Мережеве програмування в середовищі Unix
Лабораторна робота №1
Налаштування середовища розробки

Для того щоб виконувати лабораторні роботи з практичної дисципліни, такої як «Мережеве програмування в середовищі Unix», треба налаштувати відповідне середовище для розробки програм. Ця лабораторна робота присвячена вибору програмного забезпечення та його налаштуванню для виконання наступних лабораторних робіт.

Вибір Unix-подібної операційної системи

Перше, що треба зробити, – це вибрати Unix-подібну операційну систему, в якій буде відбуватися розробка та виконання розроблених програм. Бажано обрати сучасну версію Unix-подібної ОС рівня production. Така система має бути (майже) POSIX-сумісною. Є кілька варіантів інсталяції такої ОС:

1. ОС можна інсталювати на реальний комп'ютер. Для цього треба мати достатньо вільного простору в накопичувачі та достатньо оперативної пам'яті. Якщо комп'ютер використовується разом з іншої ОС, тоді може знадобитися перерозподіл наявної ФС для звільнення місця в накопичувачі та треба визначити як буде вибиратися ОС під час завантаження (неправильне виконання цих дій призведе до того, що наявна ОС та/або інстальована ОС не завантажуватиметься, а наявна ФС буде неправильно змінена).

2. ОС можна інсталювати у віртуальній машині, що виконується в наявній ОС. Для цього треба мати достатньо вільного простору в наявній ФС, достатність оперативної пам'яті буде впливати на швидкість роботи віртуальної машини. В залежності від наявної ОС та ФС зарезервований простір для накопичувача у віртуальні машині не обов'язково буде використаний повністю відразу. Така інсталяція ОС не може нічого порушити в наявні ОС та ФС.

3. ОС можна використовувати в різних середовищах віртуалізації, які надає наявна ОС. Зазвичай такий підхід реалізований в Unix-подібних ОС. Тому цей підхід для лабораторних робіт не має сенсу, оскільки можна використовувати наявну ОС для розробки та виконання розроблених програм.

4. Якщо на комп'ютері використовується Windows, тоді можна скористатися Windows Subsystem for Linux (WSL). Для цього треба відповідна версія Windows та наявність відповідної технології віртуалізації в процесорі.

Вибір компілятора

Оберіть компілятор, який відповідає сучасному стандарту мови системного програмування, якою ви буде розробляти програми. Для C та C++ можна обрати компілятор з найновішої версії **gcc** або **clang** та **g++** або **clang++** відповідно. Компілятор доцільно інсталювати програмними засобами, які надає ОС для інсталяції пакетів.

Документація

Треба завантажити копію Single UNIX Specification, version 4, найновішого видання з сайту `https://www.opengroup.org/` (там треба зареєструватися, після реєстрації можна читати документацію online або завантажити архів, також можна знайти URL цього документа на цьому сайті).

Треба інсталювати man-сторінки 2-го (системні виклики) та 3-го (функції бібліотек) розділів. Якщо є переклади man-сторінок, тоді краще послуговуватись оригінальною документацію, тобто документацією написаною англійською. Man-сторінки інсталюються програмними засобами, які надає ОС для інсталяції пакетів.

Бажано завантажити копію стандарту мови програмування, якою ви будете розробляти програми. Для C та C++ посилання на документи з draft-версіями стандартів можна знайти на сайті `https://cppreference.com/` (цей сайт також можна використовувати як джерело документації для C та C++).

Вибір середовища розробки

Є кілька варіантів вибору середовища для розробки програм:

1. Можна використовувати текстовий редактор для редагування тексту програми, який може мати тільки базові засоби для редагування текстів та виконує підсвічування тексту програми.

2. Можна використовувати текстовий редактор для редагування тексту програми, який має засоби для редагування текстів, виконує підсвічування тексту програми та дозволяє налаштувати компіляцію та збирання програми (також може дозволяти розширювати свій функціонал за рахунок залучення сторонніх модулів).

3. Можна використовувати IDE, яке надає весь функціонал для розробки програм в одному середовищі і це середовище вже має бути налаштовано самою IDE.

Самостійно оберіть середовище розробки, яке для вас буде зручнішим. Для цього можна подивитися в Internet порівняння різник редакторів та IDE, які використовуються для розробки програм відповідною мовою програмування.

Програми для налагодження

Під час розробки програми, яка працює з мережею, іноді треба перевіряти коректність її роботи з мережевими з’єднаннями. Це можна зробити шляхом додавання виведення повідомлень до вихідного коду програм, це схоже на звичайне налагодження або на увімкнений режим виведення налагоджувальної інформації в програмі. Якщо треба перевірити куди та які дані відправляє програма або звідки та які дані отримує програма, тоді треба увімкнути в ядрі відповідну підтримку (зазвичай така підтримку увімкнута за замовченням) та використати програми, які дозволяють отримувати дані ядра, які мають відношення до мережевих з’єднань, та/або перехоплювати дані, які відправляються мережевими з’єднаннями.

Програма **netstat** виводить різноманітну інформацію про мережеві з’єднання. Програми **tcpdump** (консольна програма, CLI) та **Wireshark** (GUI) виводять інформацію про мережевий трафік у різні способи. Програма **lsof** виводить інформацію, що має відношення до дескрипторів файлів. Програма **nc** або **ncat** дозволяє відправляти дані в та отримувати дані з мережевих з’єднань, цю програму можна використовувати для тестування як клієнтської та серверної частин програми, що розробляється. Також будуть у нагоді програми **nslookup** та **dig** для отримання інформації про доменні імена та мережеві адреси.

Автоматизація компіляції та збирання програм

Вихідний код програми можна компілювати та збирати вручну або використовувати спеціально розроблені для цього програмні засоби. Якщо використовується IDE, тоді це середовище розробки може надавати власні засоби автоматизації компіляції та збирання програм. Далі розглядаються stand alone варіанти автоматизації компіляції та збирання програм.

Для програм, які написані мовою програмування C або C++, автоматизація компіляції та збирання виконується за допомогою використання **Makefile**'ів та програми **make**. Але зазвичай замість ручного редагування **Makefile**'ів використовують спеціальні програми, які генерують необхідні **Makefile**'и. Таких програм є декілька. Можна використовувати **autoconf**, **automake** та **libtool**. Але є тренд на використання **CMake**, тому далі розглянемо мінімальний варіант конфігурації для **CMake**.

Нехай є директорія **prog**, де будуть міститися імена файлів програми, яка буде розроблятися мовою програмування C. Створимо директорію **prog/src**, ця директорія буде містити імена файлів вихідного коду програми. Нехай вихідний код програми складається з одного файлу **main.c**. Створюємо два файли **CMakeLists.txt**:

```
$ cd prog
$ cat CMakeLists.txt
cmake_minimum_required(VERSION 3.8)
project(Prog LANGUAGES C)
add_subdirectory(src)
$ cat src/CMakeLists.txt
set(CMAKE_C_EXTENSIONS off)
set(CMAKE_C_STANDARD 23)
if(CMAKE_C_COMPILER_ID STREQUAL "GNU" OR
   CMAKE_C_COMPILER_ID MATCHES "Clang")
    add_compile_options(-Wall -Wextra -Wconversion -pedantic)
endif()
add_executable(prog main.c)
$
```

Створюємо мінімально необхідний вміст файлу **main.c**.

Створюємо директорію **src/build**, де будуть міститися імена файлів, необхідних для автоматизації компіляції та збирання програми:

```
$ mkdir build
$ cd build
$ cmake ..
```

Тепер можна виконати програму **make** у директорії **prog/build** для компіляції та збирання програми.

Наведемо вміст файлів **CMakeLists.txt** для програми, яка буде розроблятися мовою програмування C++:

```
$ cd prog
$ cat CMakeLists.txt
cmake_minimum_required(VERSION 3.8)
project(Prog LANGUAGES CXX)
add_subdirectory(src)
$ cat src/CMakeLists.txt
set(CMAKE_CXX_EXTENSIONS off)
set(CMAKE_CXX_STANDARD 23)
```

```
if(CMAKE_CXX_COMPILER_ID STREQUAL "GNU" OR
   CMAKE_CXX_COMPILER_ID MATCHES "Clang")
    add_compile_options(-Wall -Wextra -Wconversion -pedantic)
endif()
add_executable(prog main.cpp)
$
```

До `add_compile_options()` можна додати опції `-g -O2`.

Може бути так, що під час компіляції програми написаною мовою програмування C або C++ компілятор повідомляє про відсутність декларації функції із POSIX або повідомляє якісь інші помилки, які маються відношення до POSIX. Для того, щоб вказати який стандарт SUS підтримувати під час компіляції програми, треба призначити відповідне значення макросу `_XOPEN_SOURCE`. У разі використання **CMake** це можна зробити, додавши такий рядок до файлу **src/CMakeLists.txt**:

```
add_compile_definitions(_XOPEN_SOURCE=700)
```

У цьому випадку підтримуватиметься стандарт SUSv4.

Звіт

Виконання цієї лабораторної роботи не потребує звіту та не оцінюється балами.