Linux Assembly

Armine Hayrapetyan

Outline

- Command Line Arguments

What are Command Line Arguments?

When a program is executed from the command line, arguments can be passed into it.

After the name of the program to execute, the arguments are separated by spaces.

All arguments are *strings*, not integers.

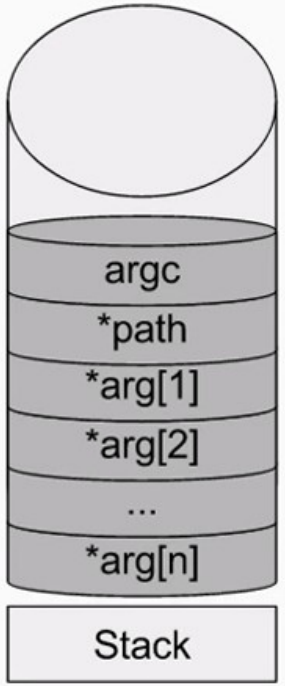
```
$ ./argtest argument1 argument2 argument3
```

Arguments on the Stack

When the program is executed, the arguments are automatically loaded onto the stack.

The top item is the number of arguments. This number is always at least 1.

The next items in the stack are pointers to the zero-terminated strings starting with the path then of each individual argument.

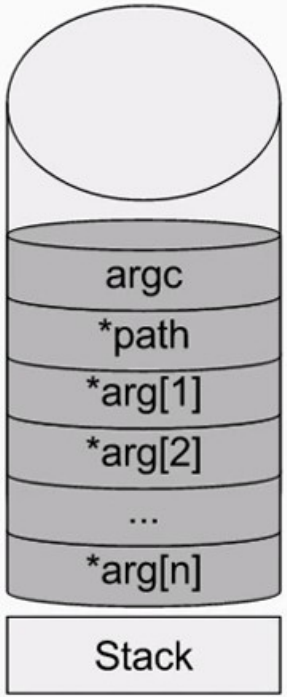


Arguments on the Stack

The path is technically the first argument, and arg[1] is technically the second argument, and arg[2] is technically the third argument, etc.

However, the operating system provides the *path, so *arg[1] is the first *user defined* argument.

This is why argc is 1 even if no arguments are specified.

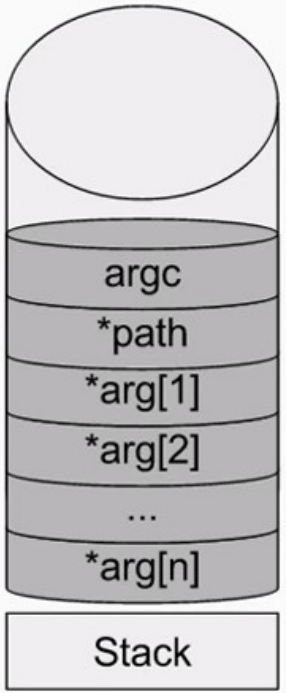


Arguments on the Stack

Example

```
$ ./argtest argument1 argument2 argument3
```

argc	4
*path	"./argtest"
*arg[1]	"argument1"
*arg[2]	"argument2"
*arg[3]	"argument3"



Displaying Argument Count

This code will display the argument count.

It pops the top item off the stack then prints it to the screen. This top item will always be the argument count.

“linux64.inc”

http://pastebin.com/wCNZs3RN

```
%include "linux64.inc"

section .data
        newline db 10,0

section .text
        global _start

_start:
        pop rax
        printVal rax
        print newline
        exit
```

argtest

Compile and run the program “argtest.asm” to get a feel for how arguments work.

This program will display an argument count followed by all the arguments inputted into the program.

```
$ ./argtest hello my name is johnny
Argument(s): 6
Argument #1: ./argtest
Argument #2: hello
Argument #3: my
Argument #4: name
Argument #5: is
Argument #6: johnny
```

http://pastebin.com/MAA0NFsK

Why Arguments?

Arguments allow the user to input data into a program without the program requiring a user interface.

This can also be used to link the output of one command line application to the input of your command line application.

Notice how the output file list from the “ls” command is fed into argtest.

```
$ ls
argtest  argtest.asm  argtest.o  linux64.inc
$ ./argtest `ls`
Argument(s): 5
Argument #1: ./argtest
Argument #2: argtest
Argument #3: argtest.asm
Argument #4: argtest.o
Argument #5: linux64.inc
```

References

- https://www.youtube.com/watch?v=xtFs1yBVinc&list=PLetF-YjXm-sCH6FrTz4AQhfH6INDQvQSn&index=9