Министерство образования и науки РФ

Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Высшая школа программной инженерии

Отчёт по лабораторной работе №6

по дисциплине «Системное программное обеспечение GNU/Linux»

Сборка ядра Linux

Работу выполнил студент

Хикматова Аиша Хасан кизи

Группа: 3530904/20006

Руководитель:

Петров Александр Владимирович

Санкт-Петербург 2023

1

**Содержание**

Содержание........................................................................................................................................ 2 Аппаратнаяплатформа....................................................................................................................... 3 Программнаяплатформа.................................................................................................................... 3 Задание................................................................................................................................................ 4 Цели..................................................................................................................................................... 4 Задачи.................................................................................................................................................. 4 Подготовка к выполнению работы.................................................................................................. 5 Подготовка к сборке ядра................................................................................................................. 5

Сборка ядра........................................................................................................................................ 5

Установка ядра.................................................................................................................................. 5

Завершение..........................................................................................................................................5 Выполнение работы........................................................................................................................... 6 Заключение......................................................................................................................................... 7

2

**Аппаратная платформа:** СPU: Apple M1

8-core CPU 7-core GPU RAM:8GB SSD: 256GB

**Программная платформа:**

3

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.

4. 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.

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 ccache без очищения дерева сборки (для ИЗ).

12. Подведение итогов.

Подготовка к выполнению работы: Подготовка к сборке ядра:

1. Перешел в каталог, который будет использоваться для сборки ядра. *cd /usr/src/*

2. Скачала и распаковал последнее стабильное ядро версии **6.2.2**. *Wget https://cdn.kernel.org/pub/linux/kernel/v6.x/ linux-6.2.2.tar.xz tar -xvJf linux-6.2.2.tar.xz*

*-x распаковывает файлы из архива*

*-v выводит всю информацию о текущем процессе и показывает сведения об завершенном процессе*

*-J фильтрует архив через xz*

*-f выводит результат файлов*

3. Установил все нужные пакеты.

*apt install fakeroot libncurses5-dev debhelper dh-make devscripts build-essential automake bison flex*

4. Скопировал прежнюю конфигурацию ядра sudo *cp -v /boot/config-$(uname -r) .config*

5. Запустил конфигурацию и сохранил *sudo make menuconfig*

Сборка ядра:

1. Сборка ядра:

Sudo make -j4

2. После успешной сборки очистил дерево сборки sudo make clean

**Установка ядра**:

1. sudo make

2. sudo make modules

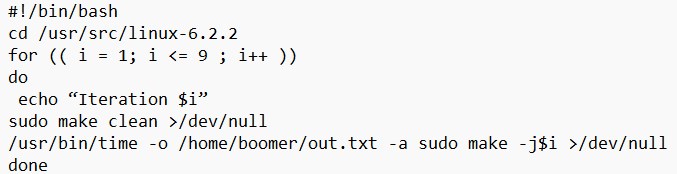
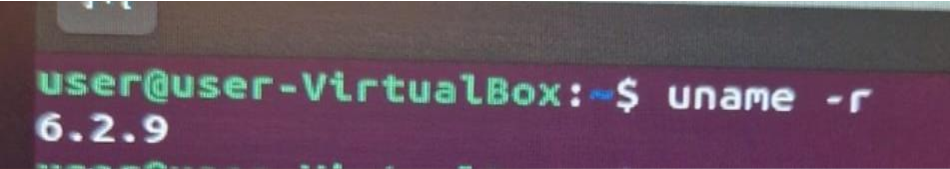
3. sudo make modules\_install

4. sudo make install

5. reboot

Завершение, проверка на правильность установки:

5



Выполнение работы:

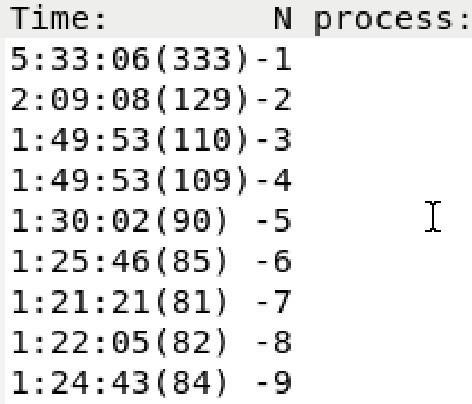
1. написала скрипт для определения скорости сборки ядра на разном кол-ве потоков nano fstscript.sh

Запуск скрипта:

1. chmod +x fstscript.sh

2. ./ fstscript.sh

6

Спроектирован график по результатам работы скрипта

Время(min) относительно «N process»

400 4



300 3



200 2



100 1



0 0 1 2 3 4 5 6 7 8 9 10 N



N process

7

Заключение:

1. В результате выполнения данной работы была достигнута цель поиска наиболее подходящего количества потоков при сборке ядра, всего было пересобрано N+1 ядро, где N – число ядер процессора. Кроме того, сборка на 0 потоков оказалась невозможна.

2. Проделанные действия в ходе выполнения задач: 2.1. Подготовка системы.

2.2. Установка исходного кода ядра. 2.3. Конфигурация ядра.

2.4. Сборка ядра. 2.5. Установка ядра.

2.6. Очищение дерева сборки.

2.7. Проверка работоспособности.

2.8. Написание сценария, собирающего ядро на потоках от 1 до N+1 и выводящего время.

2.9. Поиск оптимального числа потоков для сборки ядра.

3. По результатам тестового сценария, собирающего ядро на разном количестве потоков, было выявлено, что наиболее оптимальным числом потоков при сборке является N = 7.

4. Возникли трудности при первой сборке ядра, была ошибка *make[1]: \*\*\* No rule to make target 'debian/canonical-certs.pem', needed by 'certs/x509\_certificate\_list'. Stop. make: \*\*\* [Makefile:1809: certs] Error 2*

Проблема была решена поиском ошибки в интернете и следующими изменениями в конфигурации ядра: *CONFIG\_SYSTEM\_TRUSTED\_KEYS=””, CONFIG\_SYSTEM\_REVOCATION\_KEYS=””*

Индивидуальное задание

Цель работы – Ускорение повторной сборки ядра с помощью ccache

Задание – собрать ядро без использования символьной информации. После собрать ядро повторно (не очищая дерево сборки) без использования и с использованием инструмента ccache; определить ускорение времени повторной сборки за счёт инструмента ccache.

Выполнение

Для сборки ядра с помощью ccache требуется его установка и настройка

sudo apt-get install ccache

На всякий случай установил максимальный размер кэша ccache

ccache -M 10G

добавил и изменил некоторые переменные окружения

export PATH=”/usr/lib/ccache:$PATH” позволяет использовать ccache вместо обычных компиляторов

export CC=”ccache gcc” добавляет ccache к gcc

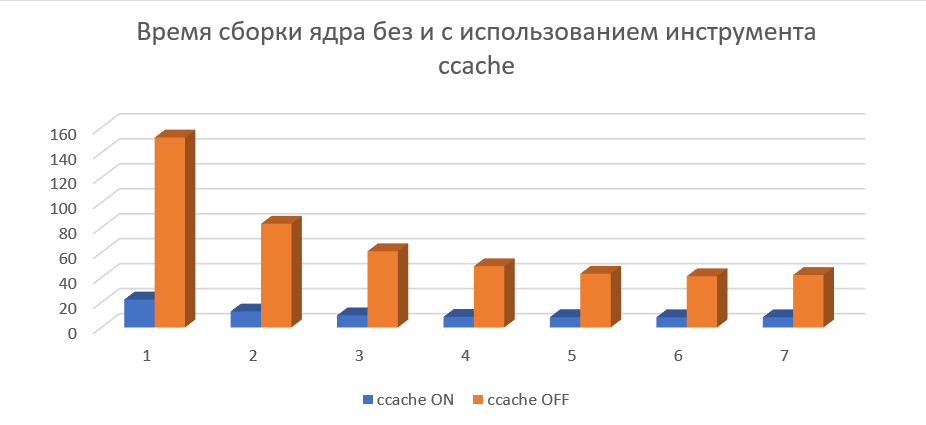
export CXX=”ccache g++” добавляет ccache к g++

запустил сборку ядра командой из лабораторной 6, после первого запуска, изменений не заметил. Со второго запуска стал наблюдать заметное уменьшение времени компиляции в несколько раз

time sudo make -jN

Время сборки ядра без использования инструмента ccache и с использованием данного инструмента:

|  |  |  |
| --- | --- | --- |
| Количество потоков | Время сборки ядра с ccache (мин:сек) | Время сборки ядра без использования ccache(мин) |
| 1 | 22:28 | 152 |
| 2 | 12:49 | 83 |
| 3 | 9:40 | 61 |
| 4 | 8:36 | 49 |
| 5 | 8:15 | 43 |
| 6 | 8:13 | 41 |
| 7 | 8:14 | 42 |



## 

## Вывод

В результате проведённой сборки ядра с использованием инструмента ccache и без него мы можем сделать вывод, что в среднем время сборки ядра существенно сократилась, примерно в 7 раз

Возникшие трудности

По началу не мог разобраться куда именно сохраняется кэш ccache, но прописав строчку с указанием пути “”export CCACHE\_DIR=/home/hhasan/ccache””смог сам посмотреть какие конкретно файлы уходят в кэш

Так как оптимальным числом потоков для сборки ядра являлось 6 потоков, было принято решение проверять работоспособность инструмента именно на 6 потоках, но время компиляции 15:04 меня не устроило, и путем повторного замера было зафиксировано время 8:13, что опять является оптимальным числом потоков для сборки ядра с использованием инструмента ccache.

8

9