

12/27/17

Problem 1 – using strace

Not sure why, might be a windows locale thingy, but all `\` symbols print out to be the Korean currency symbol ₩ for some reason.

```
dhpark@HAL-9001: /mnt/d/Dropbox/School/Copper Onion/3.Juneur/Operating Systems/Homework/Assignment 7 Assembly/src
```

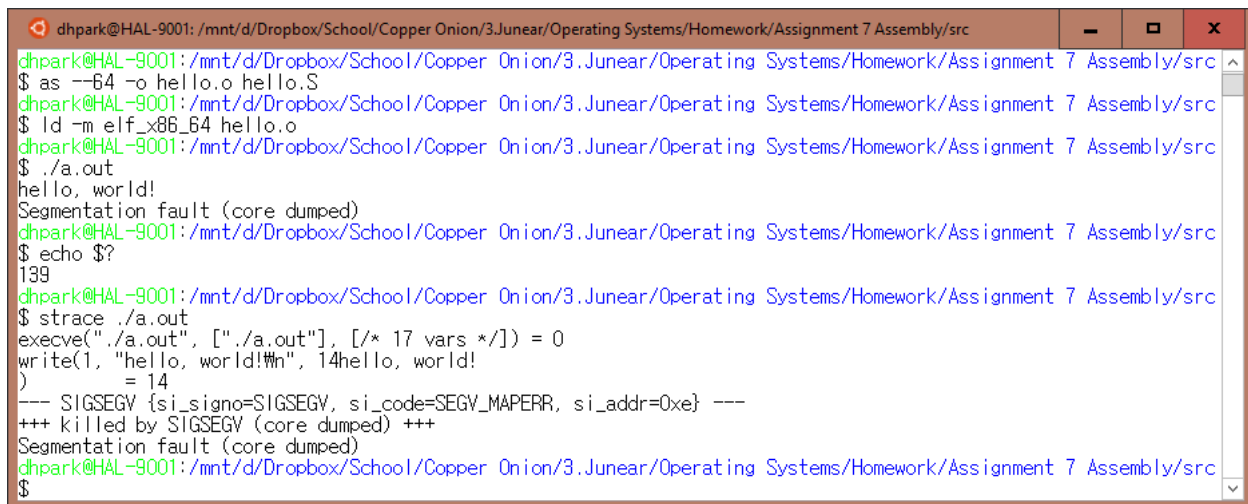
```
$ gcc -o hello.exe hello.c  
dhpark@HAL-9001: /mnt/d/Dropbox/School/Copper Onion/3.Juneur/Operating Systems/Homework/Assignment 7 Assembly/src  
$ strace ./hello.exe  
execve("./hello.exe", ["/hello.exe"], [/ * 17 vars */]) = 0  
brk(NULL) = 0x6db000  
access("/etc/ld.so.nohwcap", F_OK) = -1 ENOENT (No such file or directory)  
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f4e1c350000  
access("/etc/ld.so.preload", R_OK) = -1 ENOENT (No such file or directory)  
open("/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3  
fstat(3, {st_mode=S_IFREG|0644, st_size=24473, ...}) = 0  
mmap(NULL, 24473, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f4e1c357000  
close(3) = 0  
access("/etc/ld.so.nohwcap", F_OK) = -1 ENOENT (No such file or directory)  
open("/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3  
read(3, "\217\251\252\251\253\254\255\256\257\258\259\260\261\262\263\264\265\266\267\268\269\270\271\272\273\274\275\276\277\278\279\280\281\282\283\284\285\286\287\288\289\290\291\292\293\294\295\296\297\298\299\300\301\302\303\304\305\306\307\308\309\310\311\312\313\314\315\316\317\318\319\320\321\322\323\324\325\326\327\328\329\330\331\332\333\334\335\336\337\338\339\340\341\342\343\344\345\346\347\348\349\350\351\352\353\354\355\356\357\358\359\360\361\362\363\364\365\366\367\368\369\370\371\372\373\374\375\376\377\378\379\380\381\382\383\384\385\386\387\388\389\390\391\392\393\394\395\396\397\398\399\400\401\402\403\404\405\406\407\408\409\410\411\412\413\414\415\416\417\418\419\420\421\422\423\424\425\426\427\428\429\430\431\432\433\434\435\436\437\438\439\440\441\442\443\444\445\446\447\448\449\450\451\452\453\454\455\456\457\458\459\460\461\462\463\464\465\466\467\468\469\470\471\472\473\474\475\476\477\478\479\480\481\482\483\484\485\486\487\488\489\490\491\492\493\494\495\496\497\498\499\500\501\502\503\504\505\506\507\508\509\510\511\512\513\514\515\516\517\518\519\520\521\522\523\524\525\526\527\528\529\530\531\532\533\534\535\536\537\538\539\540\541\542\543\544\545\546\547\548\549\550\551\552\553\554\555\556\557\558\559\560\561\562\563\564\565\566\567\568\569\570\571\572\573\574\575\576\577\578\579\580\581\582\583\584\585\586\587\588\589\590\591\592\593\594\595\596\597\598\599\600\601\602\603\604\605\606\607\608\609\610\611\612\613\614\615\616\617\618\619\620\621\622\623\624\625\626\627\628\629\630\631\632\633\634\635\636\637\638\639\640\641\642\643\644\645\646\647\648\649\650\651\652\653\654\655\656\657\658\659\660\661\662\663\664\665\666\667\668\669\670\671\672\673\674\675\676\677\678\679\680\681\682\683\684\685\686\687\688\689\690\691\692\693\694\695\696\697\698\699\700\701\702\703\704\705\706\707\708\709\710\711\712\713\714\715\716\717\718\719\720\721\722\723\724\725\726\727\728\729\730\731\732\733\734\735\736\737\738\739\740\741\742\743\744\745\746\747\748\749\750\751\752\753\754\755\756\757\758\759\760\761\762\763\764\765\766\767\768\769\770\771\772\773\774\775\776\777\778\779\780\781\782\783\784\785\786\787\788\789\790\791\792\793\794\795\796\797\798\799\800\801\802\803\804\805\806\807\808\809\810\811\812\813\814\815\816\817\818\819\820\821\822\823\824\825\826\827\828\829\830\831\832\833\834\835\836\837\838\839\840\841\842\843\844\845\846\847\848\849\850\851\852\853\854\855\856\857\858\859\860\861\862\863\864\865\866\867\868\869\870\871\872\873\874\875\876\877\878\879\880\881\882\883\884\885\886\887\888\889\890\891\892\893\894\895\896\897\898\899\900\901\902\903\904\905\906\907\908\909\910\911\912\913\914\915\916\917\918\919\920\921\922\923\924\925\926\927\928\929\930\931\932\933\934\935\936\937\938\939\940\941\942\943\944\945\946\947\948\949\950\951\952\953\954\955\956\957\958\959\960\961\962\963\964\965\966\967\968\969\970\971\972\973\974\975\976\977\978\979\980\981\982\983\984\985\986\987\988\989\990\991\992\993\994\995\996\997\998\999\1000", 832) = 832  
fstat(3, {st_mode=S_IFREG|0755, st_size=1868984, ...}) = 0  
mmap(NULL, 3971488, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f4e1bc30000  
mprotect(0x7f4e1bd00000, 2097152, PROT_NONE) = 0  
mmap(0x7f4e1bff0000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1c0000) = 0x7f4e1bff0000  
mmap(0x7f4e1bff6000, 14752, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f4e1bff6000  
close(3) = 0  
mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f4e1c340000  
mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f4e1c330000  
mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f4e1c320000  
arch_prctl(ARCH_SET_FS, 0x7f4e1c330700) = 0  
mprotect(0x7f4e1bff0000, 16384, PROT_READ) = 0  
mprotect(0x600000, 4096, PROT_READ) = 0  
mprotect(0x7f4e1c225000, 4096, PROT_READ) = 0  
munmap(0x7f4e1c357000, 24473) = 0  
fstat(1, {st_mode=S_IFCHR|0660, st_rdev=makedev(4, 1), ...}) = 0  
ioctl(1, TCGETS, {B38400 oposit isig icanon echo ...}) = 0  
brk(NULL) = 0x6db000  
brk(0x6fc000) = 0x6fc
```

Problem 2 – pure assembly

```
.text
.global _start

_start:
    mov $1,%rax    #write system call
    mov $1,%rdi    #output to fd 1(stdout)
    mov $msg,%rsi   #msg to output
    mov $14,%rdx   #length of msg
    syscall
    .data

msg:
    .ascii "hello, world!\n"
```



```
dhpark@HAL-9001: /mnt/d/Dropbox/School/Copper Onion/3.Juneur/Operating Systems/Homework/Assignment 7 Assembly/src
$ as -64 -o hello.o hello.S
dhpark@HAL-9001: /mnt/d/Dropbox/School/Copper Onion/3.Juneur/Operating Systems/Homework/Assignment 7 Assembly/src
$ ld -m elf_x86_64 hello.o
dhpark@HAL-9001: /mnt/d/Dropbox/School/Copper Onion/3.Juneur/Operating Systems/Homework/Assignment 7 Assembly/src
$ ./a.out
hello, world!
Segmentation fault (core dumped)
dhpark@HAL-9001: /mnt/d/Dropbox/School/Copper Onion/3.Juneur/Operating Systems/Homework/Assignment 7 Assembly/src
$ echo $?
139
dhpark@HAL-9001: /mnt/d/Dropbox/School/Copper Onion/3.Juneur/Operating Systems/Homework/Assignment 7 Assembly/src
$ strace ./a.out
execve("./a.out", ["/a.out"], [/* 17 vars */]) = 0
write(1, "hello, world!\n", 14)hello, world!
) = 14
--- SIGSEGV {si_signo=SIGSEGV, si_code=SEGV_MAPERR, si_addr=0xe} ---
+++ killed by SIGSEGV (core dumped) +++
Segmentation fault (core dumped)
dhpark@HAL-9001: /mnt/d/Dropbox/School/Copper Onion/3.Juneur/Operating Systems/Homework/Assignment 7 Assembly/src
$
```

Problem 3 – exit code

After the write system call, the program terminates with the exit value 139 as seen by the echo \$?, and the signal was SIGSEGV as seen by the strace. This makes sense because SIGSEGV is value 11, and the UNIX convention follows that the errno value of 128 added to 11 results in 139. The reason behind this error is due to the fact that the %rax register is volatile and thus not available at the program's termination.

```
.text
.global _start

_start:
    mov $1,%rax    #write system call
    mov $1,%rdi    #output to fd 1(stdout)
    mov $msg,%rsi   #msg to output
    mov $14,%rdx   #length of msg
```

```

syscall
mov $1,%rdi      #return code for exit syscall
mov $60,%rax     #exit system call
syscall
.data

```

```

msg:
.ascii "hello, world!\n"

```

```

dhpark@HAL-9001: /mnt/d/Dropbox/School/Copper Onion/3.Juneur/Operating Systems/Homework/Assignment 7 Assembly/src
dhpark@HAL-9001: /mnt/d/Dropbox/School/Copper Onion/3.Juneur/Operating Systems/Homework/Assignment 7 Assembly/src
$ as --64 -o hello.o hello.S
dhpark@HAL-9001: /mnt/d/Dropbox/School/Copper Onion/3.Juneur/Operating Systems/Homework/Assignment 7 Assembly/src
$ ld -m elf_x86_64 hello.o
dhpark@HAL-9001: /mnt/d/Dropbox/School/Copper Onion/3.Juneur/Operating Systems/Homework/Assignment 7 Assembly/src
$ ./a.out
hello, world!
dhpark@HAL-9001: /mnt/d/Dropbox/School/Copper Onion/3.Juneur/Operating Systems/Homework/Assignment 7 Assembly/src
$ echo $?
1
dhpark@HAL-9001: /mnt/d/Dropbox/School/Copper Onion/3.Juneur/Operating Systems/Homework/Assignment 7 Assembly/src
$ strace ./a.out
execve("./a.out", ["/a.out"], [/* 17 vars */]) = 0
write(1, "hello, world!\n", 14)hello, world!
) = 14
exit(1) = ?
+++ exited with 1 +++
dhpark@HAL-9001: /mnt/d/Dropbox/School/Copper Onion/3.Juneur/Operating Systems/Homework/Assignment 7 Assembly/src
$

```

Problem 4 – system call validation

1. Replaced write system call value with 100

```

dhpark@HAL-9001: /mnt/d/Dropbox/School/Copper Onion/3.Juneur/Operating Systems/Homework/Assignment 7 Assembly/src
$ as --64 -o hello.o hello.S
dhpark@HAL-9001: /mnt/d/Dropbox/School/Copper Onion/3.Juneur/Operating Systems/Homework/Assignment 7 Assembly/src
$ ld -m elf_x86_64 hello.o
dhpark@HAL-9001: /mnt/d/Dropbox/School/Copper Onion/3.Juneur/Operating Systems/Homework/Assignment 7 Assembly/src
$ strace ./a.out
execve("./a.out", ["/a.out"], [/* 17 vars */]) = 0
times(0x1) = -1 EFAULT (Bad address)
exit(1) = ?
+++ exited with 1 +++
dhpark@HAL-9001: /mnt/d/Dropbox/School/Copper Onion/3.Juneur/Operating Systems/Homework/Assignment 7 Assembly/src
$

```

2. Passed invalid address for write string

```

dhpark@HAL-9001: /mnt/d/Dropbox/School/Copper Onion/3.Juneur/Operating Systems/Homework/Assignment 7 Assembly/src
$ as --64 -o hello.o hello.S
dhpark@HAL-9001: /mnt/d/Dropbox/School/Copper Onion/3.Juneur/Operating Systems/Homework/Assignment 7 Assembly/src
$ ld -m elf_x86_64 hello.o
dhpark@HAL-9001: /mnt/d/Dropbox/School/Copper Onion/3.Juneur/Operating Systems/Homework/Assignment 7 Assembly/src
$ strace ./a.out
execve("./a.out", ["/a.out"], [/* 17 vars */]) = 0
write(1, NULL, 14) = -1 EFAULT (Bad address)
exit(1) = ?
+++ exited with 1 +++
dhpark@HAL-9001: /mnt/d/Dropbox/School/Copper Onion/3.Juneur/Operating Systems/Homework/Assignment 7 Assembly/src
$

```