Princeton University
COS 217: Introduction to Programming Systems
GDB Tutorial for Assembly Language Programs (Part 2)

Motivation

Suppose you are developing the assembly language swap function. Further suppose that the function assembles and links cleanly, but is executing incorrectly. How can you use gdb to debug the function?

Debugging the swap function is somewhat difficult because it uses the stack and an array. This is an appropriate sequence...

Building for gdb

To prepare to use gdb, build your program with the -g option:

```
% gcc -g -Wall -ansi -pedantic mysort.c quicksort.c partition.c swap.s -o mysort
```

Doing so places extra information into the mysort file that gdb uses.

Creating a Data File

Create a data file to be sorted. Let's use the file junk.txt as an example:

```
% cat junk.txt
one
two
three
four
```

Running gdb

Run gdb from within xemacs:

```
% xemacs
<Esc key> x gdb <Enter key> mysort <Enter key>
```

Setting Breakpoints

Set breakpoints at appropriate places. Breakpoints at the beginning of the main and swap functions would be appropriate.

```
(gdb) break main
(gdb) break swap
```

Running Your Program

Run the program, redirecting stdin to junk.txt.

```
(gdb) run < junk.txt
```

Continue past the breakpoint at the beginning of the main function.

```
(gdb) continue
```

At this point, execution is paused after the two-instruction prolog of the first call of the swap function.

Examining Memory

Use the print command to determine the contents of the EBP register:

```
(gdb) print/a $ebp
bfff8ad8
```

Thus we know the address of the base of the current stack frame. (In general, that address will be different each time you run the program.) Now use the x command repeatedly to examine swap's parameters.

```
(gdb) x/a 0xbfff8ad8
bfff8f18
(gdb) x/a 0xbfff8adc
080486af
(gdb) x/a 0xbfff8ae0
095b6048
(gdb) x/d 0xbfff8ae4
0
(gdb) x/d 0xbfff8ae8
3
(gdb) x/a 0x095b6048
095b6018
(gdb) x/a 0x095b604c
095b6028
(gdb) x/a 0x095b6050
095b6038
(gdb) x/a 0x095b6054
095b6008
(gdb) x/c 0x095b6018
o
(gdb) x/s 0x095b6018
one
(gdb) x/c 0x095b6028
t
(gdb) x/s 0x095b6028
two
(gdb) x/c 0x095b6038
t
(gdb) x/s 0x095b6038
three
(gdb) x/c 0x095b6008
f
(gdb) x/s 0x095b6008
four
```

As you traverse memory, draw a diagram of it as shown on the next page.