Debugging programs on Linux:

An Overview of gdb, idb, Insure++, Valgrind, ccmalloc and mpatrol

Jarno Laitinen (Jarno.Laitinen@cern.ch)

Slides: `http://www.lut.fi/u/jtl7950/dbg_slides.pdf`

CERN
openlab for DataGrid applications

Debugging

- Problems occurs - especially with C/C++ pointers and memory handling (new, malloc/calloc, free, delete)
- Segmentation fault: exceeding memory area
- But where and when?!

Printf as debugging method

- Simple, no need for tools
- Needs recompiling, inserting lines (where?!) and removing them (in good case)
- Too much stdout is problematic (especially in loops etc.) to follow and slows execution
- Maybe problematic in multithread or process case (logfiles?)
- Often useful to print source code file and line number

```
printf("Thread.c: %d: numThreads: %d\n",__LINE__,numThreads);
```

Compiler options to enable debugging

- Usually recommended to reduce optimisation (to `-O1` or `-O0`)
- Option `-g` is needed to show source code by many debuggers (some might not want it)
- gcc: For gdb exists also `-ggdb`
- gcc: For `-g` and `-ggdb` can be set levels 1-3. e.g. ”Some debuggers support macro expansion when you use -g3”
- Warnings may help to spot the problem. It seems that icc produces more warnings by default than gcc.
- More gcc warning with `-Wall`. Also several others e.g. `-Wfloat-equal`

```
Reporter.c:723: warning: comparing floating point with == or != is unsa
```

Common functionality of debuggers

- Run & kill, break & continue the execution
- Breakpoints: to stop on certain line/function (may have condtion)
- Watchpoints: stop if certain variable read or written (variables can be seen on local stack)
- clear & delete, disable & enable temporarily
- Step (to next line) or next (through subroutine) to line (#lines can be given as a parameter)
- Show backtrace of calls when crashed
- Print value or type of an variable or structure
- Some may allow setting value and executing printf

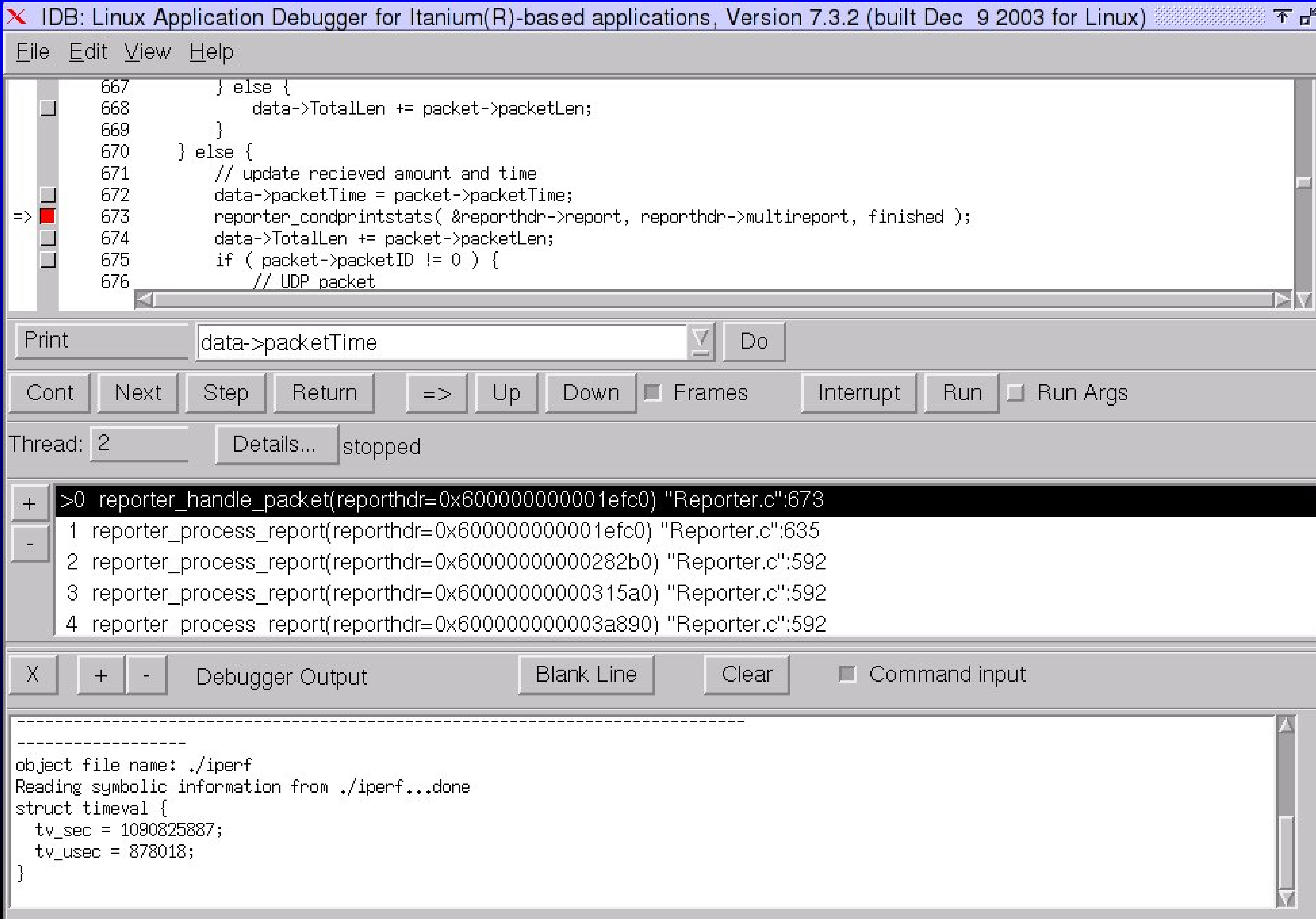
- Some may support (more or less better) POSIX threads (in gdb `info threads`, `thread` *`th_number`*
- Also running processes may be attached (`attach` *`process_id`* and `detach` in gdb)
- Special action need to debug child processes
  tutorial
- `info` (in gdb) about processes, threads, breakpoints..
- OpenMP (threads) and message passing libraries (MPI, PVM) exists specific debuggers

Usage of idb

- Documentation on Intel’s web site
- A bit different than gdb, can be set to `-gdb` mode (other mode dbx (Berkeley UNIX symbolic debugger)).
- Include source code files with `-I` ***`directory`***
- GUI (`-gui`) is worth of trying! ddd can use idb too.
- Exists at least in some oplapro (IA-64) machines
- Has limitations as well e.g. does not support new and delete C++ calls

IDB example

```
Thread received signal SEGV
stopped at [<opaque> __cfree(...) 0x200000000038b5a0]

Information:  An <opaque> type was presented during execution of the
previous command.  For complete type information on this symbol,
recompilation of the program will be necessary.  Consult the compiler
man pages for details on producing full symbol table information using
the ’-g’ (and ’-gall’ for cxx) flags.

(idb) where
>0  0x200000000038b5a0 in __cfree(...) in /lib/tls/libc.so.6.1
#1  0x2000000000151480 in /usr/lib/libstdc++.so.5
#2  0x2000000000151550 in /usr/lib/libstdc++.so.5
#3  0x400000000000bc90 in Settings_Destroy(mSettings=0x6000000000012ce0)
"Settings.cpp":262
#4  0x4000000000003f80 in thread_run_wrapper(paramPtr=<no value>)
"Thread.c":264
#5  0x2000000000288510 in start_thread(...) in /lib/tls/libpthread.so.0
```



```
        } else {
            data->TotalLen += packet->packetLen;
        }
    } else {
        // update recieved amount and time
        data->packetTime = packet->packetTime;
=>      reporter_condprintstats( &reporthdr->report, reporthdr->multireport, finished );
        data->TotalLen += packet->packetLen;
        if ( packet->packetID != 0 ) {
            // UDP packet
```

Print: data->packetTime

Thread: 2 stopped

```
>0  reporter_handle_packet(reporthdr=0x600000000001efc0) "Reporter.c":673
 1  reporter_process_report(reporthdr=0x600000000001efc0) "Reporter.c":635
 2  reporter_process_report(reporthdr=0x60000000000282b0) "Reporter.c":592
 3  reporter_process_report(reporthdr=0x60000000000315a0) "Reporter.c":592
 4  reporter_process_report(reporthdr=0x600000000003a890) "Reporter.c":592
```

```
------------------------------------------------------------------------------
------------------
object file name: ./iperf
Reading symbolic information from ./iperf...done
struct timeval {
  tv_sec = 1090825887;
  tv_usec = 878018;
}
```

GNU debugger (gdb)

- Homepage and docs: `http://www.gnu.org/software/gdb/gdb.html`
- Many GUIs cgdb (ncurses), ddd (X), kdbg, (KDE, probably best of these) and insight (X)
- Start debugging: `gdb [--args] ./executable [arg1]`
- Set source file directories `dir` *`sub1`*`:`*`sub2`*
- Starting the execution: `run` Stopping the execution: `kill` (`quit` to exit gdb)
- Help: help [command]
- List code lines: `list` (params can be file, function, lines or address)
- To see, where it crashed: `where` or `backtrace`, which gives list of frames

- To choose frame: `frame` *nmb* or up or down. Print vars `info locals`

```
(gdb) up
#1  0x0804b7da in Server::Run (this=0x806f220) at Server.cpp:110
110                 currLen = recv( mSettings->mSock, mBuf,
mSettings->mBufLen, 0 );
(gdb) info locals
currLen = 8192
mBuf_UDP = (UDP_datagram *) 0x806f238
reportstruct = (ReportStruct *) 0x805b060
(gdb) list
105         if ( reportstruct != NULL ) {
106             reportstruct->packetID = 0;
```

- Breakpoints: (temporary (deleted when hit)) `tbreak` and non-temporary: `break`. Both takes as a parameter function name or line number

- Priting variable: `print variable`

- Setting varible: `set variable=3`

- Possibly to monitor (`watch`) a variable (read,write,condition)

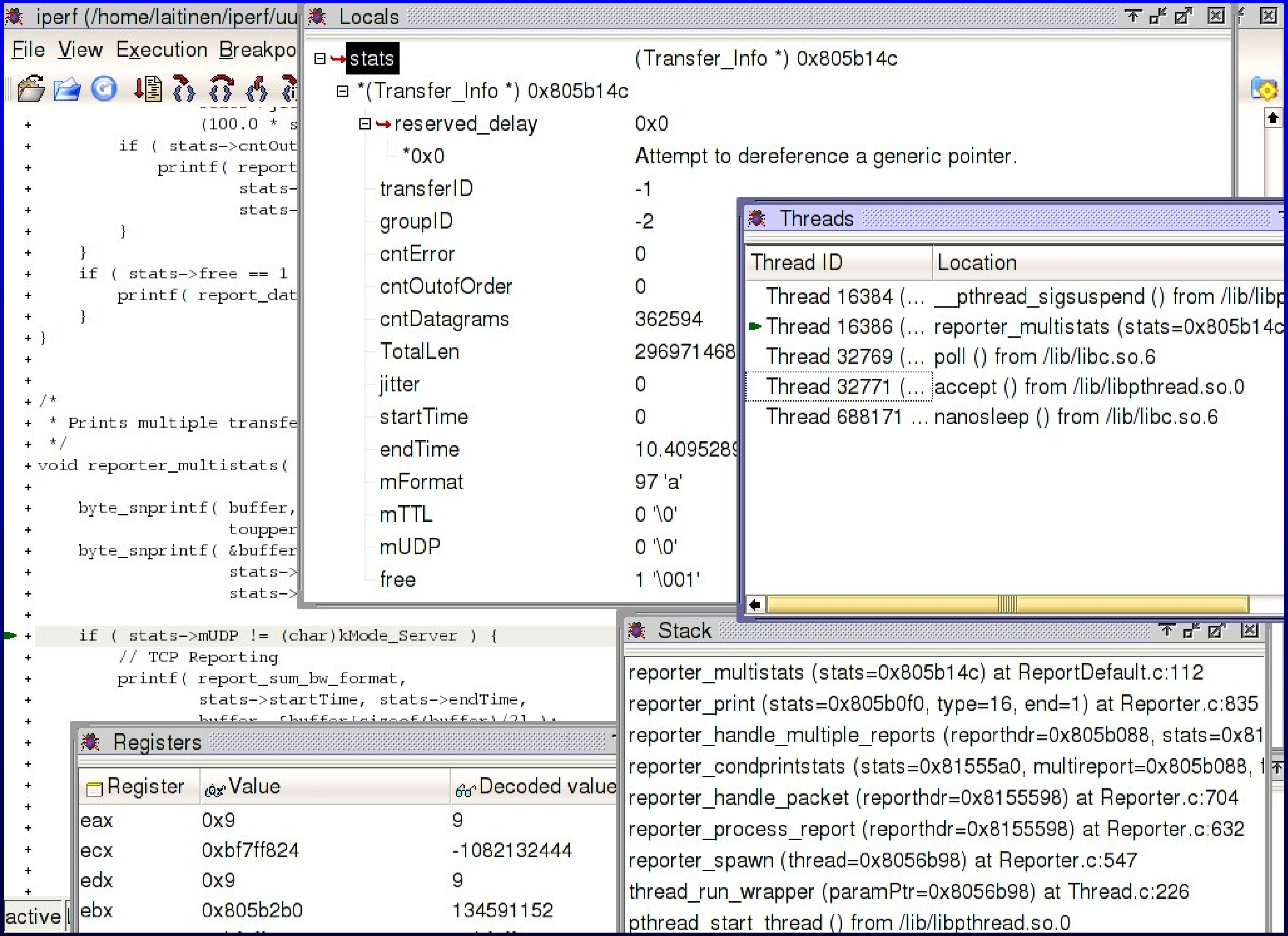
- Has problems with multiple threads (NPTL library)
  - ⋆ `reading register r1 (#1): No such process.` or
  - ⋆ `Cannot find thread 655401: no thread to satisfy query`
  - ⋆ Probably using back port `setenv LD_ASSUME_KERNEL 2.4.1` works better

```
Program received signal SIGSEGV, Segmentation fault.
[Switching to Thread 65541 (LWP 2447)]
0x0804fbf1 in ReportPacket (agent=0xd156728, packet=0x805bab8)
    at Reporter.c:321
321             int index = agent->reporterindex;
Current language:  auto; currently c

(gdb) list ReportPacket
314      * the arrival or departure of a "packet" (for TCP it
315      * will actually represent many packets). This needs to
316      * be as simple and fast as possible as it gets called for
317      * every "packet".
318      */
319     void ReportPacket( ReportHeader* agent, ReportStruct *packet ) {
320         if ( agent != NULL ) {
321             int index = agent->reporterindex;
```

```
322             /*
323              * First find the appropriate place to put the
information

(gdb) print index
$1 = 1074857152
(gdb) whatis agent->reporterindex
type = int

(gdb) print agent->reporterindex
Cannot access memory at address 0xd156728

(gdb) break ReportPacket
Breakpoint 1 at 0x804fbe4: file Reporter.c, line 320.

(gdb) run
The program being debugged has been started already.
Start it from the beginning? (y or n) y

Breakpoint 1, ReportPacket (agent=0x8083d28, packet=0x8083d08)
    at Reporter.c:320
320         if ( agent != NULL ) {
```

```
(gdb) next
321             int index = agent->reporterindex;

(gdb) print agent->reporterindex
$3 = 699

(gdb) clear ReportPacket
Deleted breakpoints 2 1

(gdb) continue
Program received signal SIGSEGV, Segmentation fault.
[Switching to Thread 65541 (LWP 2542)]
0x401d114c in memcpy () from /lib/libc.so.6

(gdb) where
#0  0x401d114c in memcpy () from /lib/libc.so.6
#1  0x0804fc67 in ReportPacket (agent=0x805baf8, packet=0x805c000)
    at Reporter.c:341
#2  0x0804b8e7 in Server::Run() (this=0x805baf8) at Server.cpp:122
#3  0x0804a243 in server_spawn (thread=0x805c000) at Launch.cpp:85
#4  0x08049c7f in thread_run_wrapper (paramPtr=0x805c000) at
Thread.c:216
#5  0x40109c40 in pthread_start_thread_event () from
```

```
/lib/libpthread.so.0

(gdb) list Reporter.c:341
336                 thread_rest();
337                 index = agent->reporterindex;
338             }
339
340             // Put the information there
341             memcpy( agent->data + agent->agentindex, packet,
sizeof(ReportStruct) );

(gdb) print agent->data
No symbol "agent" in current context.
```



```
+                (100.0 * s
+           if ( stats->cntOut
+               printf( report
+                       stats-
+                       stats-
+           }
+       }
+       if ( stats->free == 1
+           printf( report_dat
+       }
+ }
+
+
+ /*
+  * Prints multiple transfe
+  */
+ void reporter_multistats(
+
+       byte_snprintf( buffer,
+                      toupper
+       byte_snprintf( &buffer
+                      stats->
+                      stats->
+
+       if ( stats->mUDP != (char)kMode_Server ) {
+           // TCP Reporting
+           printf( report_sum_bw_format,
+                   stats->startTime, stats->endTime,
```

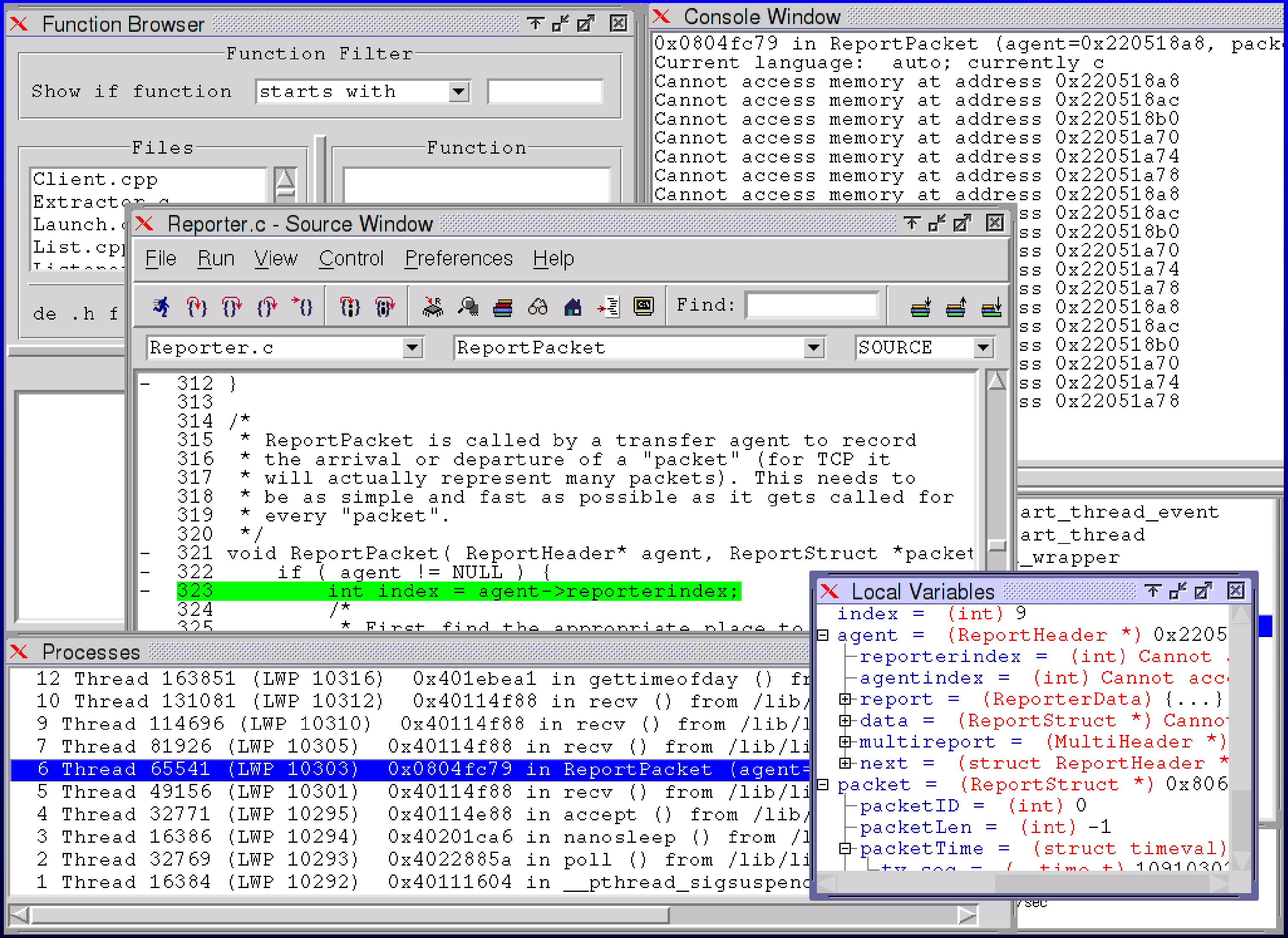
```
⊟ stats                   (Transfer_Info *) 0x805b14c
  ⊟ *(Transfer_Info *) 0x805b14c
    ⊟ reserved_delay      0x0
        *0x0              Attempt to dereference a generic pointer.
      transferID          -1
      groupID             -2
      cntError            0
      cntOutofOrder       0
      cntDatagrams        362594
      TotalLen            296971468
      jitter              0
      startTime           0
      endTime             10.409528
      mFormat             97 'a'
      mTTL                0 '\0'
      mUDP                0 '\0'
      free                1 '\001'
```

Thread ID	Location
Thread 16384 (…	__pthread_sigsuspend () from /lib/libp
Thread 16386 (…	reporter_multistats (stats=0x805b14c
Thread 32769 (…	poll () from /lib/libc.so.6
Thread 32771 (…	accept () from /lib/libpthread.so.0
Thread 688171 …	nanosleep () from /lib/libc.so.6

```
reporter_multistats (stats=0x805b14c) at ReportDefault.c:112
reporter_print (stats=0x805b0f0, type=16, end=1) at Reporter.c:835
reporter_handle_multiple_reports (reporthdr=0x805b088, stats=0x81
reporter_condprintstats (stats=0x81555a0, multireport=0x805b088, 
reporter_handle_packet (reporthdr=0x8155598) at Reporter.c:704
reporter_process_report (reporthdr=0x8155598) at Reporter.c:632
reporter_spawn (thread=0x8056b98) at Reporter.c:547
thread_run_wrapper (paramPtr=0x8056b98) at Thread.c:226
pthread_start_thread () from /lib/libpthread.so.0
```

Register	Value	Decoded value
eax	0x9	9
ecx	0xbf7ff824	-1082132444
edx	0x9	9
ebx	0x805b2b0	134591152

Figure 2: kdbg: A nice user interface for gdb



```
Client.cpp
Extractor.c
Launch.c
List.cp
```

```
0x0804fc79 in ReportPacket (agent=0x220518a8, pack
Current language:  auto; currently c
Cannot access memory at address 0x220518a8
Cannot access memory at address 0x220518ac
Cannot access memory at address 0x220518b0
Cannot access memory at address 0x22051a70
Cannot access memory at address 0x22051a74
Cannot access memory at address 0x22051a78
Cannot access memory at address 0x220518a8
ss 0x220518ac
ss 0x220518b0
ss 0x22051a70
ss 0x22051a74
ss 0x22051a78
ss 0x220518a8
ss 0x220518ac
ss 0x220518b0
ss 0x22051a70
ss 0x22051a74
ss 0x22051a78
```

```
}

/*
 * ReportPacket is called by a transfer agent to record
 * the arrival or departure of a "packet" (for TCP it
 * will actually represent many packets). This needs to
 * be as simple and fast as possible as it gets called for
 * every "packet".
 */
void ReportPacket( ReportHeader* agent, ReportStruct *packet
    if ( agent != NULL ) {
        int index = agent->reporterindex;
        /*
         * First find the appropriate place to
```

```
art_thread_event
art_thread
_wrapper
```

```
index = (int) 9
agent = (ReportHeader *) 0x2205
  reporterindex = (int) Cannot
  agentindex = (int) Cannot acc
  report = (ReporterData) {...}
  data = (ReportStruct *) Canno
  multireport = (MultiHeader *)
  next = (struct ReportHeader *
packet = (ReportStruct *) 0x806
  packetID = (int) 0
  packetLen = (int) -1
  packetTime = (struct timeval)
```

```
12 Thread 163851 (LWP 10316)  0x401ebea1 in gettimeofday () fr
10 Thread 131081 (LWP 10312)  0x40114f88 in recv () from /lib/
9 Thread 114696 (LWP 10310)  0x40114f88 in recv () from /lib/l
7 Thread 81926 (LWP 10305)  0x40114f88 in recv () from /lib/li
6 Thread 65541 (LWP 10303)  0x0804fc79 in ReportPacket (agent=
5 Thread 49156 (LWP 10301)  0x40114f88 in recv () from /lib/li
4 Thread 32771 (LWP 10295)  0x40114e88 in accept () from /lib/
3 Thread 16386 (LWP 10294)  0x40201ca6 in nanosleep () from /l
2 Thread 32769 (LWP 10293)  0x4022885a in poll () from /lib/li
1 Thread 16384 (LWP 10292)  0x40111604 in __pthread_sigsuspend
```

Figure 3: Insight: Yet another GUI for gdb. A bit nicer than ddd.

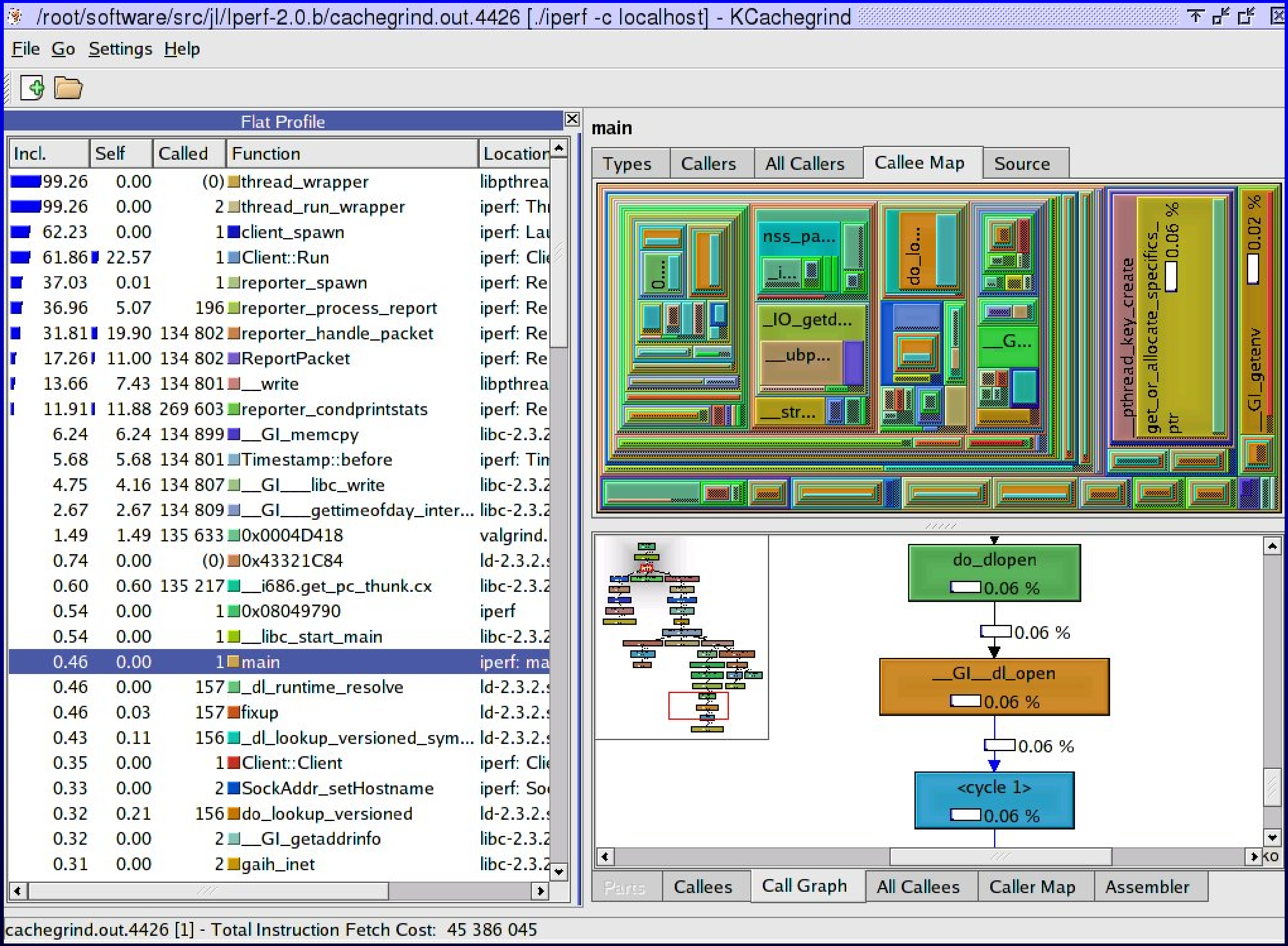
Post-mortem analysis of a core dump

- No need to run in debugger
- To get core dump when segfaulting, `ulimit -c none` (may be a big)

```
(gdb) core core.23257
Core was generated by ‘./iperf -s’.
Program terminated with signal 11, Segmentation fault.
Reading symbols from /usr/lib/libstdc++.so.5...done.
Loaded symbols for /usr/lib/libstdc++.so.5
..
(gdb) backtrace
#0  0x200000000038ddd0 in _int_free () from /lib/tls/libc.so.6.1
#1  0x200000000038b620 in free () from /lib/tls/libc.so.6.1
#2  0x2000000000151480 in operator delete(void*) ()
  from /usr/lib/libstdc++.so.5
#3  0x400000000000c070 in Settings_Destroy
(mSettings=0x600000000001bc80)    at Settings.cpp:266
..
```

Valgrind

- GPL licensed: `http://valgrind.kde.org/`
- Support NPTL threads, but not IA-64
- Takes binary file, written in any language, as a parameter
- Can start debugger on errorneous point (–db-attach=yes)
- Logfile: `--log-file=something`
- Couple projects to create GUI (Gnome/Kde devel needed)
- Also profiler projects (picture from kcachegrind



Incl.	Self	Called	Function	Location
99.26	0.00	(0)	thread_wrapper	libpthrea
99.26	0.00	2	thread_run_wrapper	iperf: Thı
62.23	0.00	1	client_spawn	iperf: Lau
61.86	22.57	1	Client::Run	iperf: Clie
37.03	0.01	1	reporter_spawn	iperf: Re
36.96	5.07	196	reporter_process_report	iperf: Re
31.81	19.90	134 802	reporter_handle_packet	iperf: Re
17.26	11.00	134 802	ReportPacket	iperf: Re
13.66	7.43	134 801	__write	libpthrea
11.91	11.88	269 603	reporter_condprintstats	iperf: Re
6.24	6.24	134 899	__GI_memcpy	libc-2.3.2
5.68	5.68	134 801	Timestamp::before	iperf: Tin
4.75	4.16	134 807	__GI___libc_write	libc-2.3.2
2.67	2.67	134 809	__GI___gettimeofday_inter...	libc-2.3.2
1.49	1.49	135 633	0x0004D418	valgrind.
0.74	0.00	(0)	0x43321C84	ld-2.3.2.s
0.60	0.60	135 217	__i686.get_pc_thunk.cx	libc-2.3.2
0.54	0.00	1	0x08049790	iperf
0.54	0.00	1	__libc_start_main	libc-2.3.2
0.46	0.00	1	main	iperf: ma
0.46	0.00	157	_dl_runtime_resolve	ld-2.3.2.s
0.46	0.03	157	fixup	ld-2.3.2.s
0.43	0.11	156	_dl_lookup_versioned_sym...	ld-2.3.2.s
0.35	0.00	1	Client::Client	iperf: Clie
0.33	0.00	2	SockAddr_setHostname	iperf: Soc
0.32	0.21	156	do_lookup_versioned	ld-2.3.2.s
0.32	0.00	2	__GI_getaddrinfo	libc-2.3.2
0.31	0.00	2	gaih_inet	libc-2.3.2

cachegrind.out.4426 [1] - Total Instruction Fetch Cost: 45 386 045

Figure 4: kcachegrind

- Valgrind core can be extended with plugins (–tool)

  ⋆ **memcheck**: To detect memory-management problems (C/C++) use of uninitialised mem, R/W freed memory or inappropriate places, memory leaks, mismatched allocation and frees (C vs. C++), passing uninitialised/unaddressible mem to syscalls and misuse of some POSIX theads

```
==9112== Thread 2:
==9112== Invalid read of size 8
==9112==    at 0x8050782: reporter_handle_multiple_reports
(Reporter.c:716)
==9112==    by 0x8050AAF: reporter_condprintstats (Reporter.c:781)

==9112== Thread 2:
==9112== Invalid read of size 4
==9112==    at 0x80508CF: reporter_handle_multiple_reports
(Reporter.c:741)
==9112==    by 0x8050AAF: reporter_condprintstats (Reporter.c:781)
==9112==    by 0x805071A: reporter_handle_packet (Reporter.c:702)
==9112==    by 0x80504E7: reporter_process_report (Reporter.c:630)
==9112==  Address 0x1BB69FD4 is not stack'd, malloc'd or (recently)
free'd
```

```
==9112==  Address 0x1BB69FE4 is 12 bytes before a block of size 152
free’d
==9112==    at 0x1B904249: free (vg_replace_malloc.c:153)
==9112==    by 0x1B9FC880: my_free (vg_libpthread.c:346)
==9112==    by 0x1B9FD830: thread_wrapper (vg_libpthread.c:850)
==9112==    by 0xB000F6A8: do__quit (vg_scheduler.c:1861)

==9112== ERROR SUMMARY: 259 errors from 26 contexts (suppressed: 21
1)
==9112== malloc/free: in use at exit: 10644 bytes in 10 blocks.
==9112== malloc/free: 85 allocs, 75 frees, 273392 bytes allocated
```

- **addrcheck**: Checks fewer errors Memcheck (faster, less mem needed)
- **cachegrind**: Aims to point cache misses in code. Gives amount of cache misses, memory references and instructions per line/function/programs

```
==9142== I   refs:      19,709,481
==9142== I1  misses:        13,196
==9142== L2i misses:         3,286
==9142== I1  miss rate:        0.6%
==9142== L2i miss rate:        0.1%
==9142==
==9142== D   refs:      13,011,762  (8,491,314 rd + 4,520,448 wr)
```

```
==9142== D1  misses:        290,854  (  116,930 rd +   173,924 wr)
==9142== L2d misses:         15,739  (    8,875 rd +     6,864 wr)
==9142== D1  miss rate:         2.2% (      1.3%   +       3.8%  )
==9142== L2d miss rate:         0.1% (      0.1%   +       0.1%  )
==9142==
==9142== L2 refs:           304,050  (  130,126 rd +   173,924 wr)
==9142== L2 misses:          19,025  (   12,161 rd +     6,864 wr)
==9142== L2 miss rate:          0.0% (      0.0%   +       0.1%  )
```

⋆ **massif**: Profiles program's heap. Produces graphs of usage over time and places where most memory was allocated

```
==24753== Total spacetime:   5,465,480,517 ms.B
==24753== heap:              96.0%
==24753== heap admin:         0.2%
==24753== stack(s):           3.6%
```

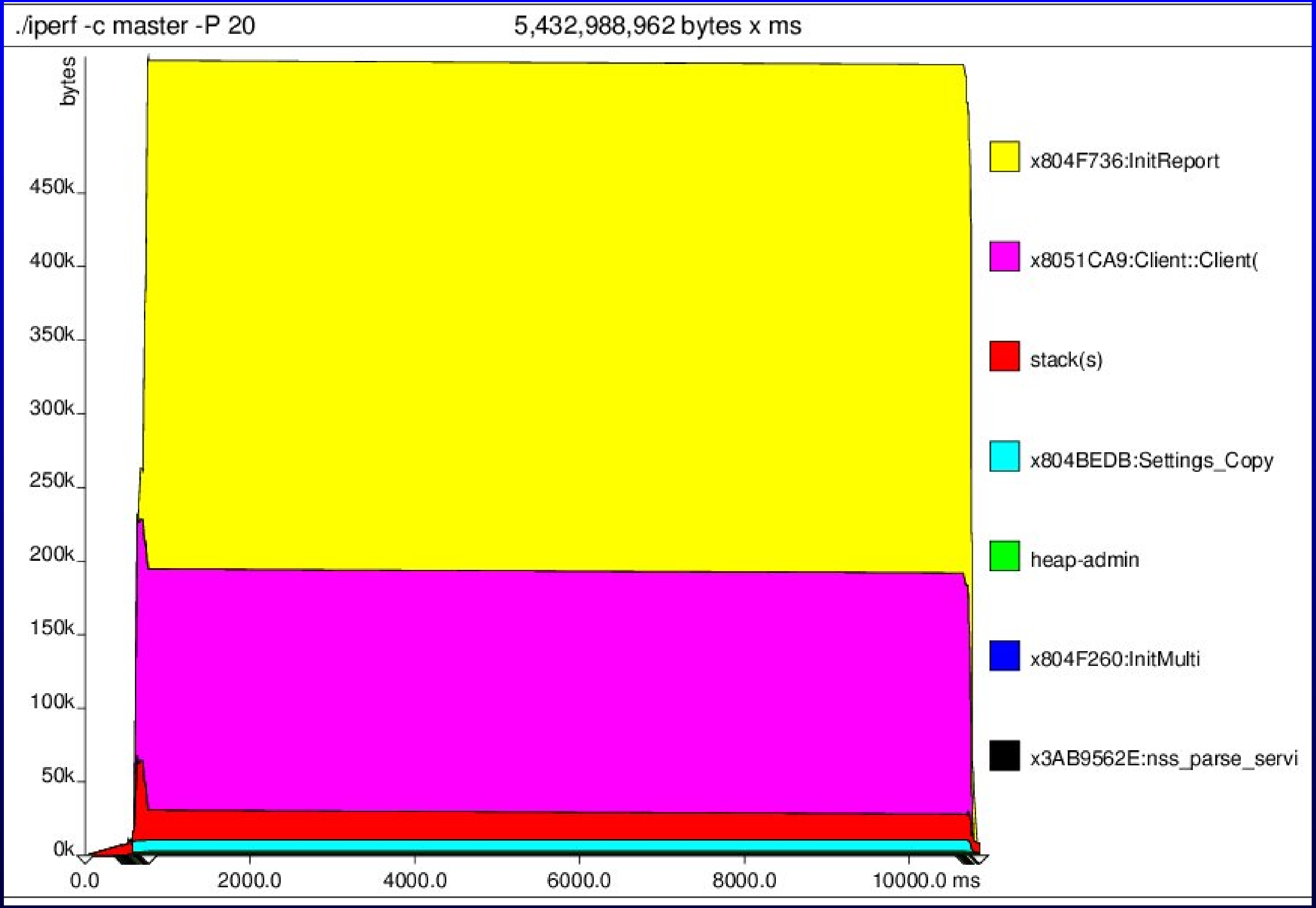


Figure 5: A graph from Valgrind's massif heap profiler

⋆ **helgrind**: detects data races for C/C++ Pthreads. Experimental.

```
==9322== Thread 2:
```

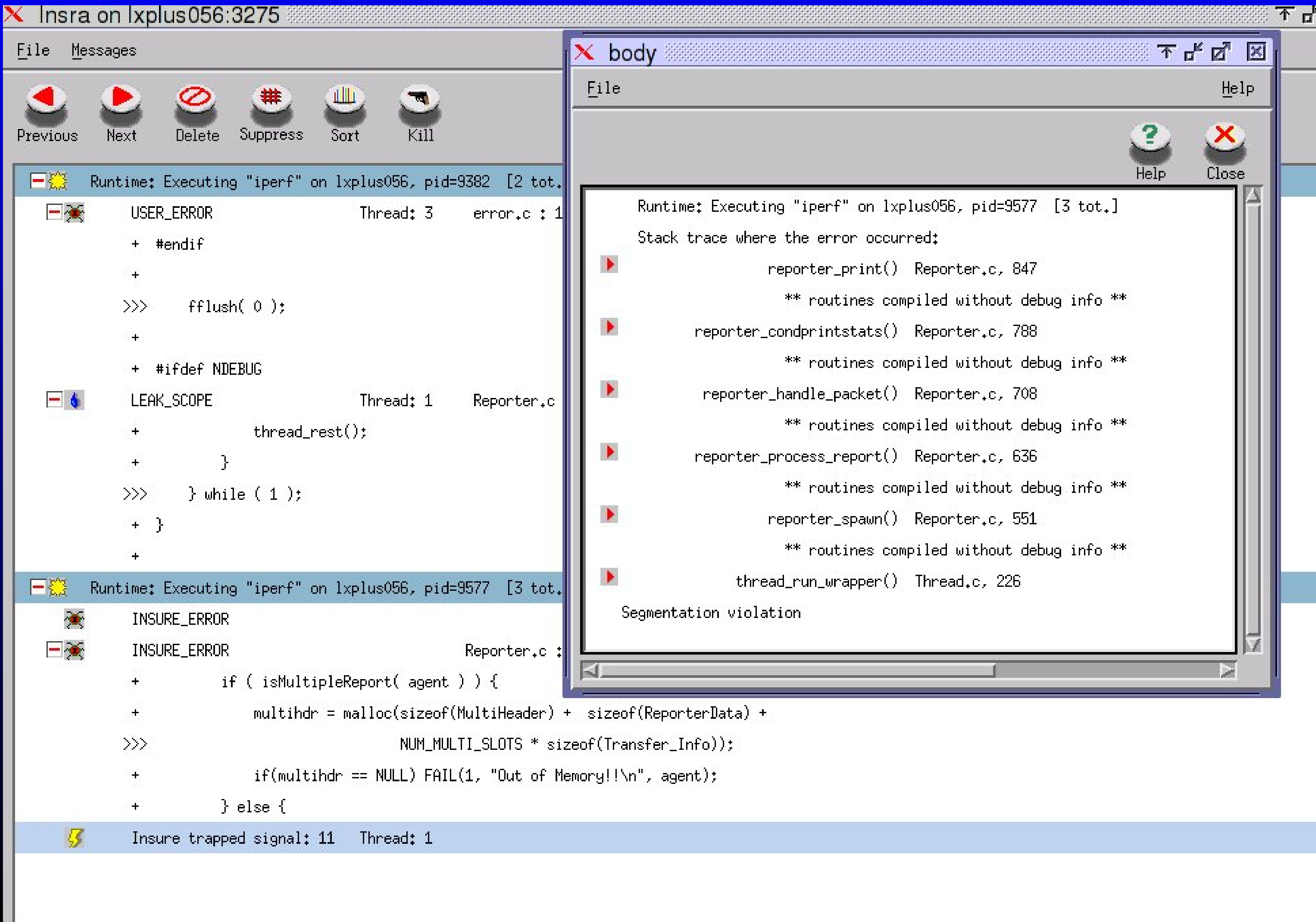
```
==9322== Possible data race reading variable at 0x805720C (ReportRoo
==9322==    at 0x8050228: reporter_spawn (Reporter.c:543)
```

Insure++

- Memory corruption/leak check on runtime for C/C++ codes
- Commercial product of Panasoft http://www.parasoft.com/jsp/products/screensho
- Expensive (1 year license $15k), but CERN has five floating lic.
- Should be more advanced than mpatrol or Valgrind

To use CERN’s Insure++

- Instructions and license usage: `http://product-support.web.cern.ch/product-support/sdt/insu` `html`
- Works at least in lxplus machines (remember `ssh -X`)
- Copy to your dir `insure.csh (.sh)` from `/afs/cern.ch/pttools/Insure/insure/linux/`
- Correct: `setenv INSURE /afs/cern.ch/pttools/Insure/insure/` (official instruction creates symlinks)
- Run: `source insure.csh` (or `./insure.sh`) (sets the PATH)
- Use `insure` instead of your compiler
- Execute program as usual. Messages appears for eache process to gui, where user can browse them through and dive to source code (from vi editor, get out with shift+zz)

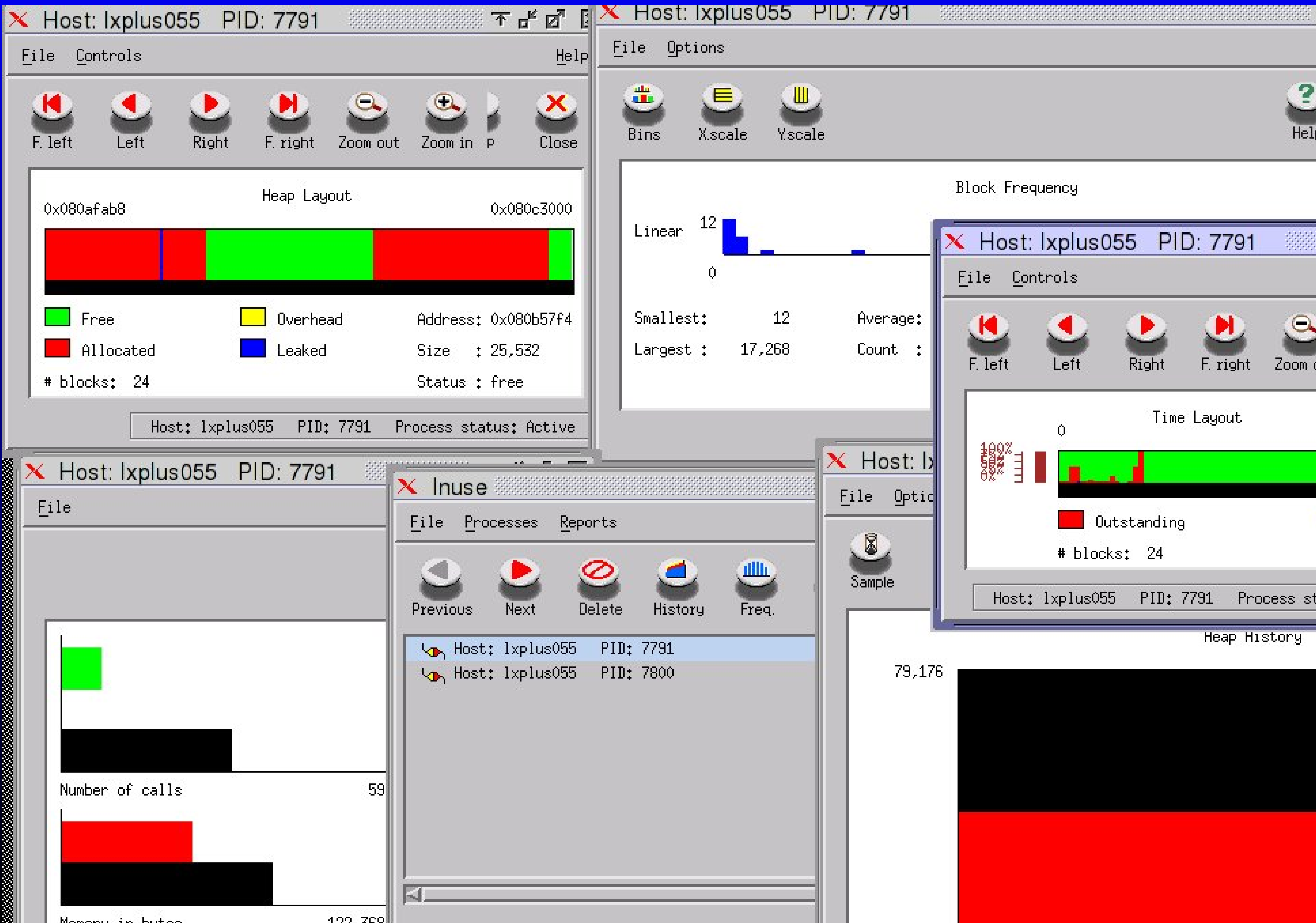


```
Runtime: Executing "iperf" on lxplus056, pid=9382  [2 tot.
    USER_ERROR                 Thread: 3     error.c : 1
        +  #endif
        +
        >>>     fflush( 0 );
        +
        +  #ifdef NDEBUG
    LEAK_SCOPE                 Thread: 1     Reporter.c
        +               thread_rest();
        +           }
        >>>     } while ( 1 );
        +  }
        +
Runtime: Executing "iperf" on lxplus056, pid=9577  [3 tot.
    INSURE_ERROR
    INSURE_ERROR                            Reporter.c :
        +           if ( isMultipleReport( agent ) ) {
        +               multihdr = malloc(sizeof(MultiHeader) +  sizeof(ReporterData) +
        >>>                                  NUM_MULTI_SLOTS * sizeof(Transfer_Info));
        +               if(multihdr == NULL) FAIL(1, "Out of Memory!!\n", agent);
        +           } else {
    Insure trapped signal: 11    Thread: 1
```

```
Runtime: Executing "iperf" on lxplus056, pid=9577  [3 tot.]
Stack trace where the error occurred:
               reporter_print()  Reporter.c, 847
                    ** routines compiled without debug info **
         reporter_condprintstats()  Reporter.c, 788
                    ** routines compiled without debug info **
          reporter_handle_packet()  Reporter.c, 708
                    ** routines compiled without debug info **
         reporter_process_report()  Reporter.c, 636
                    ** routines compiled without debug info **
               reporter_spawn()  Reporter.c, 551
                    ** routines compiled without debug info **
            thread_run_wrapper()  Thread.c, 226
Segmentation violation
```

Inuse

- In the same directory as Insure++
- Inuse shows charts of memory usage (when, how much)
- and memory leaks



Heap Layout

0x080afab8 0x080c3000

🟩 Free 🟨 Overhead Address: 0x080b57f4

🟥 Allocated 🟦 Leaked Size : 25,532

blocks: 24 Status : free

Host: lxplus055 PID: 7791 Process status: Active

Block Frequency

Linear 12 0

Smallest: 12 Average:

Largest : 17,268 Count :

Time Layout

0

🟥 Outstanding

blocks: 24

Host: lxplus055 PID: 7791 Process st

Host: lxplus055 PID: 7791

Host: lxplus055 PID: 7800

Heap History

79,176

Number of calls 59

Memory in bytes 122,768

ccmalloc

- GPL licensed. "Swiss made" `http://www.inf.ethz.ch/personal/biere/projects/ccmalloc/`
- Notices at least access to freed data, but not all illegal reads.
- Replace compiler with ccmalloc and include couple libraries in linking. Configurations made to file .ccmalloc (template included).

```
.----------------.
|ccmalloc report|
=======================================================
| total # of|   allocated | deallocated |     garbage |
+-----------+-------------+-------------+-------------+
|      bytes|      279080 |      260708 |       18372 |
+-----------+-------------+-------------+-------------+
|allocations|          68 |          61 |           7 |
+-----------------------------------------------------+
| number of checks: 1                                 |
| number of counts: 129                               |
| retrieving function names for addresses ... done.    |
| reading file info from gdb ... done.                 |
| sorting by number of not reclaimed bytes ... done.   |
| number of call chains: 6                             |
| number of ignored call chains: 0                     |
| number of reported call chains: 6                    |
| number of internal call chains: 6                    |
| number of library call chains: 0                     |
=======================================================
|
* 44.6% = 8192 Bytes of garbage allocated in 1 allocation
|       |
|       @ [
|       |   +0x0808d34c
|       | ]
|       |
|       |       0x???????? in <???>
|       |
|       |       0x4023469a in <clone>
|       |
|       |       0x40111e51 in <pthread_start_thread>
|       |
|       |       0x0804a3b0 in <thread_run_wrapper>
|       |                 at Thread.c:232
|       |
|       |      0x0804a7ba in <listener_spawn>
|       |                 at Launch.cpp:68
|       |
|       |      0x0804ab35 in <Listener>
|       |                 at Listener.cpp:94
|       |
|       |      0x400b84df in <operator new[](unsigned int)>
|       |
|       |      0x400b83ae in <operator new(unsigned int)>
|       |
|       ‘-----> 0x080536bf in <malloc>
```

mpatrol

- Mpatrol `http://www.cbmamiga.demon.co.uk/mpatrol/`
- Not so easy to use as Valgrind, but quite near to it. Manual 264 pages.
- Has plenty of options. Long list of features.
- mpatrol.h must be included to source file before others to instrument it or to use `--dynamic` switch
- Should support threads and work in multiple operation systems (originally Amiga project)
- Can produce some kind of profile data also (last three below) (apparently I had problems to tell source code files)

```
ERROR: [ILLMEM]: illegal memory access at address 0x00000000
    0x00000000 (24 bytes) {malloc:471:0} [-|-|-]
        0x400F748E ???
        0x08052286 _ZN6Client3RunEv+394
```

```
            0x0804A287 client_spawn+123
            0x08049C91 thread_run_wrapper+103
            0x4014FE51 ???
            0x4027269A ???

        call stack
            0x401C4658 ???
            0x4002E87E ???
            0x4003C328 ???
            0x4004025A ???
            0x400F748E ???
            0x08052286 _ZN6Client3RunEv+394
            0x0804A287 client_spawn+123
            0x08049C91 thread_run_wrapper+103
            0x4014FE51 ???
            0x4027269A ???
....
allocation count:  472
allocation peak:   257 (1047793 bytes)
allocation limit:  0 bytes
allocated blocks:  259 (1047841 bytes)
marked blocks:     0 (0 bytes)
freed blocks:      0 (0 bytes)
free blocks:       70 (103 bytes)
internal blocks:   8 (131072 bytes)
total heap usage:  1183744 bytes
total compared:    164 bytes
total located:     636 bytes
total copied:      33643 bytes
total set:         31056 bytes
total warnings:    0
total errors:      1
...
```

```
                                    ALLOCATION BINS

                                 (number of bins: 1024)

                       allocated                              unfreed
                ---------------------------------  ---------------------------------
     size     count        %      bytes        %    count        %      bytes        %

        4         2     0.20          8     0.00        0     0.00          0     0.00
        5         3     0.29         15     0.01        0     0.00          0     0.00
...

                                      MEMORY LEAKS

                                (maximum stack depth: 1)

                 unfreed                              allocated
----------------------------------------  ----------------
      %      bytes        %    count        %      bytes    count   function

  83.76       1712   100.00       12   100.00       1712       12   0x40122781
   9.20        188    35.01        1    33.33        537        3   0x401219EA
....

                                   DIRECT ALLOCATIONS

                         (0 < s <= 32 < m <= 256 < l <= 2048 < x)

          allocated                               unfreed
-----------------------------  -----------------------------
   bytes        %    s  m  l  x      bytes       %    s  m  l  x    count   function

   73856    37.27            37          0    0.00                       7   _objalloc_alloc
   72448   36.56     37               0    0.00                    283  0x080593C1
   23736   11.98      .  1 11         0    0.00                      4  bfd_malloc
...
```

The backup slide..

- Also debuggers get stuck quite often
- If **Ctrl+C** does not help. Try **Ctrl+Z** and `kill %1`
- To see running prosesses `ps -aux`
- to kill them `kill [-9] process_id`
- or `killall [-9] process_name`