T2K-Tsukuba 上で危険なことをしてしまって それに対応した話

杉崎 行優*

平成 26 年 2 月 2 日

概 要

以前、T2K-Tsukuba 上でプロセスを過剰に起動したため、それ以上プロセスを `fork` することが不可能な状態になってしまった。

本報告書では、その状態に至った経緯と解決までの手順を述べる。

目 次

*並木中等教育学校 4 年次

1　プログラムの作成

2、3ヶ月程前、僕は MPI を用いて並列に hello world を出力するプログラムを美しくしようと考えていた。初めは、通信をせずただ単に同時に hello world を出力するものであったが、最終的に、リレー方式で “hello world” という文字列を回していき、最後に受け取った者がそれを出力するというものにした。全員が勢いをつけながらも協調する、とても美しいプログラムだ。

2　プログラムの実行と問題発生

そのプログラムを 1 台のマシン上で 16 プロセスとして実行したところ、正常に実行終了した。そこで僕は、そのプログラムをできるだけ多くのプロセス数で実行したいと考えた。そして、1 台のマシン上で 300 プロセスとして実行することにし、下のコマンドを実行した。

```
$ mpiexec -n 300 ./mpihello
```

僕はわくわくしながら hello world が出力されるのを待った。

しばらく経って、僕はプログラムの最後でメモリを開放していないことに気がついた。このままでは美しくないと思い、Control キーと C キーを押してプログラムの中断を試みた。しかしなぜかプログラムは終了しない。僕は別に開かれていた SSH セッションから下のコマンドを実行した。

```
$ ps U $USER
```

すると予期しない出力が返ってきた。

```
fork:  resource temporary unavilable
```

これはつまり、プロセステーブルがいっぱいになってしまい、これ以上新しいプロセスを実行することができないということを表している。

3　原因の究明

しかし、通常のシステムではプロセステーブルの大きさは 32767 程に設定されていて、300 プロセスを実行した位では全く影響がないはずだ。しかし、僕には思い当たりがあり、下のコマンドを実行した。

```
$ ulimit -a
```

このコマンドを実行すると、ユーザのリソース制限の値が表示される。出力は以下である。

```
core file size          (blocks, -c) 0
data seg size           (kbytes, -d) unlimited
scheduling priority             (-e) 0
file size               (blocks, -f) unlimited
pending signals                 (-i) 257221
max locked memory       (kbytes, -l) unlimited
max memory size         (kbytes, -m) 20971520
open files                      (-n) 1024
pipe size            (512 bytes, -p) 8
POSIX message queues     (bytes, -q) 819200
real-time priority              (-r) 0
stack size              (kbytes, -s) 10240
cpu time               (seconds, -t) unlimited
max user processes              (-u) 200
virtual memory          (kbytes, -v) unlimited
file locks                      (-x) unlimited
```

この中の

```
max user processes              (-u) 200
```

はユーザが実行できる最大のプロセス数が 200 であることを示している。僕が実行した 300 のプロセスはこの最大数を余裕で上回っている。そして、現在 hello world のプロセスが全ての hello world のプロセスの起動を待っており、デッドロックしている状態なのだ。

4 問題の解決

僕は管理者に頼むか自分で解決するか悩んだ。最終的に、前者の場合では最悪システム使用停止命令を出されてしまうので、自分で解決することを試み、どうしても無理な場合は前者に頼ることにした。

新たなプロセスを起動できないので、プログラムを実行するにはシェル上で exec するかシェルの built-in コマンドを実行する必要がある。

SSH セッションは 2 つ開かれている。まず僕は 1 つの SSH セッションを犠牲に下のコマンドを実行した。

```
$ exec ps U $USER
```

以下が出力された。

```
  PID TTY      STAT   TIME COMMAND
 7638 pts/2    S      0:00 ./mpihello
 7639 pts/2    S      0:00 ./mpihello
 7640 pts/2    S      0:00 ./mpihello
 7641 pts/2    S      0:00 ./mpihello
   (以下略)
```

そして、シェル上で以下の関数を定義し、/proc にある各プロセスの情報が記されたファイルを走査した。

```
$ static_cat(){
    S="notblank"
    while test "$S"; do
       read S
       echo $S
    done
 }
```

すると、/proc/$PID/cmdline(ここで$PID はプロセス ID を表す) にプロセス名が記されていることが分かった。そこで、以下のスクリプトを実行し、./mpihello と名のついたプロセスを全て終了させた。

```
$ for f in /proc/*/cmdline; do
    read NAME <$f
    if test "$NAME" == "./mpihello"; then
       tmp1=${NAME%/*}
       tmp2=${tmp1#*/}
       TARGET_PID=${tmp2#*/}
       echo Killing PID $TARGET_PID
       kill -KILL $TARGET_PID
    fi
 done
```

これで全 hello world のプロセスが終了したので、問題が解決した。