# **Debugging and Profiling 2**

Lecture 14 March 2nd 2023 | COMP 211-002 | Joshua Bakita

### Welcome!

#### Today:

#### → More on I/O

- ▶ mmap()
- Performance profiling
- → Debugging review

Logistics:

- → 54% of the class has started on Assignment 3
- → Tomorrow is the last day to drop the class with a "W" grade.

Most terminals support Ctrl+w to delete the last word you typed, and Ctrl+u to delete the whole line.

## **Performance Profiling**

### Performance Profiling Last time... but with detail

time <prog> <args>

\time -v <prog> <args>

- → Good for quickly checking a program's runtime
- → Low precision (millisecond-scale at best)
- → Does not provide granular information about what is slow

perf record -F <freq> --call-graph
dwarf,2048 <prog> <args>

- → Replace < freq> with sample frequency in HZ
- → Creates the perf.data with profile data

Visualization:

→ Run perf report for an interactive viewer

#### **Performance Profiling**

### perf report

→ This is a profile of the fread() version of ./cat from Lecture 8

Wouldn't it be nice if we could visualize this?

Children	Solf	Comma	nd Shared Object	Symbol			
85.04%	0.00%	cat_	cat	[] start			
start	0.00/0	cut					
- libc	start mai	n (inl	ined)				
- 85.0	3% main		(incu)				
+ 5	1.94% G	I IO	fread (inlined)				
+ 2	+ 29.16% GI TO fwrite (inlined)						
0	.58% 0x55	 7ec9c1	.90d0				
85.04%	0.00%	cat	libc-2.31.so	[.] libc start_main (inlined)			
85.03%	2.73%	cat	cat	[.] main			
53.19%	0.00%	cat					
47.89%	0.00%	cat	Can use up/down	arrow keys to browse, and "+"			
44,05%	1.35%	cat		terminale) to expand a node			
42.51%	15.06%	cat	(or enter in some	terminals) to expand a node			
.37.16%	0.00%	cat	and look at how	much time it's children use.			
35.74%	0.00%	cat					
32.05%	0.00%	cat	l1bc-2.31.so	[.]G110_fwrite (inlined)			
23.99%	0.00%	cat	<del>libc 2.31.30</del>	<pre>[.] _IO_new_file_xsputn (inlined</pre>			
23.92%	0.07%	cat	[kernel.kallsvms]	[k] x64 svs read			
23.86%	0.27%	cat	% of samples when	re this function was at the ton			
22.88%	0.45%	cat	v or sumples when				
19.83%	0.04%	cat	of the call stack				
19.69%	0.34%	cat	≈				
19.35%	0.16%	cat	% of time on	ont in this function only			
19.27%	0.15%	cat	∞ or time sp				
18.93%	1.66%	cat	[kernel.kallsyms]	<pre>[k] generic_file_buffered_read</pre>			
17.90%	17.90%	cat	% of samples with	the function in their call stack			
16.47%	0.00%	cat	% of samples with				
16.222	0.00%	cat		≈			
	0.00%	cat	% of time spent i	n this function and children			
	0.33%	cat					
		cat	[kernel.kallsyms]	[k] copy_user_enhanced_fast_st			
		cat	libc-2.31.so	[.]strncasecmp_l_sse2			

### Performance Profiling Last time... but with detail

time <prog> <args>

\time -v <prog> <args>

- → Good for quickly checking a program's runtime
- → Low precision (millisecond-scale at best)
- → Does not provide granular information about what is slow

perf record -F <freq> --call-graph
dwarf,2048 <prog> <args>

- → Replace <freq> with sample frequency in HZ
- → Creates the perf.data with profile data

Visualization:

- → Run perf report for an interactive viewer
- → Run /playpen/FlameGraph/211gen.sh to generate a visualization of perf.data in graph.svg

# Visualizing Performance Profiles

FlameGraphs!

#### ./cat with fread() and cold page cache



#### ./cat with fread() and hot page cache



https://www.cs.unc.edu/~jbakita/teach/comp211-s23/l13/before\_flush.svg

https://www.cs.unc.edu/~jbakita/teach/comp211-s23/l13/after\_flush.svg

# How Does I/O Really Work?

A sampling from one of my research presentations...

# **Debugging Revisited**

Likely relevant to Assignment 3!

#### Key Commands—From Lecture 7

#### **Command Line**

valgrind prog	Run prog with valgrind		
gdb prog	Start the GNU Debugger on prog		
info thing	View detailed manual for thing		
xxd file	Print file as hexadecimal		
wget addr	Download file from addr		
rm file	Delete file		
cd dir	Move to dir		
cat file	Print contents of file		
cp fileA fileB	Copy fileA to fileB		

#### Vim Commands (Normal Mode)

dd	Delete current line
D	Delete from cursor to end-of-line
>>	Increase indent
<<	Decrease indent
0	Add line above cursor and enter insert mode
Ο	Add line below cursor and enter insert mode

For Your ~/	.vimrc	Config	File
-------------	--------	--------	------

set cindent

set nowrap

#### **Debugging Revisited**

#### Key <u>GNU Debugger</u> (GDB) Commands—From L7

Full command name		Shorthand	A	ccess fo g	the full GDB manual via db on the command line		
Control Flow	backtrace	bt	List all stack frames	s break <file>:<l> break <file>:<l> break <function></function></l></file></l></file>		Set a breakpoint with optional condition at	
	<pre>select <frame#></frame#></pre>	sel	Select a stack frame as your context				Brea
	next	n	Execute the next line from your context	<pre>preak <function> if       <condition></condition></function></pre>		a location or function	kpoi
	step		Execute one line	info breakpoints	i b	List all breakpoints	nts
	list		Print source code	delete <breakpoir< td=""><td>t d</td><td>Delete a breakpoint</td><td></td></breakpoir<>	t d	Delete a breakpoint	
	print <expr></expr>	р	Execute expression and print result (can modify data)	continue	С	Resume execution	_
Data	info locals		Print value of every	run <args></args>	r	Run local program	⊳
		i lo	local variable in your	quit	q	Exit GDB	dm
	x <addr></addr>	x	Print bytes at addr in	help <cmd></cmd>	h	Print quick reference for a command	<u> </u>
	whatis <expr></expr>	wha	Print type of expr	set history save		Save command history	14

### **Questions?**

See office hour calendar on the website for availability.

Assignment 3 due Tuesday!

Contact: Email: <u>hacker@unc.edu</u> Twitter: <u>@JJBakita</u> Web: <u>https://cs.unc.edu/~jbakita</u>

