Tutoriel : utilisation et installation de BLCR

Camille Coti

1er octobre 2016

1 Qu'est-ce que BLCR

BLCR (Berkeley Lab Checkpoint/Restart) est une bibliothèque s'appuyant sur un module noyau permettant de sauvegarder l'état d'un processus à un moment donné de son exécution. Typiquement, on s'en sert pour :

- Migrer des processus en cours d'exécution : par exemple si on se rend compte que la machine ne dispose pas d'assez de mémoire, on veut migrer sur une machine qui en a plus.
- Sauvegarder périodiquement l'état d'un processus pour ne pas avoir à reprendre le calcul à zéro en cas de panne de la machine.

Le site web de BLCR se trouve ici : `https://ftg.lbl.gov/projects/CheckpointRestart`.

2 Utilisation de BLCR

BLCR dump l'état d'un processus (espace mémoire, registres, program counter, etc) dans un fichier. On lui fournit au minimum le pid de l'application à checkpointer, et il met tout dans un fichier context.¡pid¿. La documentation utilisateur officielle se trouve à l'adresse suivante : `https://upc-bugs.lbl.gov//blcr/doc/html/BLCR_Users_Guide.html`.

2.1 Précautions d'usage

Le module noyau `blcr` doit être chargé :

```
1 $ lsmod | grep blcr
2 blcr             77523     0
3 blcr_imports      2065     1   blcr
```

Les exécutables de blcr (`cr_checkpoint`, `cr_run` et `cr_restart`) doivent être dans votre `PATH` et les bibliothèques (`libcr.so`, `libcr_run.so` et `libcr_omit.so`) doivent être dans votre `LD_LIBRARY_PATH`.

2.2 Exécution de l'application

Le processus doit être *checkpoitable*. Pour celà, deux façons de rendre votre processus checkpointable :

- Compiler l'application en linkant avec la librairie `libcr_run`

```
1 $ gcc -o appli appli.c -lcr_run
```

- ou l'exécuter dans le wrapper `cr_run`

```
1 $ cr_run appli
```

2.3 Checkpointer un processus

Pour checkpointer un processus, on utilise `cr_checkpoint` en lui passant le pid du processus concerné. Autre option intéressante : on peut spécifier le nom du fichier dans lequel on va dumper le checkpoint. D'autres options sont disponibles, utilisez `cr_checkpoint --help` pour plus d'informations.

```
$ cr_checkpoint --pid 4321 --file moncheckpoint
```

Le checkpoint du processus est dumpé dans le fichier moncheckpoint. C'est ce fichier qui contient l'état du processus : on peut relancer le processus à partir de ce fichier.

2.4 Relancer un processus checkpointé

On a besoin du fichier contenant le checkpoint du processus. On utilise `cr_restart` pour le relancer en lui passant le chemin vers le fichier du checkpoint. Des options sont disponibles, utilisez `cr_restart --help` pour plus d'informations.

```
$ cr_restart moncheckpoint
```

Attention au pid : par défaut, le processus est relancé avec le même pid. Comme on ne peut pas avoir deux processus avec le même pid tournant sur la même machine, le processus initial doit être arrêté. On peut demander à `cr_restart` de relancer avec un nouveau pid en utilisant l'option `--no-restore-pid`.

2.5 Petit exemple

Pour observer l'utilisation de blcr on peut utiliser le code suivant. Il affiche la valeur d'un compteur et incrémente sa valeur toutes les 2 secondes.

```
#include <stdio.h>
#include <stdlib.h>

int main(){
    int i;
    i = 0;
    while( i < 100 ) {
        printf( "i = %d\n", i );
        sleep( 2 );
        i++;
    }
    return EXIT_SUCCESS;
}
```

La première possibilité d'utilisation consiste à compiler normalement l'application et à l'exécuter dans `cr_run`.

```
$ gcc -o test_blcr test_blcr.c
$ ./test_blcr
i = 0
i = 1
i = 2
i = 3
i = 4
....
```

La deuxième consiste à compiler en linkant avec la bibliothèque `libcr_run` et à exécuter l'application normalement.

```
$ gcc -o test_blcr test_blcr.c -lcr_run
$ cr_run ./test_blcr
i = 0
i = 1
i = 2
i = 3
i = 4
....
```

Avec `ps ux | grep test_blcr` on repère le pid de notre processus `test_blcr` et on le checkpoint :

```
$ ps ux | grep test_blcr
coti    7679   0.0   0.0    1564   388 pts/13   S+   14:01   0:00   ./test_blcr
$ cr_checkpoint --pid 7679
```

On a alors un nouveau fichier `context.7679` contenant le checkpoint. On peut alors redémarrer l'application dans l'état dans lequel elle était au moment du checkpoint.

```
$ cr_restart context.7679
i = 7
i = 8
i = 9
i = 10
...
```

3 Installation de BLCR

3.1 Pré-requis

BLCR est une bibliothèque et un module noyau Linux. Il faut donc dispose des sources du noyau qui tourne sur la machine. En particulier il a été testé pour des noyaux allant de 2.6.0 à 3.7.1.

3.2 Téléchargement

La dernière version (0.8.5) se télécharge sur le siteLa documentation administrateur officielle se trouve à l'adresse suivante : `https://upc-bugs.lbl.gov//blcr/doc/html/BLCR_Admin_Guide.html`.

3.3 Configuration et compilation

Des versons précédentes de BLCR (0.8.2) nécessitaient de patcher les sources du noyau pour certaines versions de celui-ci. Ce n'est désormais plus le cas. Sa compilation se fait de façon classique :

```
$ ./configure --enable-shared
$ make -j 3 > /dev/null
'depmod' check PASSED
# make install > /dev/null
'depmod' check PASSED
# make insmod > /dev/null
'depmod' check PASSED
```

3.4 Vérification

Par défaut, si vous ne passez pas d'option `--prefix` au script de configuration, blcr s'installe dans /usr/local. Vous pouvez donc vérifier que les exécutables et les bibliothèques s'y trouvent bien. Vous pouvez aussi vérifier que la dernière étape d'installation a bien chargé les modules noyau.

```
$ lsmod | grep blcr
blcr            77523     0
blcr_imports     2065     1   blcr
```