Programmation assembleur (ASM)

Daniel Rossier / Magali Frölich

Exercices de programmation système en C

14 septembre 2015
Echéance: 27 septembre 2015

Objectif du laboratoire

L'objectif de ce laboratoire est de vérifier les acquis en programmation système en C avec des manipulations simples. On profite de ce premier laboratoire pour faire connaissance avec l'environnement de développement et les différents fichiers de base qui seront utilisés par la suite.

Le gestionnaire de version utilisé dans le cadre de ces laboratoires est ***git***. Vous trouverez de la documentation utile sur le site du cours (*http://www.reds.ch/fr/Formations/Bachelor/ASM*).

Consignes de laboratoire

- Ce laboratoire est individuel.
- Le laboratoire est évalué selon les conditions établies dans le document annexe qui vous a été distribué en début de semestre.

Etape no. 1 - Mise en place de l'environnement, utilisation de *git*

L'environnement de développement utilisé dans ce cours est l'IDE ***Eclipse***. Un *workspace Eclipse* a été préconfiguré dans le dépôt *git* (fichiers dans le répertoire caché `.metadata/` à la racine).

a) Il faut tout d'abord récupérer le dépôt étudiant pour les laboratoires ASM à l'aide de la commande suivante (via une fenêtre de *terminal*):

```
$ git clone firstname.lastname@eigit.heig-vd.ch:/home2/reds/asm/asm_student
```

Le nom du *workspace* est *asm_student* et correspond au répertoire cloné.

b) Lancez *eclipse* et ouvrez le *workspace asm_student*. Vous devriez obtenir la liste des projets (à gauche). Chaque projet a un lien symbolique dans la racine du *workspace*.

Etape no. 2 - Manipulation de pointeurs

Le projet à utiliser est le projet ***c***

a) Ouvrez le fichier ***exo1.c*** et complétez la fonction *copymem()* - sans utiliser la fonction *memcpy()* ou *strcpy()* - qui doit copier un bloc mémoire de taille quelconque. Pour compiler votre application, il faut utiliser un *shell* et lancer la commande *make*.

b) Lancez l'application à partir de l'image *ELF* (extension *.elf*)

c) Tester l'application avec une chaîne de caractères. Affichez le résultat de la copie.

Etape no. 3 - Manipulation d'opérateurs logiques

Cette étape consiste à code une petite application C qui effectue une inversion du format d'encodage d'un entier 32 bits, de *little-endian* vers *big-endian*.

a) Ouvrez le fichier ***exo2.c*** et implémentez une fonction qui convertit un nombre entier 32 bits codé en *little-endian* (par défaut) vers *big-endian*.

b) Testez votre application en affichant le résultat d'une conversion.

Etape no. 4 - Fichiers binaires et *debugger*

Dans cette étape, on s'intéresse à analyser le comportement d'un programme.

a) Examinez les fichiers produits par la commande *make*

⇒ Tentez d'expliquer les différences entre les fichiers *.elf* et *.bin*. Ouvrez le fichier *Makefile* associé au projet et essayez de trouver la commande générant le fichier *.bin*

Pour cela, vous pouvez vous aider des utilitaires *objdump* et *readelf*, selon l'exemple suivant:

```
$ objdump -d exo1.elf
$ readelf -h exo1.elf
```

b) Dans *eclipse*, démarrez le *debugger* en utilisant la configuration "*C (native)*" prédéfinie pour l'application de l'étape 2 (*exo1*). Insérez un *breakpoint* dans le code source afin de visualiser les instructions désassemblées en cours d'exécution. Avancez en pas-à-pas (touche *F5*) et observez les modifications des registres.