RRDebug

B.Tech. Project

Ankur Dahiya (2008CS10159)

Guide: Prof. Sorav Bansal

May 10, 2012

Outline

1. Motivation
2. Goals
3. Progress
   - RRDebug
   - Indexer
   - Database
   - Query Engine
4. Demonstration
5. Future Work

Motivation

- Concurrency bugs are a major issue for multithreaded software
- Hard to reproduce => Hard to debug
- VM RR can solve the reproduction problem

Project Goals

- Develop a debugger that takes advantage of VM RR
- Do this *efficiently*
- Develop an intuitive GUI for the debugger
- Focus on a single query for kernel variables - “Where was this variable modified?”

RRDebug

- GDB-like interface to answer queries
- Built using python
- Uses gdb to interact with the VM
- Runs real-time queries against kernel variables
- Runs the Indexer

Indexer

- Single steps through each instruction
- Logs all instructions that affect memory in a SQLite Database
- Results in a tremendous slowdown!

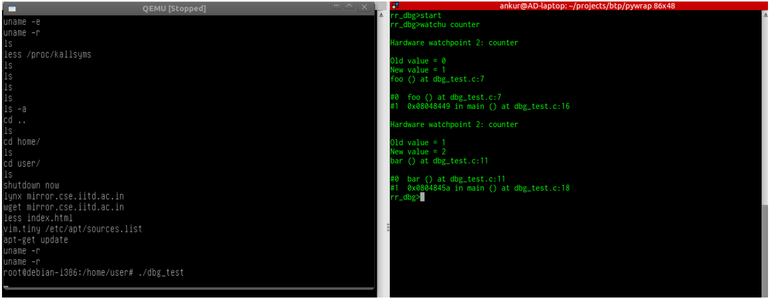
Database

- Database columns: EIP, Timestamp, Memory Address, Old Data, New Data, Backtrace.
- Indexed on Memory Address and Timestamp

Query Engine

- A GUI built using python
- Allows user to query the database built by the indexer
- Queries can include variable names, which are converted to addresses using gdb
- Queries can go backward in time

RRDebug

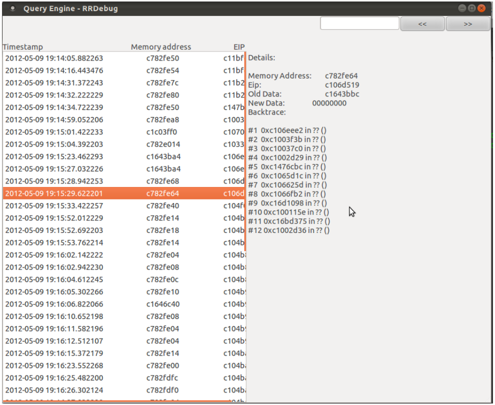


```
uname -e
uname -r
ls
less /proc/kallsyms
ls
ls
ls
ls
ls -a
cd ..
ls
cd home/
ls
cd user/
ls
shutdown now
lynx mirror.cse.iitd.ac.in
wget mirror.cse.iitd.ac.in
less index.html
vim.tiny /etc/apt/sources.list
apt-get update
uname -r
uname -r
root@debian-i386:/home/user# ./dbg_test
```

```
rr_dbg>start
rr_dbg>watchu counter

Hardware watchpoint 2: counter

Old value = 0
New value = 1
foo () at dbg_test.c:7

#0  foo () at dbg_test.c:7
#1  0x08048449 in main () at dbg_test.c:16

Hardware watchpoint 2: counter

Old value = 1
New value = 2
bar () at dbg_test.c:11

#0  bar () at dbg_test.c:11
#1  0x0804845a in main () at dbg_test.c:18
rr_dbg>
```

Figure: Screenshot of RRDebug along with qemu

Query Engine



Timestamp	Memory address	EIP
2012-05-09 19:14:05.882263	c782fe50	c11bf
2012-05-09 19:14:16.443476	c782fe54	c11bf
2012-05-09 19:14:31.372243	c782fe7c	c11b2
2012-05-09 19:14:32.222229	c782fe80	c11b2
2012-05-09 19:14:34.722239	c782fe50	c147b
2012-05-09 19:14:59.052206	c782fea8	c1003
2012-05-09 19:15:01.422233	c1c03ff0	c1070
2012-05-09 19:15:04.392203	c782e014	c1033
2012-05-09 19:15:23.462293	c1643ba4	c106e
2012-05-09 19:15:27.032226	c1643ba4	c106e
2012-05-09 19:15:28.942253	c782fe68	c106d
2012-05-09 19:15:29.622201	c782fe64	c106d
2012-05-09 19:15:33.422257	c782fe40	c104f
2012-05-09 19:15:52.012229	c782fe14	c104b
2012-05-09 19:15:52.692203	c782fe18	c104b
2012-05-09 19:15:53.762214	c782fe14	c104b
2012-05-09 19:16:02.142222	c782fe04	c104b
2012-05-09 19:16:02.942230	c782fe08	c104b
2012-05-09 19:16:04.612245	c782fe0c	c104b
2012-05-09 19:16:05.302266	c782fe10	c104b
2012-05-09 19:16:06.822066	c1646c40	c104b
2012-05-09 19:16:10.652198	c782fe08	c104b
2012-05-09 19:16:11.582196	c782fe04	c104b
2012-05-09 19:16:12.512107	c782fe04	c104b
2012-05-09 19:16:15.372179	c782fe14	c104b
2012-05-09 19:16:23.552268	c782fe00	c104b
2012-05-09 19:16:25.482200	c782fdfc	c104b
2012-05-09 19:16:26.302124	c782fdf0	c104b

```
Details:

Memory Address:    c782fe64
Eip:               c106d519
Old Data:          c1643bbc
New Data:        00000000
Backtrace:

#1  0xc106eee2 in ?? ()
#2  0xc1003f3b in ?? ()
#3  0xc10037c0 in ?? ()
#4  0xc1002d29 in ?? ()
#5  0xc1476cbc in ?? ()
#6  0xc1065d1c in ?? ()
#7  0xc106625d in ?? ()
#8  0xc1066fb2 in ?? ()
#9  0xc16d1098 in ?? ()
#10 0xc100115e in ?? ()
#11 0xc16bd375 in ?? ()
#12 0xc1002d36 in ?? ()
```

Figure: Screenshot of RRDebug Query Engine

Future Work

- Optimize the Indexer
  - No need to always single-step
  - Crucial parts can be re-written in C
- Add more features to the Query Engine
- Test the system more thoroughly

Thanks.
Suggestions and Feedback?