Лабораторная работа №6

“Сборка ядра Linux”

Работу выполнил студент

Березнева Екатерина Ивановна

Группа: 5130904/30005

Руководитель:

Петров Александр Владимирович

# **Содержание**

[Содержание 2](#_Toc4003)

[Аппаратная платформа 2](#_Toc4004)

[Программная платформа 2](#_Toc4005)

[Задание 2](#_Toc4006)

[Подготовка к выполнению работы 4](#_Toc4007)

[Подготовка к сборке ядра 4](#_Toc4008)

[Сборка ядра: 4](#_Toc4009)

[Завершение и проверка на правильность установки: 5](#_Toc4010)

[Выполнение работы: 5](#_Toc4011)

[Заключение 6](#_Toc4012)

# **Аппаратная платформа**

HONOR Magicbook модель: NMH-WDX9

СPU: AMD Ryzen 7 5700U with Radeon Graphics, 1.80 GHz, 8-core, CPU 16-core

# **Программная платформа**

PRETTY\_NAME="Debian GNU/Linux 12 (bookworm)"

NAME="Debian GNU/Linux"

VERSION\_ID="12"

VERSION="12 (bookworm)"

VERSION\_CODENAME=bookworm

ID=debian

HOME\_URL="https://www.debian.org/"

SUPPORT\_URL="https://www.debian.org/support"

BUG\_REPORT\_URL="<https://bugs.debian.org/>"

# **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.
5. Число ядер можно узнать по cat /proc/cpuinfo. Сценарий возвращает только время работы сборки на процессоре (используйте time, а все сообщения перенаправляйте в /dev/null). На каждой итерации очищайте дерево исходного кода (например, make clean).
6. Предоставить отчет о проделанной работе. Дополнительно необходимо предоставить файл конфигурации ядра.
7. Отчет и файл конфигурации необходимо представить в виде архива, названного в соответствии со следующим шаблоном: <первая буква имени студента><фамилия студента><номер группы студента>.
8. 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. Собрать ядро для дистрибутива, используя исходный код, предоставляемый дистрибутивом.(для ИЗ).
11. Собрать ядро для дистрибутива, используя исходный код, предоставляемый ресурсом kernel.org, с наложением на него заплат от дистрибутива.(для ИЗ).
12. Подведение итогов.

# **Подготовка к выполнению работы**

## Подготовка к сборке ядра

1. Установка пакетов:

* build-essential - набор пакетов, необходимых для компиляции программ в Debian
* kernel-package - инструмент для создания пакетов ядра Debian
* gcc - компилятор GNU для языка C
* make - утилита для автоматизации компиляции программ
* libncurses5-dev - библиотека для работы с текстовым пользовательским интерфейсом
* dpkg-dev - инструменты для работы с форматом пакетов Debian

1. Загрузила и извлекла исходный код ядра:

◦ wget <https://www.kernel.org/pub/linux/kernel/v6.x/linux-6.7.5.tar.xz> && tar xvf linux-6.7.5.tar.xz

1. Перешла в каталог, который будет использоваться для сборки ядра и скопировала существующий файл конфигурации: ◦ cd home/catalina/linux-6.7.5

◦ cp -v /boot/config-$(uname -r) .config

# **Сборка ядра:**

1. sudo make -j8 – сборка ядра на 8 потоках
2. sudo make install\_modules – команда для установки модулей ядра
3. sudo make install – команда для установки ядра
4. reboot

# **Завершение и проверка на правильность установки:**

Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Автоматически созданное описание

# **Выполнение работы:**

Написала скрипт для определения скорости сборки ядра на разном количестве потоков:

1. touch diffSpeed.sh
2. nano diffSpeed.sh

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Запуск скрипта:

1. sudo su
2. chmod +x diffSpeed.sh
3. ./ diffSpeed.sh

График соотношения количества потоков ко времени сборки:

|  |  |
| --- | --- |
| N, потоков | Время, сек |
| 1 | 10327,743 |
| 2 | 5050,010872 |
| 3 | 3519,208746 |
| 4 | 2829,08889 |
| 5 | 2536,604242 |
| 6 | 2150,62006 |
| 7 | 2188,038221 |
| 8 | 2180,484609 |
| 9 | 2084,408114 |
| 10 | 2020,88439 |
| 11 | 2107,619981 |
| 12 | 2018,389557 |
| 13 | 1974,314375 |
| 14 | 1947,346622 |
| 15 | 1918,368126 |
| 16 | 1867,138816 |

# 

# **Заключение**

1. В результате выполнения данной работы была достигнута цель поиска наиболее подходящего количества потоков при сборке ядра. Была выполнено 16 итераций сборки ядра.
2. Проделанные действия в ходе выполнения работы:

◦ Установка зависимостей для сборки;

◦ Скачивание исходного кода для ядра, поддерживаемого дистрибутивом;

◦ Сборка ядра;

◦ Установка собранного ядра;

◦ Загрузка ядра;

◦ Написание скрипта для определения скорости сборки ядра на разном количестве потоков;

◦ Поиск оптимального количества потоков для сборки;

1. Самым оптимальным количеством потоков на аппаратной платформе является 16.