Лекция 2
GNU Toolchain

Проект GNU

- Возникновение — 80-е годы XX века
- Философская основа: что такое программное обеспечение, что такое информация, как соотносятся «свобода слова», «свобода обмена информацией» и «авторское право» (copyright)
- Проект GNU: программное обеспечение — это информация, знания
- Недопустимо препятствовать распространению знания

Свободное ПО (4 свободы)

- Свобода выполнять программу как вам угодно в любых целях
- Свобода изучать работу программы и модифицировать программу — предполагает доступ к исходному тексту
- Свобода передавать копии ПО другим людям
- Свобода передавать модифицированные копии ПО другим людям

GNU Public License (GPL)

- Лицензионное соглашение (гражданско-правовой договор) между автором ПО и пользователями
- Программа распространяется вместе с исходными текстами, и любой пользователь обладает 4 свободами
- При последующем распространении не должны ущемляться 4 свободы использования
- GPLv3: защита от патентных «троллей», защита от закрытия в прошивках, защита от DRM

Другие формы Open Source

- Public Domain
- MIT-style license
- BSD-style license

Отказ от гарантий

ЭТА ПРОГРАММА ПРЕДОСТАВЛЕНА ВЛАДЕЛЬЦАМИ АВТОРСКИХ ПРАВ И/ИЛИ ДРУГИМИ СТОРОНАМИ **«КАК ОНА ЕСТЬ» БЕЗ КАКОГО-ЛИБО ВИДА ГАРАНТИЙ**, ВЫРАЖЕННЫХ ЯВНО ИЛИ ПОДРАЗУМЕВАЕМЫХ, ВКЛЮЧАЯ, НО НЕ ОГРАНИЧИВАЯСЬ ИМИ, ПОДРАЗУМЕВАЕМЫЕ ГАРАНТИИ КОММЕРЧЕСКОЙ ЦЕННОСТИ И ПРИГОДНОСТИ ДЛЯ КОНКРЕТНОЙ ЦЕЛИ. НИ В КОЕМ СЛУЧАЕ **НИ ОДИН ВЛАДЕЛЕЦ АВТОРСКИХ ПРАВ** И НИ ОДНО ДРУГОЕ ЛИЦО, КОТОРОЕ МОЖЕТ ИЗМЕНЯТЬ И/ИЛИ ПОВТОРНО РАСПРОСТРАНЯТЬ ПРОГРАММУ, КАК БЫЛО СКАЗАНО ВЫШЕ, **НЕ НЕСЁТ ОТВЕТСТВЕННОСТИ**, ВКЛЮЧАЯ ЛЮБЫЕ ОБЩИЕ, СЛУЧАЙНЫЕ, СПЕЦИАЛЬНЫЕ ИЛИ ПОСЛЕДОВАВШИЕ УБЫТКИ, ВСЛЕДСТВИЕ ИСПОЛЬЗОВАНИЯ ИЛИ НЕВОЗМОЖНОСТИ ИСПОЛЬЗОВАНИЯ ПРОГРАММЫ (ВКЛЮЧАЯ, НО НЕ ОГРАНИЧИВАЯСЬ ПОТЕРЕЙ ДАННЫХ, ИЛИ ДАННЫМИ, СТАВШИМИ НЕПРАВИЛЬНЫМИ, ИЛИ ПОТЕРЯМИ ПРИНЕСЕННЫМИ ИЗ-ЗА ВАС ИЛИ ТРЕТЬИХ ЛИЦ, ИЛИ ОТКАЗОМ ПРОГРАММЫ РАБОТАТЬ СОВМЕСТНО С ДРУГИМИ ПРОГРАММАМИ), ДАЖЕ ЕСЛИ ТАКОЙ ВЛАДЕЛЕЦ ИЛИ ДРУГОЕ ЛИЦО БЫЛИ ИЗВЕЩЕНЫ О ВОЗМОЖНОСТИ ТАКИХ УБЫТКОВ.

Проект GNU

- Свободная замена проприетарным версиям Unix
- В начале проекта: emacs, GCC
- Начало 90-х: появление Linux
- В настоящее время: полная операционная система, включая и большой набор прикладного ПО (все ПО под открытой лицензией)

GNU Toolchain

- Комплект утилит для разработки программного обеспечения
  - GNU Compiler Collection (GCC)
  - GNU Binutils — набор утилит для работы с объектными и исполняемыми файлами в разных форматах
  - GNU Make — утилита для управления сборкой программ
  - GNU configure — утилита для конфигурирования ПО

GCC (Gnu Compiler Collection)

- Компиляторы с языков:
  - C, C++, Objective C, Objective C++
  - Fortran
  - Go
  - Java
  - Ada
- Генерация кода на ~40 различных процессорных архитектур

Запуск GCC

- В зависимости от языка запускается программа-драйвер компиляции:
  - C — gcc
  - C++ - g++
  - Fortran - gfortran
  - Objective C - gcc
  - Objective C++ - g++
  - Go - gccgo
  - Java - gcj
  - Ada - gnat

Компоненты GCC

- Собственно компиляторы языков
  - Генерируют программу на языке ассемблера!
- Стандартные библиотеки языков:
  - C++
  - Java

Необходимые дополнительные компоненты

- Ассемблер (as) — должен быть в host-операционной системе
- Компоновщик (ld) — должен быть в host-операционной системе
- GNU as и GNU ld — части GNU Binutils
- Стандартная библиотека Си — должна быть в host-системе
  - Заголовочные файлы
  - Библиотечные файлы (.a или .so)

Автоопределение языка программирования

- Тип содержимого файла определяется по суффиксу его имени
  - .c — файл на C
  - .cpp, .cxx, .cc, .C — файл на C++
  - .i — препроцессированный файл на C
  - .S — непрепроцессированный файл на ассемб.
  - .s — препроцессированный файл на ассемб.
  - .o — объектный файл
  - .a — статическая библиотека

Компиляция C и C++ программ

- Препроцессирование
- Трансляция в ассемблер
- Ассемблирование
- Компоновка и Link-Time optimizations

Управление последовательностью вызовов соответствующих утилит берет на себя драйвер компиляции

gcc **-v** prog.c -o prog

Препроцессирование

- Препроцессор запускается для файлов с расширением .c, .cpp, .S
- Результат препроцессирования — единица компиляции
- Только препроцессирование: gcc -E
- По умолчанию вывод на stdout, можно использовать -o FILE для сохранения в файл
- Опция -I позволяет задавать каталоги для заголовочных файлов
- Опция -D позволяет определять макросы из командной строки

Трансляция

- Трансляция выполняется если не задана опция -E
- Трансляция выполняется для файлов-результатов препроцессирования и файлов с суффиксами .i (C) или .ii (C++)
- Завершить работу после шага трансляции можно с помощью опции -S
- Файл с программой на ассемблере называется FILE.s

```
gcc prog.c -O2 -S
```

Опции трансляции

- Большое количество различных опций
  - Разные оптимизации: -O0, -O1, -O2, -Os
  - Выдача предупреждений: -Wall, -Werror, -Wno-pointer-sign, …
  - Настройка генерации кода: -fPIC, -fomit-frame-pointer, …
  - Версии стандарта -std=gnu++14, -std=gnu11

Ассемблирование

- Запускается, если не заданы -E или -S
- Запускается для файлов-результатов трансляции или для файлов .S (предварительно препроцессируется) или .s
- Завершить работу после ассемблирования (после получения объектных файлов) — опция -c
- Объектный файл по умолчанию имеет имя FILE.o

Компоновка

- Запускается, если не заданы -E, -S, -c
- На компоновку передаются результаты ассемблирования и все файлы из командной строки, суффиксы которых не обрабатываются на предыдущих шагах
- Результат компоновки по умолчанию — файл a.out
- Изменить имя файла-результата можно -o FILE

Опции компоновщика

- -L DIR — позволяет указывать дополнительные каталоги для поиска статических и динамических библиотек
- -lLIB (пробел между -l и именем библиотеки недопустим) — позволяет указывать имена дополнительных библиотек
  - При указании опции -lNAME библиотеки ищутся по именам libNAME.so и libNAME.a
- -shared — генерация динамической библиотеки вместо исполняемого файла

Особенность Unix-компоновки

- Компоновщик — однопроходный
- Библиотеки просматриваются в том порядке, в котором они указаны в командной строке
- Возможна ситуация, когда для разрешения зависимости потребуется еще раз просмотреть предыдущую библиотеку — в этом случае может возникнуть ошибка «unresolved symbol»

Опции трансляции в целом

- Управление отладочной информацией -g
- -m32 — генерация 32-битного кода (если компилятор поддерживает, например, на x86_64)

В зависимости от языка программирования (C или C++) могут добавляться «системные» каталоги для заголовочных файлов и библиотеки

Другие компиляторы

- Clang — разрабатывается Apple, лицензия BSD
- ICC (Intel Compiler Collection) — проприетарный

На Linux компиляторы clang и icc совместимы по опциям командной строки с компилятором gcc

GCC для Windows

- Cygwin — предоставляет ограниченную поддержку для POSIX API в Windows, GCC в Cygiwin позволяет компилировать Win32 приложения
- MinGW — многие GNU утилиты портированные на WinAPI (как WinAPI приложения), содержит GCC

GNU Bintutils

- as — ассемблер
- ld — компоновщик (линкер)
- ar — работа со статическими библиотеками
- objcopy — копирование (и модификация) объектных файлов
- objdump — просмотр объектных файлов
- strings — извлечение строк
- nm — вывод списка символов в файле

Статические библиотеки

- В Unix находятся в .a файлах
- По сути — архивы (коллекция) объектных файлов и индекс, позволяющий для каждого имени определить в каком объектном файле он определен
- Компоновщик выбирает из статических библиотек только требуемые объектные файлы

Динамические библиотеки

- Исполняемые файлы в формате ELF
- Могут размещаться по любому адресу в виртуальном адресном пространстве (position-independent code)
- При запуске программы на выполнение загрузчик подгружает необходимые динамические библиотеки в адресное пространство и разрешает ссылки в исполняемых файлах

Кросс-компиляция

- Host-платформа — платформа, на которой работает компилятор
- Target-платформа — платформа, для которой компилируется код
- Если host != target — кросс-компиляция, если код для target не может быть запущен на host
- Компиляция для x86 на x86_64 — это не кросс-компиляция

Triple

- Идентифицируют платформу
- В целом имеют вид: machine-vendor-operatingsystem
- Часто vendor опускается
- Часто vendor — это pc или unknown

Примеры triple

- i686-redhat-linux
- x86_64-redhat-linux
- arm-linux-gnueabihf
- avr
- sparc-sun-solaris2.10