Advanced Programming in the UNIX Environment

Week 05, Segment 5: Unix Development Tools: The Compiler Chain, Part III

Department of Computer Science
Stevens Institute of Technology

Jan Schaumann
jschauma@stevens.edu
https://stevens.netmeister.org/631/
The GNU Compiler Collection

A compiler translates *source code* from a high-level programming language into *machine code* for a given architecture by performing a number of steps:

- preprocessing
- lexical analysis
- syntax analysis
- semantic analysis
- code generation
- code optimization
- assembly
- linking

`cpp(1)`

`cc(1)`

`as(1)`

`ld(1)`
output is in the form of an assembler code file for each non-assembler input file specified.

By default, the assembler file name for a source file is made by replacing the suffix `.c`, `.i`, etc., with `.s`.

Input files that don't require compilation are ignored.

-E Stop after the preprocessing stage; do not run the compiler proper. The output is in the form of preprocessed source code, which is sent to the standard output.

```
apue$ cc -c hello.s
apue$ ls -l hello.*
-rwx------ 1 jschauma users 1053 Sep 10 00:53 hello.c
-rw-r----- 1 jschauma users 14129 Sep 12 00:26 hello.i
-rw-r----- 1 jschauma users 1928 Sep 12 00:31 hello.o
-rw-r----- 1 jschauma users  839 Sep 12 00:28 hello.s
apue$ ./hello.o
-sh: ./hello.o: permission denied
apue$ chmod +x hello.o
apue$ ./hello.o
-sh: Cannot execute ELF binary ./hello.o
apue$ ```
The GNU Compiler Collection

The compiler chain or driver usually performs preprocessing (e.g. via cpp(1)), compilation (cc(1)), assembly (as(1)) and linking (ld(1)).

• use cc(1) with the “-S” flag to stop after compilation
• use the “-OX” flags to set optimization
• assemble intermediate code to machine-dependent object files using as(1) or “cc -c”

To be continued...