2. Подготовка к сборке

2. Подготовка хост-системы

Содержание

2.1. Введение

В этой главе проверяются и при необходимости устанавливаются основные инструменты, необходимые для построения LFS. Затем подготавливается раздел, в котором будет размещаться система LFS. Мы создадим сам раздел, создадим на нем файловую систему и смонтируем его.

2.2. Требования к хост-системе

2.2.1. Аппаратное обеспечение

Редакторы LFS рекомендуют, чтобы процессор имел не менее четырех ядер и не менее 8 ГБ памяти. Старые системы, не отвечающие этим требованиям, будут по-прежнему работать, но время сборки пакетов будет значительно больше, чем указано в документации.

2.2.2. Программное обеспечение

Ваша хост-система должна иметь следующее программное обеспечение с указанными минимальными версиями. Это не должно быть проблемой для большинства современных дистрибутивов Linux. Также обратите внимание на то, что многие дистрибутивы помещают заголовочные файлы в отдельные пакеты, как правило в формате <package-name>-devel или <package-name>-dev. Обязательно установите эти пакеты, если ваш дистрибутив их предоставляет.

Более ранние версии перечисленных ниже пакетов могут работать, но это не проверялось.

- Bash-3.2 (/bin/sh должен быть символической или жесткой ссылкой на bash)
- Binutils-2.13.1 (Версия выше 2.42 не рекомендуется, так как она не тестировалась)
- Bison-2.7 (/usr/bin/yacc должен быть ссылкой на bison или небольшой скрипт,

запускающий bison)
- Coreutils-8.1
- Diffutils-2.8.1
- Findutils-4.2.31
- Gawk-4.0.1 (/usr/bin/awk должен быть ссылкой на gawk)
- GCC-5.2, включая компилятор C++, g++ (версии выше 13.2.0 не рекомендуются, поскольку они не тестировались). Также должны присутствовать стандартные библиотеки C и C++ (с заголовочными файлами), чтобы компилятор C++ мог осуществлять сборку программ.
- Grep-2.5.1a
- Gzip-1.3.12
- Linux Kernel-4.19

Причиной, по которой указаны минимальные требования к версии ядра, является то, что мы указываем эту версию при сборке glibc в Глава 5 и Глава 8. Так как более старые ядра не поддерживаются, скомпилированный пакет glibc немного меньше и быстрее. По состоянию на февраль 2024 г. 4.19 является самой старой версией ядра, поддерживаемой разработчиками ядра. Некоторые версии ядра, более старые, чем 4.19, могут по-прежнему поддерживаться сторонними командами, но они не считаются официальными выпусками ядра; подробности читайте на странице https://kernel.org/category/releases.html

Если версия ядра хоста более ранняя, чем 4.19, вам необходимо обновить ядро на более современную версию. Есть два способа сделать это. Во-первых, посмотрите, предоставляет ли ваш дистрибутив Linux пакет ядра 4.19 или более позднюю версию. Если это так, установите его. Если ваш дистрибутив не предлагает приемлемый пакет ядра или вы предпочитаете не устанавливать его, вы можете скомпилировать ядро самостоятельно. Инструкции по компиляции ядра и настройке загрузчика (при условии, что хост использует GRUB) находятся в Глава 10.

Для сборки LFS необходимо, чтобы ядро хоста поддерживало псевдотерминал UNIX 98 (PTY). Обычно он включен на всех настольных или серверных дистрибутивах, поставляющих Linux 4.19 или более новое ядро. Если на хосте вы используете самостоятельно собранное ядро, убедитесь, что для параметра CONFIG_UNIX98_PTYS установлено значение y в конфигурационном файле ядра.

- M4-1.4.10
- Make-4.0
- Patch-2.5.4
- Perl-5.8.8
- Python-3.4
- Sed-4.1.5
- Tar-1.22
- Texinfo-5.0
- Xz-5.0.0

Важно Обратите внимание, что упомянутые выше символические ссылки необходимы для создания системы LFS с использованием инструкций, содержащихся в этой книге. Симлинки, указывающие на другое программное обеспечение (например, dash, mawk и т. д.), могут работать, но не тестируются и

не поддерживаются командой разработчиков LFS, и могут потребовать либо отклонения от инструкций, либо дополнительных исправлений для некоторых пакетов.

Чтобы узнать, есть ли в вашей хост-системе все необходимые пакеты и возможность компилировать программы, выполните следующий скрипт:

```
cat > version-check.sh << "EOF"
#!/bin/bash
# A script to list version numbers of critical development tools

# If you have tools installed in other directories, adjust PATH here AND
# in ~lfs/.bashrc (section 4.4) as well.

LC_ALL=C
PATH=/usr/bin:/bin

bail() { echo "FATAL: $1"; exit 1; }
grep --version > /dev/null 2> /dev/null || bail "grep does not work"
sed '' /dev/null || bail "sed does not work"
sort   /dev/null || bail "sort does not work"

ver_check()
{
   if ! type -p $2 &>/dev/null
   then
     echo "ERROR: Cannot find $2 ($1)"; return 1;
   fi
   v=$($2 --version 2>&1 | grep -E -o '[0-9]+\.[0-9\.]+[a-z]*' | head -n1)
   if printf '%s\n' $3 $v | sort --version-sort --check &>/dev/null
   then
     printf "OK:    %-9s %-6s >= $3\n" "$1" "$v"; return 0;
   else
     printf "ERROR: %-9s is TOO OLD ($3 or later required)\n" "$1";
     return 1;
   fi
}

ver_kernel()
{
   kver=$(uname -r | grep -E -o '^[0-9\.]+')
   if printf '%s\n' $1 $kver | sort --version-sort --check &>/dev/null
   then
     printf "OK:    Linux Kernel $kver >= $1\n"; return 0;
   else
     printf "ERROR: Linux Kernel ($kver) is TOO OLD ($1 or later
required)\n" "$kver";
     return 1;
   fi
}
```

```
# Coreutils first because --version-sort needs Coreutils >= 7.0
ver_check Coreutils      sort     8.1 || bail "Coreutils too old, stop"
ver_check Bash           bash     3.2
ver_check Binutils       ld       2.13.1
ver_check Bison          bison    2.7
ver_check Diffutils      diff     2.8.1
ver_check Findutils      find     4.2.31
ver_check Gawk           gawk     4.0.1
ver_check GCC            gcc      5.2
ver_check "GCC (C++)"    g++      5.2
ver_check Grep           grep     2.5.1a
ver_check Gzip           gzip     1.3.12
ver_check M4             m4       1.4.10
ver_check Make           make     4.0
ver_check Patch          patch    2.5.4
ver_check Perl           perl     5.8.8
ver_check Python         python3  3.4
ver_check Sed            sed      4.1.5
ver_check Tar            tar      1.22
ver_check Texinfo        texi2any 5.0
ver_check Xz             xz       5.0.0
ver_kernel 4.19

if mount | grep -q 'devpts on /dev/pts' && [ -e /dev/ptmx ]
then echo "OK:    Linux Kernel supports UNIX 98 PTY";
else echo "ERROR: Linux Kernel does NOT support UNIX 98 PTY"; fi

alias_check() {
   if $1 --version 2>&1 | grep -qi $2
   then printf "OK:    %-4s is $2\n" "$1";
   else printf "ERROR: %-4s is NOT $2\n" "$1"; fi
}
echo "Aliases:"
alias_check awk GNU
alias_check yacc Bison
alias_check sh Bash

echo "Compiler check:"
if printf "int main(){}" | g++ -x c++ -
then echo "OK:    g++ works";
else echo "ERROR: g++ does NOT work"; fi
rm -f a.out

if [ "$(nproc)" = "" ]; then
   echo "ERROR: nproc is not available or it produces empty output"
else
   echo "OK: nproc reports $(nproc) logical cores are available"
fi
EOF
```

```
bash version-check.sh
```

2.3. Этапы сборки системы LFS

LFS разработан для сборки за один сеанс. То есть инструкция предполагает, что система не будет выключаться в процессе. Это не означает, что система должна быть собрана за один присест. Для возобновления сборки в точке предыдущей остановки (после перезагрузки/выключения), необходимо выполнить некоторые процедуры повторно.

2.3.1. Главы 1–4

Эти главы выполняются на хост-системе. После перезагрузки обратите внимание на следующее:

При выполнении операций, от имени пользователя root после Раздела 2.4, ДЛЯ ПОЛЬЗОВАТЕЛЯ root должна быть установлена переменная окружения LFS.

2.3.2. Главы 5–6

Раздел /mnt/lfs должен быть смонтирован.

Эти две главы должны быть выполнены из-под пользователя lfs. Перед выполнением любой задачи в этих главах необходимо выполнить команду su - lfs. В противном случае вы рискуете установить пакеты на хост и сделать его непригодным для использования.

Выполнение процедур из Общие инструкции по компиляции имеет решающее значение. Если есть какие-либо сомнения по поводу установки пакета, убедитесь, что все ранее распакованные tar-архивы удалены, затем повторно извлеките файлы и выполните все инструкции, приведенные в этом разделе.

2.3.3. Главы 7–10

Раздел /mnt/lfs должен быть смонтирован.

Некоторые операции, такие как «Смена владельца» или «Вход в среду Chroot», должны быть выполнены от имени пользователя root с переменной окружения $LFS, установленной для пользователя root.

При входе в chroot переменная среды LFS должна быть установлена для пользователя root. Переменная LFS не используется после входа в среду chroot.

Виртуальные файловые системы должны быть смонтированы. Это можно сделать до или после входа в chroot, переключившись на виртуальный терминал хоста и от имени пользователя root выполнив команды, описанные в Раздел 7.3.1, «Монтирование и заполнение /dev» и Раздел 7.3.2, «Монтирование виртуальных файловых систем ядра».

2.4. Создание нового раздела

Как и большинство других операционных систем, LFS обычно устанавливается на выделенный раздел. Рекомендуемый подход к построению системы LFS состоит в том, чтобы использовать доступный пустой раздел или, если у вас достаточно неразмеченного пространства, использовать его

Минимальная система требует раздел размером около 10 гигабайт (ГБ). Этого достаточно для хранения всех архивов с исходным кодом и компиляции пакетов. Однако, если система LFS предназначена для использования в качестве основной системы Linux, вероятно, будет установлено дополнительное программное обеспечение, для которого потребуется дополнительное пространство. Раздел размером 30 ГБ является разумным размером для расширения. Сама система LFS не займет столько места. Большая часть этого требования заключается в предоставлении достаточного временного хранилища, а также в добавлении дополнительных возможностей после сборки LFS. Кроме того, для компиляции пакетов может потребоваться много места на диске, которое будет освобождено после установки пакета.

Поскольку для компиляции не всегда достаточно оперативной памяти (ОЗУ), рекомендуется использовать небольшой раздел диска в качестве раздела подкачки. Он используется ядром для хранения редко используемых данных и оставляет больше памяти для активных процессов. Раздел подкачки для системы LFS может совпадать с разделом, используемым хост-системой, и в этом случае нет необходимости создавать еще один.

Запустите программу создания разделов диска, такую как cfdisk или fdisk, с параметром командной строки, указав имя жесткого диска, на котором будет создан новый раздел, например, /dev/sda для основного диска. Создайте раздел Linux и раздел подкачки, если это необходимо. Пожалуйста, обратитесь к справке cfdisk(8) или fdisk(8), если вы еще не знаете, как пользоваться этими программами.

> Примечание Для опытных пользователей возможны и другие схемы разбиения. Система LFS может располагаться на программном RAID-массиве или логическом томе LVM. Однако для некоторых опций требуется initramfs, что является сложной темой. Эти методы разбиения не рекомендуются начинающим пользователям LFS.

Запомните обозначение созданного раздела (например, sda5). В этой книге он будет называться разделом LFS. Также запомните обозначение раздела подкачки. Эти имена понадобятся позже для файла /etc/fstab.

2.4.1. Другие вопросы по созданию разделов

Рекомендации по созданию разделов системы часто публикуются в списках рассылки LFS. Это очень субъективная тема. По умолчанию для большинства дистрибутивов используется весь диск, за исключением небольшого раздела подкачки. Это не оптимально для LFS по нескольким причинам. Это снижает гибкость, затрудняет совместное использование данных между несколькими дистрибутивами или сборками LFS, делает резервное копирование более

трудоемким и может тратить дисковое пространство из-за неэффективно распределенной файловой системы.

2.4.1.1. Корневой раздел

Корневой раздел LFS (не путать с каталогом /root) размером в 20 гигабайт является хорошим компромиссом для большинства систем. Он обеспечивает достаточно места для построения LFS и большей части BLFS, но достаточно мал, чтобы можно было легко создать несколько разделов для экспериментов.

2.4.1.2. Раздел подкачки

Большинство дистрибутивов автоматически создают раздел подкачки. Обычно рекомендуемый размер раздела подкачки примерно в два раза превышает объем физической памяти, однако это требуется редко. Если дисковое пространство ограничено, установите размер раздела подкачки в два гигабайта и контролируйте его объемом.

Если вы хотите использовать режим гибернации (suspend-to-disk) Linux, которая записывает содержимое ОЗУ в раздел подкачки перед выключением машины. Установите размер раздела подкачки не меньше объема установленной оперативной памяти.

Использование файла подкачки - это не очень хорошо. Для механических жестких дисков вы можете определить, что система использует раздел подкачки, просто слыша активность диска и наблюдая, как система реагирует на команды. Для SSD-накопителя вы не сможете услышать, что используется раздел подкачки, но сможете оценить, сколько места на разделе подкачки занято, используя команды top или free. По возможности следует избегать использования SSD-накопителя для раздела подкачки. Первой реакцией на активность раздела подкачки должна быть проверка на необоснованное применение какой-либо команды, например, попытка редактирования пятигигабайтного файла. Если использование раздела подкачки становится обычным явлением, лучшее решение — приобретение большего объема оперативной памяти для вашей системы.

2.4.1.3. Раздел GRUB

Если загрузочный диск размечен с помощью таблицы разделов GUID (GPT), необходимо создать небольшой раздел, обычно размером 1 МБ, если он еще не существует. Этот раздел не форматируется, но должен быть доступен для использования GRUB во время установки загрузчика. Обычно он помечен как 'BIOS Boot' при использовании fdisk или имеет код EF02 при использовании gdisk.

[Примечание] Примечание Раздел Grub Bios должен находиться на диске, который BIOS использует для загрузки системы. Это не обязательно тот же диск, на котором расположен корневой раздел LFS. Диски в системе могут использовать разные типы таблиц разделов. Наличие раздела Grub Bios зависит только от типа таблицы разделов на загрузочном диске.

2.4.1.4. Разделы, используемые для удобства

Есть несколько других разделов, которые не являются обязательными, но их следует учитывать при разработке схемы диска. Следующий список не является исчерпывающим, а представлен в качестве справочного руководства.

- /boot – Настоятельно рекомендуется. Используйте этот раздел для хранения ядер и другой загрузочной информации. Чтобы свести к минимуму возможные проблемы с загрузкой дисков большого размера, сделайте этот раздел первым физическим разделом на первом диске. Размер раздела в 200 мегабайт вполне достаточен.
- /boot/efi – Системный раздел EFI, используемый для загрузки системы с помощью UEFI. Подробнее читайте на странице BLFS.
- /home – Настоятельно рекомендуется. Предоставьте общий доступ к своему домашнему каталогу и пользовательским настройкам нескольким дистрибутивам или сборкам LFS. Размер, как правило, довольно большой и зависит от доступного места на диске.
- /usr – в LFS, /bin, /lib, и /sbin являются символическими ссылками на их аналоги в /usr. Таким образом /usr содержит все двоичные файлы, необходимые для работы системы. Для LFS отдельный раздел /usr не требуется. Если он вам необходим, вы должны сделать раздел достаточно большим, чтобы поместить туда все программы и библиотеки в системе. В этой конфигурации, корневой раздел может быть очень маленьким (возможно, всего один гигабайт), поэтому он подходит для тонкого клиента или бездисковой рабочей станции (где /usr монтируется с удаленного сервера). Однако вы должны знать, что для загрузки системы с отдельного раздела /usr потребуется initramfs (не включенный в LFS).
- /opt – Этот каталог наиболее полезен для BLFS, в него можно установить некоторые большие пакеты, такие как KDE или Texlive, без использования иерархии /usr. Для /opt достаточно размера от 5 до 10 гигабайт.
- /tmp – По умолчанию, systemd монтирует здесь tmpfs. Если вы хотите переопределить это поведение, следуйте инструкции Раздел 9.10.3, «Отключение tmpfs для /tmp» при настройке системы LFS.
- /usr/src – Этот раздел очень удобен для хранения исходников BLFS и совместного использования их в сборках LFS. Его также можно использовать в качестве места для сборки пакетов BLFS. Размера в 30-50 гигабайт вполне достаточно.

Любой отдельный раздел, который вы хотите автоматически монтировать при загрузке, должен быть указан в файле /etc/fstab. Подробности о том, как указать разделы, будут обсуждаться в Раздел 10.2, «Создание файла /etc/fstab».

2.5. Создание файловой системы на разделе

Раздел - это всего лишь диапазон секторов на диске, указанный в таблице разделов. Прежде чем операционная система сможет использовать раздел для хранения каких-либо файлов, он должен быть отформатирован, чтобы содержать файловую систему, обычно состоящую из метки, блоков каталогов, блоков данных и схемы индексации для поиска конкретного файла по запросу. Файловая система также помогает операционной системе отслеживать свободное пространство на разделе, резервировать необходимые секторы при создании нового файла или расширении существующего и повторно использует свободные сегменты данных, полученные в результате удаления файлов. Она также может обеспечивать поддержку

избыточности данных и восстановления после ошибок.

LFS может использовать любую файловую систему, распознаваемую ядром Linux, но наиболее распространенными типами являются ext3 и ext4. Выбор правильной файловой системы может быть сложным; это зависит от характеристик файлов и размера раздела. Например:

ext2	подходит для небольших разделов, которые редко обновляются, например /boot.
ext3	это обновленная файловая система ext2, которая включает в себя журнал, помогающий восстановить состояние раздела в случае некорректного завершения работы. Обычно используется в качестве файловой системы общего назначения.
ext4	является последней версией файловых систем семейства ext. Она предоставляет несколько новых возможностей, включая временные метки с точностью до наносекунды, создание и использование очень больших файлов (16 ТБ) и повышение скорости работы.

Другие файловые системы, включая FAT32, NTFS, ReiserFS, JFS и XFS, полезны для конкретных задач. Более подробную информацию об этих файловых системах и многих других можно найти по адресу https://en.wikipedia.org/wiki/Comparison_of_file_systems.

LFS предполагает, что корневая файловая система (/) имеет тип ext4. Чтобы создать файловую систему ext4 на разделе LFS, выполните следующую команду:

```
mkfs -v -t ext4 /dev/<xxx>
```

Замените <xxx> именем раздела LFS

Если вы используете существующий раздел подкачки, нет необходимости его форматировать. Если был создан новый раздел подкачки, его нужно будет инициализировать с помощью этой команды:

```
mkswap /dev/<yyy>
```

Замените <yyy> именем раздела подкачки.