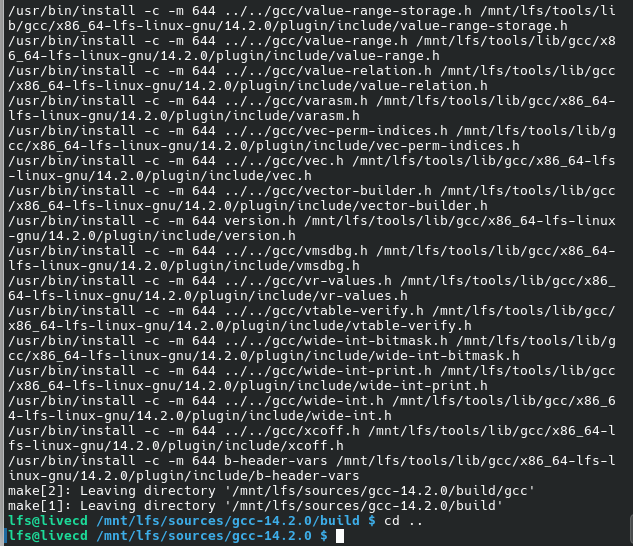




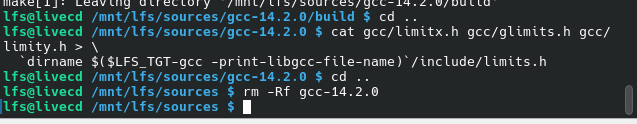
В документации GCC рекомендуется собирать его в отдельном каталоге. Создаем каталог и подготавливаем GCC к компиляции.



Компилируем пакет с помощью команды time make (можно просто make, time используется для подсчета итогового времени), далее устанавливаем пакет с помощью команды make install.



Создаем полную версию внутреннего заголовочного файла, затем удаляем оставшийся архив.

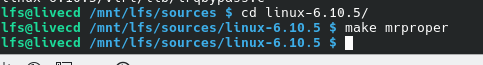


Установка заголовочных файлов.

Повторяем действия для разархивирования пакетов с заголовками и проверяем, нет ли устаревших файлов.







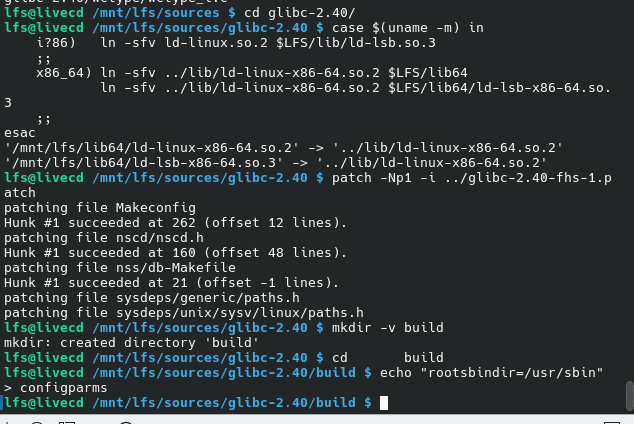
Далее извлекаем заголовочные файлы ядра из исходного кода с помощью команды make headers и команд, представленных на рисунке (вписать номер). После этого удаляем оставшийся архив.



Glibc

Извлекаем из архива Glibc-2.40.

Подготавливаем Glibc к компиляции. Затем компилируем.



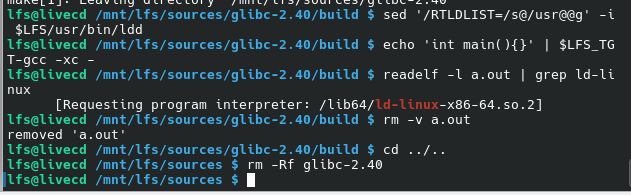
Пакет скомпилирован.



И установлен.

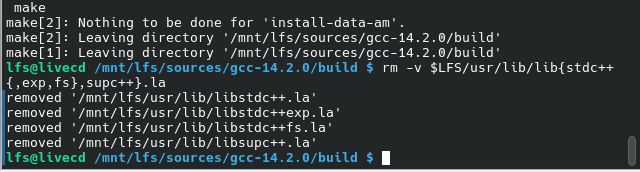


Затем исправляем путь к исполняемому загрузчику и убеждаемся, что основные функции нового кросс-тулчейна работают правильно.



Libstdc++

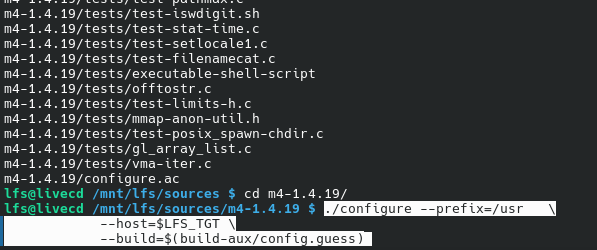
Разархивируем файл gcc-14.2.0, затем создадим каталог сборки, перейдем в него и подготовим файл для компиляции.



Библиотека установлена, потенциально опасные файлы очищены.

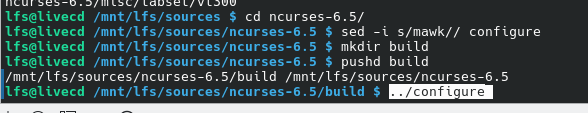
6 глава М4-1.4.19

Разархивируем пакет, подготовим его к компиляции, установим и очистим оставшиеся файлы.

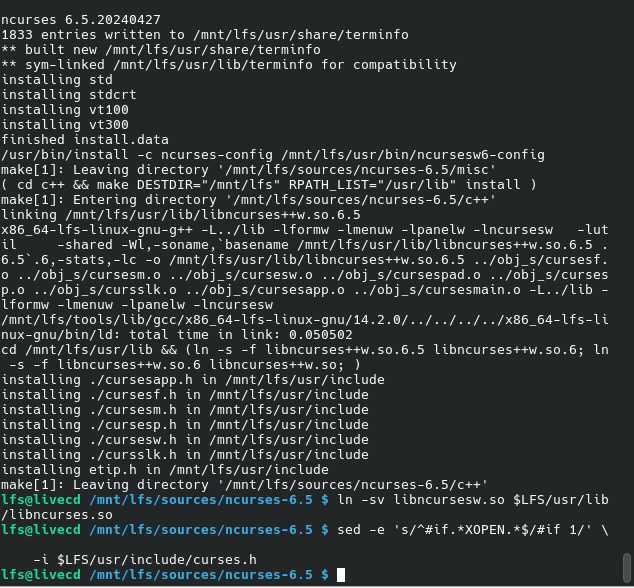


Ncurses.

Снова разархивируем пакет, проверим несколько этапов, чтобы установка прошла верно.

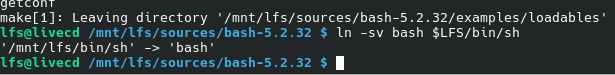


Затем подготовим файл к компиляции и скомпилируем его, как делали это ранее и скачаем, следуя инструкции.



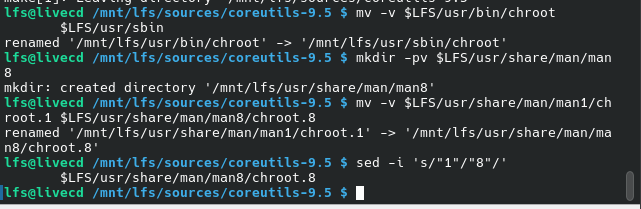
Установка пакета Bash не отличается от установки других пакетов.

В конце создаем символическую ссылку.



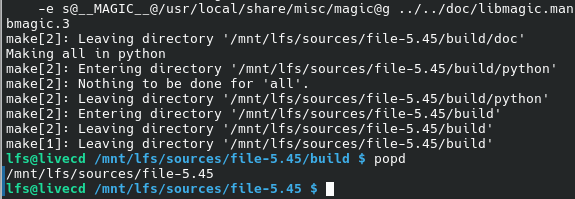
Те же действия при установке повторяем и для пакета Coreutils.

После установки помещаем пару программ в другие директории.



Повторяем действия с Diffutils.

Для установки File нужно предварительно создать временную копию команды file.



В конце удаляем архивный файл libtool, затем подчищаем.



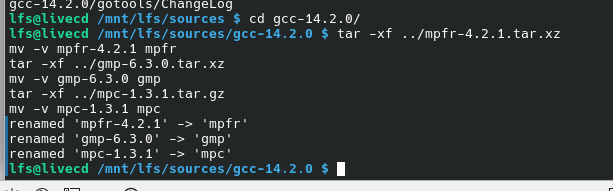
Findutils устанавливается привычным способом.

Перед установкой Gawk нужно убедиться, что некоторые ненужные файлы не будут установлены, далее повторяем предыдущие действия для установки пакета.



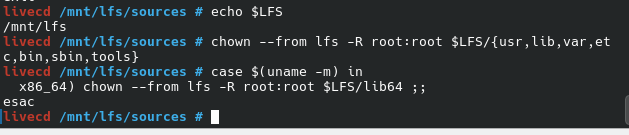
Повторяем все указания для файлов Grep, Gzip, Make, Patch, Sed, Tar, Xz, Binutils.



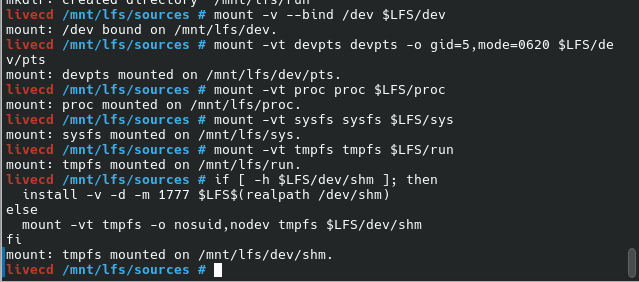


Chroot 7 глава

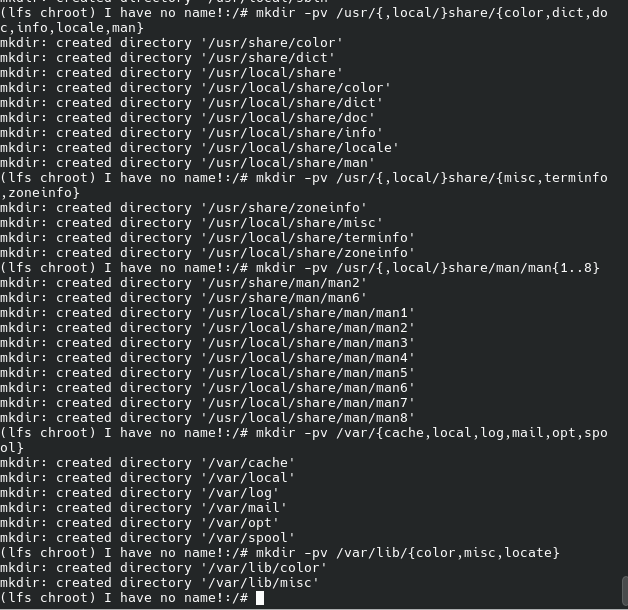
Для начала проверяем окружение командой echo $LFS, далее изменяем владельца каталогов $LFS/ на пользователя root.

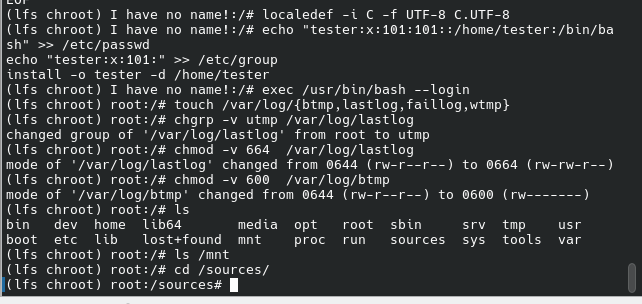


Создаем виртуальную файловую систему kernel и монтируем её.

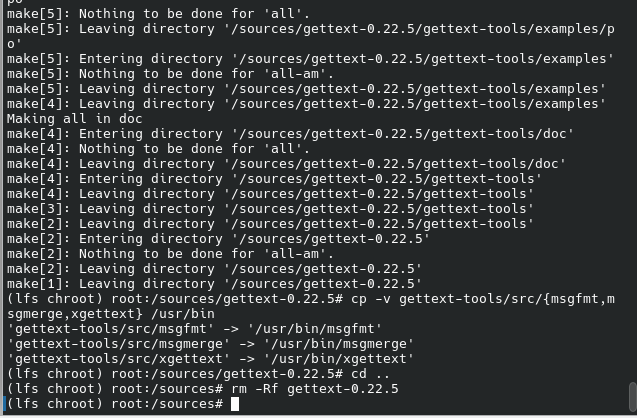


Затем входим в окружение Chroot и настраиваем его. Это понадобится для установки конечной системы.



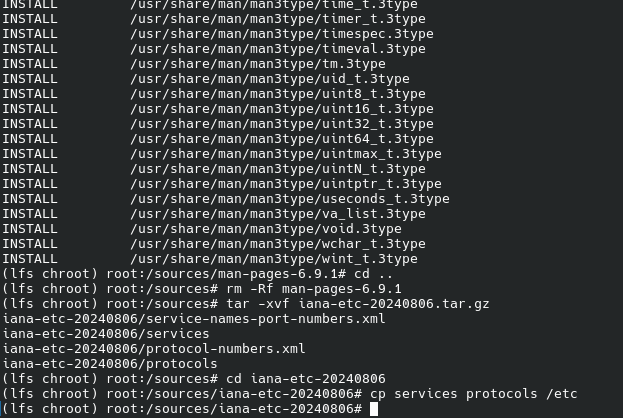


Проверяем, находимся ли мы в директории sources, затем начинаем установку пакета Gettext.

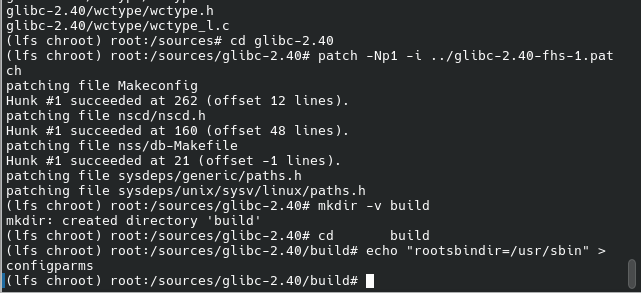


Также устанавливаем пакеты Bison, Perl, Python, Texinfo, Until-linux. Затем освобождаем место указанными в пособии командами.

Далее устанавливаем Man-pages согласно инструкции и копируем файлы iana-etc.

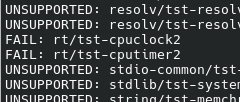


Настраиваем Glibc.

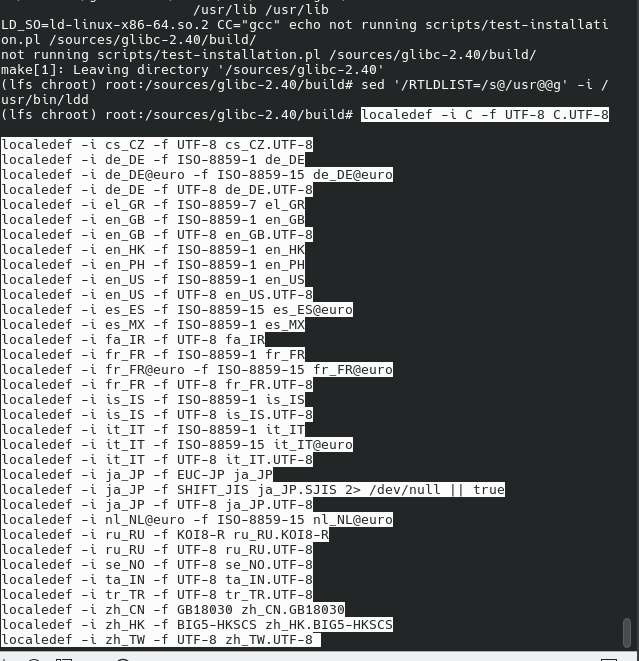


Проверка данного файла завершилась с тремя ошибками, но они незначительны. После исследования оказалось, что это ошибки из-за установки линукса на виртуальной машине. Их можно игнорировать.

rt/tst-cpulock2 и rt/tst/cputimer2



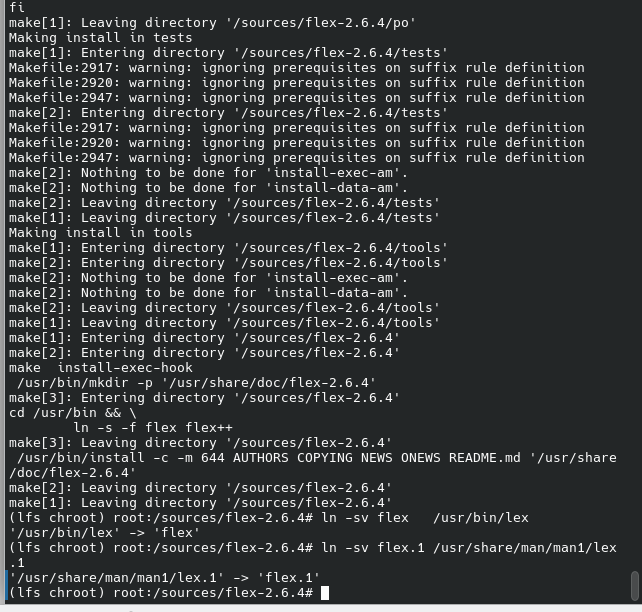
Также установим локали. Можно все не устанавливать, но я сделаю это во исключение возможных ошибок.

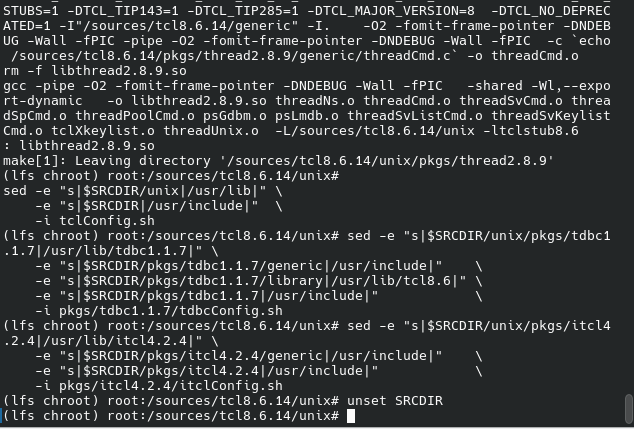


Затем выбираем время часового пояса.

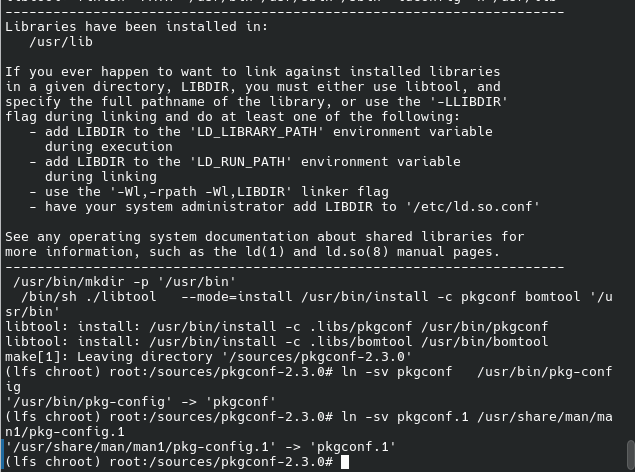


Следуем простой установке Bzip, Lz4, zstd, file, readline, m4, bc, flex, tcl, expect, dejagnu, pkgconf,

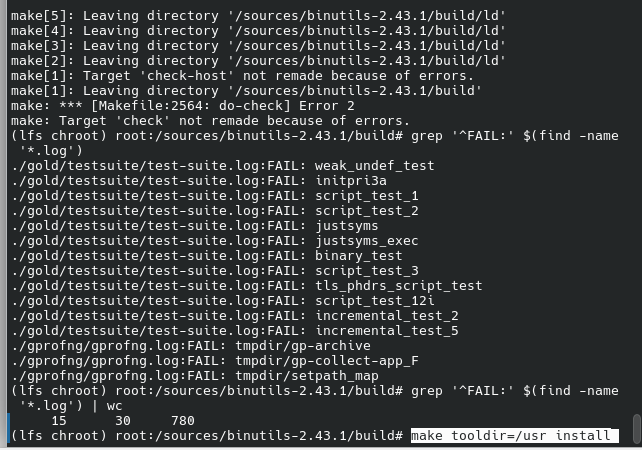




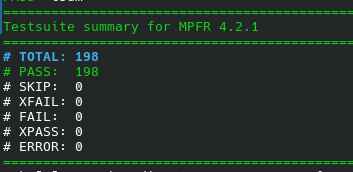
Для пакета expect проверим, правильно ли работает псевдотерминал(PTY).



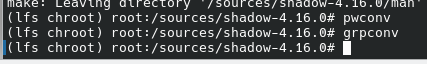
При установке пакета binutils возникло больше ошибок, чем указано в книге. После исследования выяснилось, что это нормально для виртуальной машины.



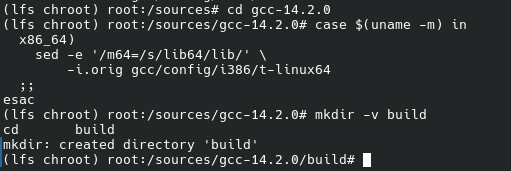
С Mpfr никаких ошибок не возникло.



При установке shadow следует включить поддержку и использование теневых паролей.

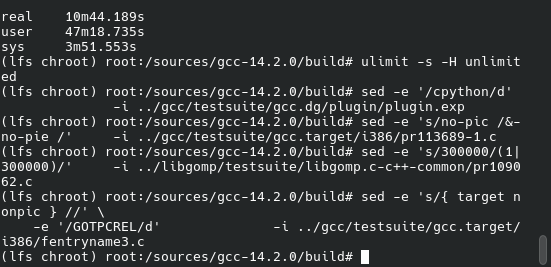


Приступаем к самому длительному для установки пакету, gcc. Меняем имя каталога, затем создаем и переходим в папку build.

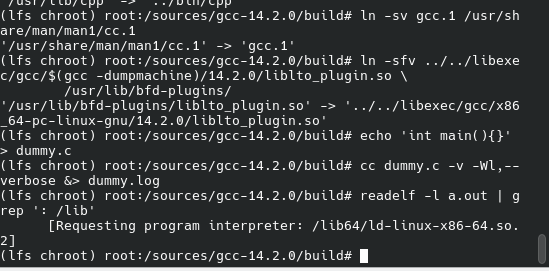


Подготавливаем gcc к компиляции и ждем.

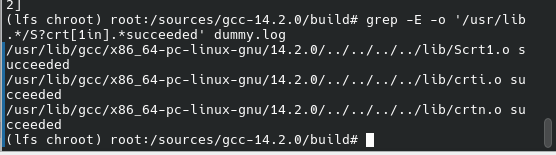
Процесс занял 10 минут. Далее выполняем некоторые настройки.



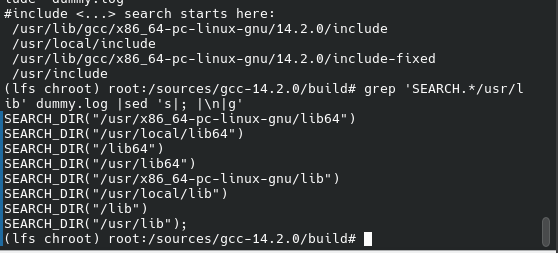
Тест завершился без ошибок. После установки пакета проверяем, что компиляция и компоновка будут работать правильно.



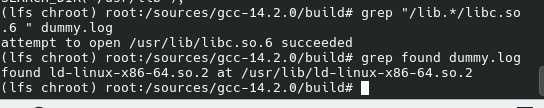
Также проверяем, что мы настроили использование правильных стартовых файлов.



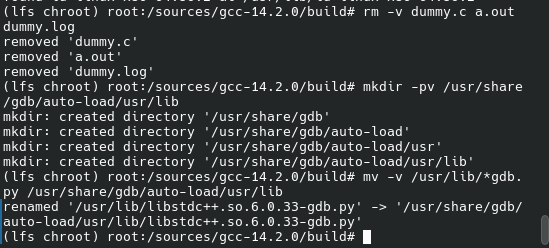
Выполняем проверку на то, что компилятор ищет правильные заголовочные файлы и что компоновщик использует правильные пути поиска.



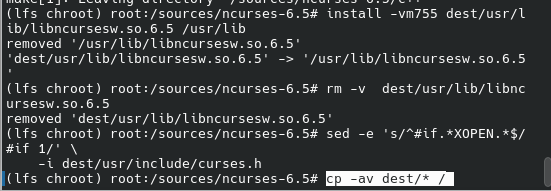
Далее проверка на правильную libc и динамический компоновщик.



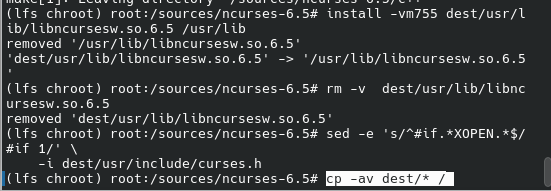
Удаляем тестовые файлы и делаем еще несколько действий для корректной работы GCC.



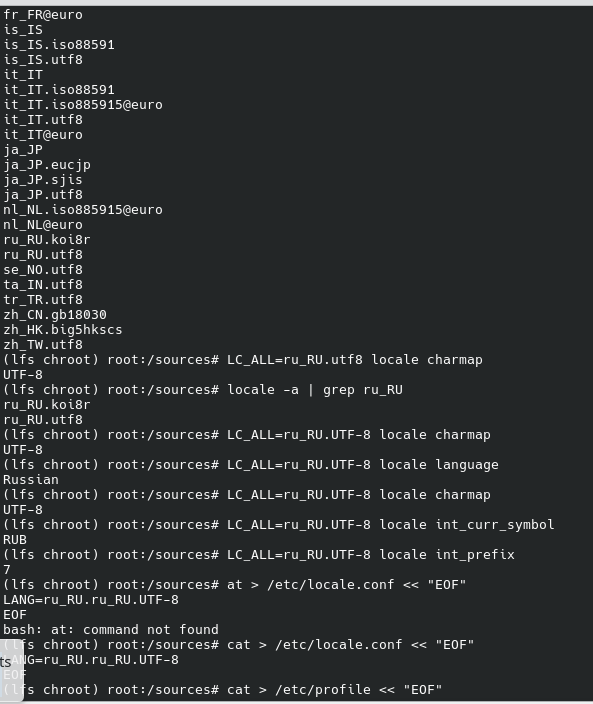
После установки пакета Ncurses нужно заменить файл библиотеки, чтобы не нарушить систему.

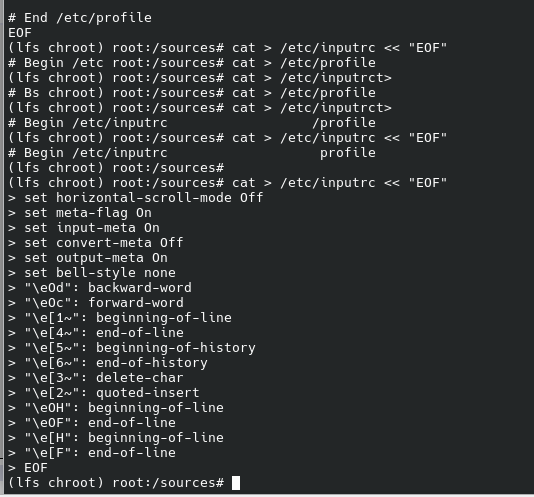


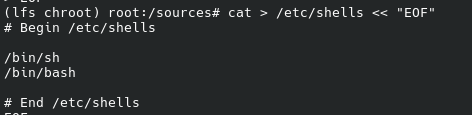
Затем связать приложения с библиотеками расширенного набора символов.

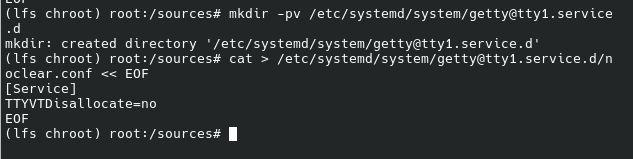


Глава 9

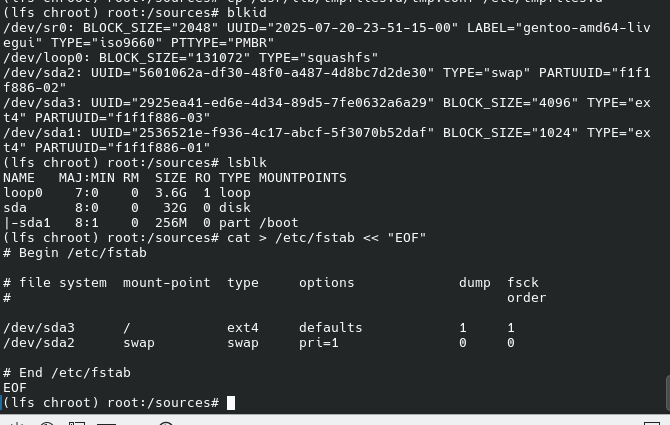


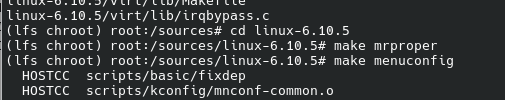


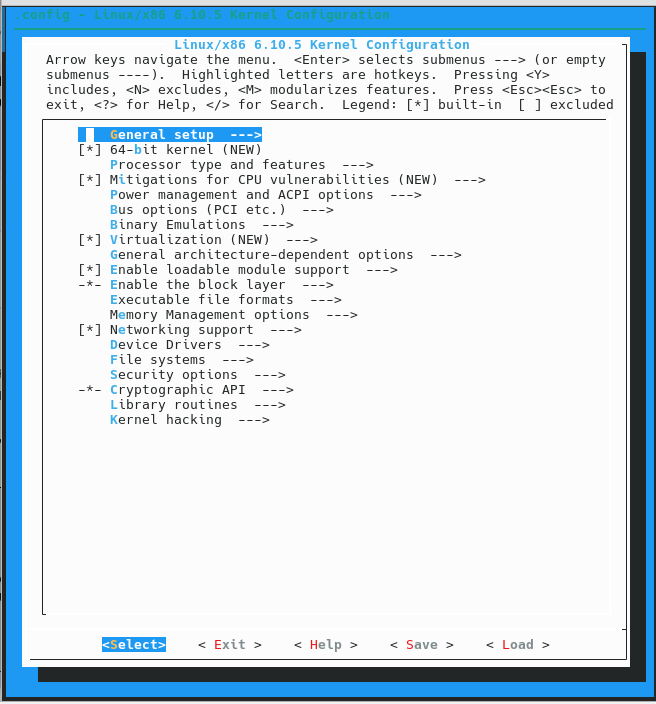




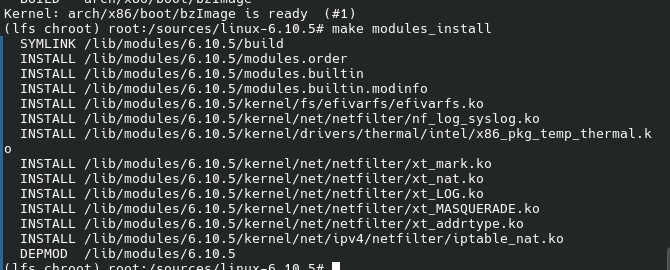




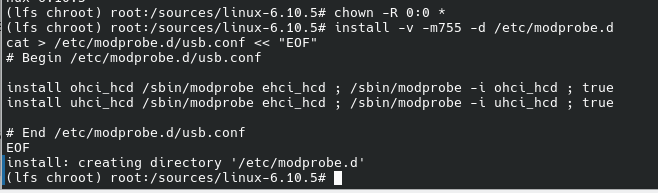


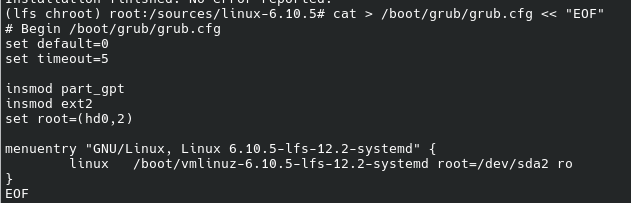


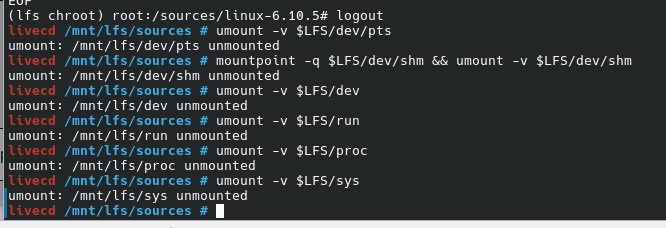


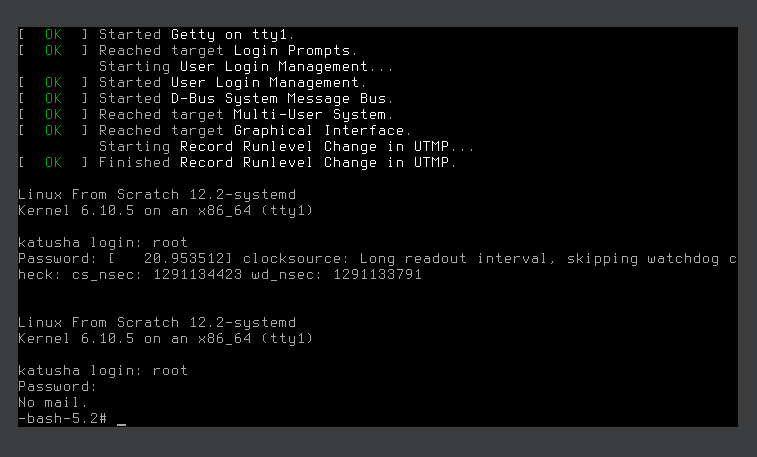












Характеристики

