

Министерство образования и науки РФ
Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и кибербезопасности
Высшая школа программной инженерии

Лабораторная работа №6

“Сборка ядра Linux”

Работу выполнил студент
Железняков Марк Викторович
Группа: 5130904/30005
Руководитель: Петров Александр Владимирович

Содержание

Содержание	1
Аппаратная платформа	1
Программная платформа	2
Подготовка к выполнению работы	3
Подготовка к сборке ядра	3
Сборка ядра	3
Выполнение работы	4

Аппаратная платформа

Honor MagicBook BOHK-WAX9X M1010

CPU: AMD Ryzen 5 3500U with Radeon Vega Mobile Gfx

Программная платформа

NAME="Arch Linux"

PRETTY_NAME="Arch Linux"

ID=arch

BUILD_ID=rolling

ANSI_COLOR="38;2;23;147;209"

HOME_URL="https://archlinux.org/"

DOCUMENTATION_URL="https://wiki.archlinux.org/"

SUPPORT_URL="https://bbs.archlinux.org/"

BUG_REPORT_URL="https://bugs.archlinux.org/"

PRIVACY_POLICY_URL="https://terms.archlinux.org/docs/privacy-policy/"

LOGO=archlinux-logo

Задание

1. Установить исходный код ядра, предоставляемый вашим дистрибутивом (ванильная версия не рекомендуется).
2. Сконфигурировать и собрать ядро из установленных исходных файлов. 3. Протестировать систему с новым ядром.
3. Разработать сценарий, который запускает сборку ядра в цикле для -jN со значениями от 1 до 2N+1, где N – число ядер в системе, включая виртуальные.
4. Число ядер можно узнать по `cat /proc/cpuinfo`. Сценарий возвращает только время работы сборки на процессоре (используйте `time`, а все сообщения `make-kpkg` перенаправляйте в `/dev/null`). На каждой итерации очищайте дерево исходного кода (например, `make-kpkg clean`).
5. Предоставить отчет о проделанной работе. Дополнительно необходимо предоставить файл конфигурации ядра.

6. Отчет и файл конфигурации необходимо представить в виде архива, названного в соответствии со следующим шаблоном: <первая буква имени студента><фамилия студента><номер группы студента>.
7. После согласования с преподавателем предоставить отчёт.

Цели

1. Сконфигурированное и собранное ядро Linux.
2. Время сборки ядра при различном числе потоков сборки.
3. Нахождение оптимального числа потоков для сборки ядра.
4. Выполнение индивидуального задания.

Задачи

1. Подготовка системы.
2. Установка исходного кода ядра.
3. Конфигурация ядра.
4. Сборка ядра.
5. Установка ядра.
6. Очищение дерева сборки.
7. Проверка работоспособности.
8. Написание сценария, собирающего ядро на потоках от 1 до $2N+1$ и выводящего время.
9. Поиск оптимального числа потоков для сборки ядра.
10. Замер времени сборки ядра без символьной информации (для ИЗ).
11. Замер времени сборки ядра без символьной информации с использованием `ssache` без очищения дерева сборки (для ИЗ).
12. Подведение итогов.

Подготовка к выполнению работы

Подготовка к сборке ядра

1. Установил пакеты:
 - a. `devtools` - набор программ для работы с пакетной базой Arch Linux. Необходим для `pkgctl`
 - b. `base-devel` - группа пакет с нужными для сборки зависимостями:
 - i. GNU Compiler Collection;
 - ii. Flex + Bison - инструментарий для создания программ со структурированным вводом;
 - iii. `Fakeroot` - инструмент для имитации прав владения суперпользователя
2. Получил исходники ядра посредством исполнения `pkgctl repo clone --protocol=https linux`
3. Внес изменения в сценарий сборки `PKGBUILD`:
 - a. Отключил генерацию документации (`make htmldocs`)
 - b. Указал количество потоков при сборке
 - c. Добавил опцию открытия `nconfig` перед началом сборки
 - d. Изменил название пакета ядра во избежание конфликтов

Сборка ядра

Выполнил команду `makepkg -s`. После успешной сборки в репозитории появились пакеты ядра `linux-mrqiz-6.7.4.arch1-1-x86_64.pkg.tar.zst` и `linux-mrqiz-headers-6.7.4.arch1-1-x86_64.pkg.tar.zst`

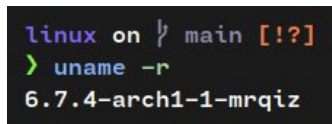
Установка ядра

1. Выполнил команду для установки пакетов ядра: `sudo pacman -U linux-mrqiz-6.7.4.arch1-1-x86_64.pkg.tar.zst linux-mrqiz-headers-6.7.4.arch1-1-x86_64.pkg.tar.zst`
2. Обновил конфигурацию загрузчика GRUB: `sudo grub-mkconfig -o /boot/grub/grub.cfg`

Загрузка ядра

Для загрузки собранного ядра в меню GRUB выбрал Advanced options for Arch Linux, после этого выбираю необходимое ядро.

После этого в командной строке была выполнена команда `uname -r`, отображающая версию ядра:

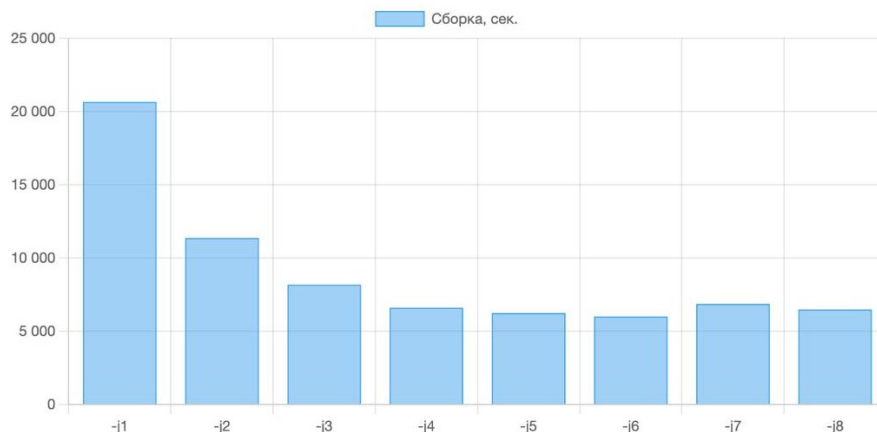


```
linux on / main [!?]
> uname -r
6.7.4-arch1-1-mrqiz
```

Выполнение работы

1. Был написан сценарий итеративной сборки BuildBench. Сценарий изолирует каждую итерацию сборки на N потоках в своей директории, сохраняя логи сборки в отдельный файл. Получить сценарий можно по ссылке shell.inkling.su/buildbench.sh, документация доступна на [GitHub Gist](#).
2. PKGBUILD был модифицирован: `make -j8` заменен `make -j$GCC_J`
3. Сценарий был запущен со следующим набором опций:
`BUILDBENCH_SOURCEDIR=~/.linux`
`BUILDBENCH_TARGETDIR=~/.buildbench-linux-sandbox`
`BUILDBENCH_TIDY=true BUILDBENCH_CMD='GCC_J=$JOBS makepkg -s'`
`BUILDBENCH_JOBS=8`

График соотношения количества потоков ко времени сборки:



Содержимое общего журнала сборки:

BuildBench @ 03/03/24 08:51:28

Source directory: /home/mrqiz/linux-abs

Target directory: /home/mrqiz/buildbench-linux-sandbox

Max processing units: 8

Using 1 cores - 20649.384s

Using 2 cores - 11366.213s

Using 3 cores - 8176.932s

Using 4 cores - 6605.947s

Using 5 cores - 6246.528s

Using 6 cores - 6009.113s

Using 7 cores - 6865.210s

Using 8 cores - 6482.969s

Finished at 03/04/24 04:58:13 (took 72405.296s).

Заключение

1. В результате выполнения данной работы была достигнута цель поиска наиболее подходящего количества потоков при сборке ядра. Была выполнено 8 итераций сборки ядра.
2. Прделанные действия в ходе выполнения работы:
 - a. Установка зависимостей для сборки;
 - b. Скачивание исходного кода для ядра, поддерживаемого дистрибутивом;
 - c. Внесение изменений в сценарий сборки PKGBUILD;
 - d. Сборка ядра;
 - e. Установка собранного ядра;
 - f. Загрузка ядра;
 - g. Написание сценария BuildBench;
 - h. Поиск оптимального количества потоков для сборки.

3. Самым оптимальным количеством потоков на аппаратной платформе является 6.