Prof. Sergei Gorlatch

Übungen zu Betriebssysteme

Blatt 6 (Dynamisches Laden zur Laufzeit)

Aufgabe 10 (Dynamisches Laden mit `libdl`)

12 Punkte

Machen Sie sich mit dem vorgegebenen Programm `reduce` vertraut. Dieses besteht aus den Dateien `loader.c`, `loader.h`, `load_op.c`, `unload_op.c` und `reduce.c`.

Implementieren Sie die Funktion `reduce` und die nötigen Funktionen in `load_op.c` und `unload_op.c` um das dynamische Laden von Operationen für `reduce` zu realisieren. Benutzen Sie dazu die Funktionen `dlsym`, `dlopen` und `dlclose` aus `libdl`, die das Laden von Symbolen und das Öffnen und Schließen einer dynamischen Bibliothek zur Laufzeit umsetzen. (`dlopen(3)`)

Gehen Sie bei der Implementierung schrittweise vor:

a) Implementieren Sie die fehlenden Funktionen aus `loader.h` in `load_op.c`, `unload_op.c` und `reduce.c`.

- `load_op(struct dynop *dest, const char* filename)`: Diese Funktion soll eine dynamische Bilbiothek mit dem Namen `filename` öffnen und in dieser Bibliothek ein Symbol `op` suchen. Dieses ist vom Typ `struct opfn`. Initialisieren Sie dann die Struktur auf die `dest` zeigt.
- `unload_op(struct dynop *op)`: Diese Funktion soll die geöffnete Bibliothek aus `op` schließen und die Struktur `dynop` danach aufräumen.
- `reduce(int *dest, const int *values, size_t count, int op(int a, int b))`: Diese Funktion bekommt ein Array mit `count` Ganzzahlen und soll es unter Anwendung der übergebenen Funktion `op` auf einen Wert reduzieren. Dieser soll nach `dest` geschrieben werden.

Hinweis: Die Dokumentation in `loader.h` könnten hilfreich sein.

b) Vervollständigen Sie die Funktion `main` in `loader.c`. Diese soll mit Hilfe von `load_op` die Operation aus einer dynamischen Bibliothek laden. Der Name der zu öffnenden Bibliothek wird als erstes Kommandzeilenargument übergeben. Nach dem erfolreichen Laden der Operation soll die von Ihnen implementierte `reduce`-Funktion mit der geladenen Reduzierungsmethode aufgerufen werden. Vergessen Sie nicht, die geöffnete Bibliothek auch wieder zu schließen!

c) Testen sie das Laden der Operationen aus den mitgelieferten Bibliotheken.

d) Implementieren Sie ihre eigene dynamische Bibliothek, die eine Reduzierungsmethode bietet in einer eigenen `.c`-Datei. Damit diese in eine dynamische Bibliothek kompiliert wird, muss der Bibliotheksname zu der Variable `LIBS` im `Makefile` hinzugefügt werden. Beachten Sie dabei die Benennung: Bei `xyz.c` sollte `lib/xyz.so` zu der Variable hinzugefügt werden, damit die automatische Ersetzung im `Makefile` funkioniert!

e) Warum wird `struct opfn` zum Laden einer Funktion aus einer Bibliothek genutzt, anstatt die Funktion direkt zu laden? Ziehen Sie dazu die `manpage` von `dlopen` (`dlopen(3)`) zu Rate!

Überprüfen Sie die Rückgabewerte aller Systemaufrufe, Bibliotheksfunktionen und ihrer eigenen Funktionen auf Fehler! Überprüfen Sie gegebenenfalls Funktionsparameter auf Validität! Geben Sie eventuelle Fehler mit `perror` aus.

Das bereitgestelle Makefile kann verwendet werden um das Programm auf den Rechnern des CIP-Pools in der Informatik zu kompilieren.

Hinweise zur Abgabe:

Das Übungsblatt muss bis zum **24.1.2023, 12:00 Uhr** abgegeben werden.
Halten Sie sich strikt an die Vorgaben im LearnWeb: *siehe hier.* Nichteinhalten der Vorgaben führt automatisch zu Punktabzug!

Die Bearbeitung muss in Gruppen von 3 oder 4 Teilnehmern erfolgen.
Fragen können in der Übung oder im LearnWeb geklärt werden. Abgabe per E-Mail an den jeweiligen Tutor der entsprechenden Übungsgruppe mit Subject „`Abgabe Uebung 6`“. Textaufgaben müssen als PDF-Datei abgegeben werden.

Bei der E-Mail Abgabe bitte nur **eine einzige** .zip oder .tgz oder .tbz Datei abgeben!

Die Abgaben sollen in den entspechenden Vorgabedateien implementiert werden. Die Abgaben müssen sich auf einem Linux-System der IVV5 mit dem bereitgestellten Makefile übersetzen lassen.

Wichtig: Bei der Abgabe in der E-Mail *alle* Namen und Matrikelnummern angeben.
Pro fehlender Angabe (Name oder Matrikelnummer) kann ein Punkt abgezogen werden!