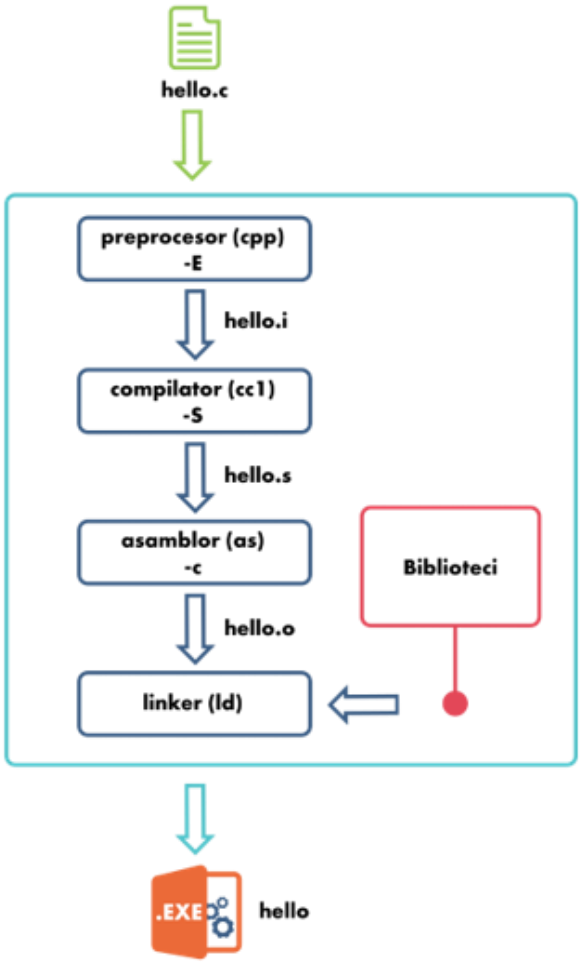
Linking

Modificat: 22-Oct-23

Suport curs

- https://github.com/systems-cs-pub-ro/iocla/tree/lecture-linking/curs/curs-18-19-linking
  - text: prezentare, explicații, demo-uri
  - fișiere de suport demo-uri

De la cod sursă la executabil



De ce să mă preocupe fazele compilării?

- Ești inginer, în pana mea!
- Sistemele / programele / procedurile dau greș
  - probleme de compilare
  - probleme de asamblare
  - probleme de linking
- Pentru testare pe configurații diferite
- Pentru analiză de performanță, securitate

De ce să știu despre linking?

- Se întâmplă oricând generăm un executabil
- Problemele de linking sunt frecvente și puțin înțelese
- În general, te vei folosi de componente software existente
  - fie firect la nivelul codului (API)
  - fie printr-o interfață binără specifică (ABI)
  - fie prin linkarea lor

Probleme de linking

- O funcție este definită static și este apelată din alt modul (compilation unit)
- O funcție este este definită în două module
- Nu se poate localiza biblioteca dorită
- Uneori probleme de linking se manifestă doar la încărcare sau la rulare
  - o funcție este folosită cu altă signatură
  - se folosește o altă bibliotecă dinamică

Momente din procesul de build / run

- compile time
- link-time
- load-time
- run-time

Linking

- module obiect + biblioteci legate / linkate
- rezultă fișier executabil (su biblioteci dinamice)

- fișierul executabil poate fi încărcat (loaded)

- încărcare = crearea unui proces din codul și datele executabilului

Fișiere folosite

- Cod sursă (editabil de programator), text
  - numit și compilation unit / translation unit
- Fișier cod obiect, modul obiect
  - obținut din compilation unit
- Bibliotecă
  - din unul sau mai multe fișiere obiect
  - colecție
- Executabil
  - din unul sau mai multe module obiect
  - selecție
  - unele module obiect pot fi în biblioteci

Module obiect vs fișiere executabile

- ambele: cod mașină (poate fi dezasamblat)
- ambele: format de fișier comun, analizabile cu utilitare separate

- doar fișierul executabil
  - entry point (de unde începe execuția), poate fi încărcat
  - adrese efective ale programului

- doar fișierul obiect
  - simboluri (variabile, funcții) nedefinite
  - sunt definite în alte module

Ce conține un fișier executabil / obiect?

- header
- secțiuni
  - .text (cod)
  - .data (date)
  - .debug (depanare)
  - .symtab / .strtab / .shstrtab (simboluri)

- secțiunile de cod pot fi dezasamblate

Responsabilitățile linkerului

- rezolvarea simbolurilor (symbol resolution)
- unificarea secțiunilor
- stabilirea adreselor (address binding)
- relocarea simbolurilor (relocation)
- stabilirea entry pointului

Stabilirea entry pointului

- adresa primei instrucțiuni executate la rulare
- localizat în header
  - readelf -h
- nu este main
- este, uzual, _start
  - _start pregătește mediul de lucru (stivă, argumentele programului)
  - apoi apelează main

Simboluri

- Variabile globale, funcții, variabile locale statice
- Au un nume, o adresă, un conținut, o dimensiune
- Funcțiile se găsesc în secțiunea .text
- Variabilele se găsesc în .data, .bss, .rodata
- Definiții de variabile – simboluri definite
  - int var = 10;
  - int func(void) {…}
- Declarații și folosiri de variabile / funcții - simboluri nedefinite;
  - extern int var;
  - int func(void);

Rezolvarea simbolurilor

- în fiecare fișier obiect se parcurg simbolurile nedefinite
  - se localizează în celelalte fișiere obiect
- dacă nu se găsesc: eroare de linking
- investigare folosind utilitarul nm sau readelf -s

Secțiuni

- compartimente ale fișierelor obiect / executabile
- conțin tipuri comune de informații: cod, date inițializate, date neinițializate
- nu au adrese stabilite în fișiere obiect
- au adrese stabilite în fișierul executabil

Unificarea secțiunilor + Stabilirea adreselor

- secțiunile de același tip sunt puse una după alta în executabil
- fiecare secțiune primește o adresă
- secțiunile unificate sunt puse consecutiv în executabil
- apoi fiecare simbol din secțiunile unificate primește o adresă finală
- readelf -S

Relocarea simbolurilor

- acolo unde simbolurile nedefinite sunt folosite se plasează placeholdere
  - putem folosi objdump pentru dezasamblarea fișierului obiect
- după stabilirea adreselor aceste placeholdere sunt completate: relocare
- fișierul executabil are placeholderele completate
- fișierele obiect se mai numesc "relocabile"
  - readelf -r

Stripping

- executabilele nu au cu adevărat nevoie de simboluri (de nume)
- simbolurile pot fi scoase (stripped) din executabile
- mare parte din executabilele finale sunt stripped

Demo time

- 01-one-file
- 02-two-files
- 03-reloc

Biblioteci

- colecții de fișiere obiect
- extensia .a (arhivă - fișiere puse unul după altul, necomprimate)
  - ar rc libtest.a a.o b.o c.o
- putem folosi nm, objdump
- pentru linkare
  - -ltest
  - -L. (directorul curent sau cel în care se află biblioteca)

Biblioteca standard C

- standard C library
- libc
- oversiune de libc pe orice sistem
- linkată implicit
- funcționalități de bază pentru programe
  - lucru cu memoria
  - lucru cu șiruri
  - lucru cu fișiere I/O
  - comunicare inter-proces (IPC)

Executabile cu linkare statică

- codul și datele din modulele obiect folosite din cadrul bibliotecilor sunt în executabil
- executabile complete
  - pot rula pe orice configurație
  - dimensiune mare
- bibliotecile sunt numite statice

Demo

- 04-lib/
- 05-static/

Executabile cu linkare dinamică

- codul și datele din biblioteci nu mai sunt plasate în executabil
  - sunt referințe
  - rezolvate la load-time de loader / dynamic linker
- bibliotecile se numesc dinamice / partajate
  - .dll pe Windows, .dylib pe macOS, .so pe Linux
- executabile mai mici
- codul bibliotecilor este partajate între procese la load-time / run-time

Comenzi pentru/cu biblioteci dinamice

- ldd /bin/ls
- gcc -shared -o libtest.so a.o b.o c.o
- gcc –L. -o main main.o -ltest
- LD_LIBRARY_PATH=. ./test

Demo

- 06-dynamic/
- 07-dynlib/

Sumar

- linking: fișiere obiect -> fișier executabil
- unificare secțiuni, stabilire adrese, rezolvare / relocare simboluri, stabilire entry point
- header, secțiuni, simboluri
- nm, objdump, readelf
- linkare statică, biblioteci statice
- linkare dinamică, biblioteci dinamice

Intrebari?