**Лекция 1**

* 1. **Libraries and Linkage**

4 стадии сборки:

Предпроцессинг, компиляция, ассемблирование, линковка

*g++ -v test.cpp* – узнать подробнее о том, что происходит во время сборки

Первым делом g++ вызывает программу cc1plus – компилятор.

На выходе получим ассемблерный файл.

Далее можем заметить такую команду: as -v –64 -o file.o file.s

Программа as – преобразование ассемблерного кода в бинарный.

В конце collect2 – линковка.

*g++ -E test.cpp* – предпроцессинг – на выходе файл .cpp

*g++ -S test.cpp* – ассемблирование – на выходе файл .s

*g++ -c test.cpp* – линковка – на выходе файл .o

После всех этапов на выходе файл .out

**hexdump** – встроенная утилита Linux, которая позволяет фильтровать и отображать содержимое разных файлов в шестнадцатеричном, десятичном, восьмеричном или ASCII-форматах.

Эта утилита может использоваться для восстановления данных, а также для просмотра исполняемого кода разных программ.

*hexdump -C test.o* – вывод побайтового представления файла с расшифровкой.

**readelf**– утилита, которая позволяет в удобочитаемом виде отобразить всю информацию ELF-файлов.

**ELF – Executable and Linkable Format**. Файл, который подлежит либо выполнению, либо линковке.

*readelf -a a.out* – отобразить все.

У ELF файлов есть заголовок – **Header**. В нем можно найти информацию о файле, в том числе ELF **magic**, который используется для идентификации файлов ELF и представляет собой всего лишь самые первые несколько байт файла.

Дальше идет само содержимое этого файла.

Что обязательно должно быть в содержимом?

.text – в этой секции хранится бинарный код. Указан адрес, с которого она начинается.

Статическая память состоит из трех частей.

* .data – статическая память.
* .rodata – read only data. Пример: std::cout << “Hello World!” – вот такое там хранится.
* .bss - когда Ваша программа начинает работу, эта секция содержит байты с нулевыми значениями. Она служит для хранения статических неинициализированных переменных. (“better save space”)

.symtab – таблица символов.

*readelf -a a.out* – вывести таблицу символов.

**Символы** – функции, о существовании которых знает линковщик.

Когда мы компилируем файл, до стадии линковки в нем могут быть объявления каких-то функций, которые в нашей программе не определены. Такие функции достаются линковщику, и линковщик пытается найти в каком-то другом месте (в какой-то библиотеке) определение этих функций.

**Библиотека** – уже скомпилированный файл формата ELF, в которой находятся символы с определением.

Посмотрим на файл до линковки. *readelf -a test.o*. Отличие в том, что Type теперь **REL – Relocatable file**. Причем в symtab можно также найти некоторые неопределенные переменные. Выглядят они примерно так:

\_ZSt3cin

\_Z – с этого начинаются все символы.

St – std::

3 – кол-во символов в названии переменной

cin – сама переменная

После линковки определение появляется.

Линковка бывает статическая и динамическая. .dynsym – для динамической.

**objdump** – утилита, позволяющая читать ELF файлы. может быть использован для разбора заголовков, дизассемблирования исполняемого файла. Ставится пакетом binutils.

*objdump -t a.out* – отображает таблицу символов.

*objdump -d a.out* – дизассемблирование (по бинарному файлу выводит ассемблерный)

*objdump -g* – показывает дебажную информацию

Если посмотреть на файл до линковки, на местах вызовов функций вместо адресов стоят заглушки.

**ldd** – позволяет посмотреть, от каких динамических либ зависит исполняемый файл.

*readelf -h libstdc++.so.6* – смотрим на либу

Тип увидим DYN (Shared object file). Означает, что это динамическая либа.

**ld**– стандартный линкер