

A method to detect memory leaks & corruption

- Who are we?
- Why another tool?
- Tool features
- The idea
- Implementation
- Limitations
- Download
- Questions

References:

- Glibc Manual (3.2.2.9 Heap Consistency Checking)
- <http://duma.sourceforge.net/>
- <http://valgrind.org/>
- <http://sourceware.org/binutils/docs-2.20/ld/index.html>
- <http://g.oswego.edu/dl/html/malloc.html>
- <http://library.gnome.org/devel/glib/stable/glib-running.html>

Ravi Sankar Guntur
ravisankar.g@gmail.com
27th Oct 2010

Who are we?

Part of team that works at Samsung India, Bangalore, for developing a **Smart-phone platform** based on GNU/Linux.

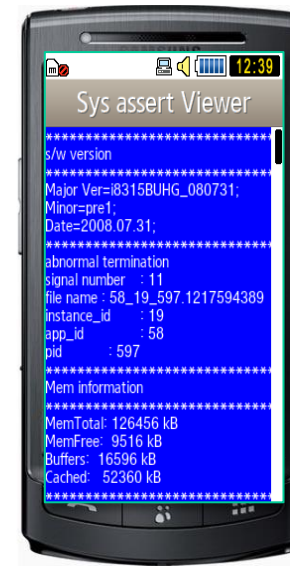
- Platform

- ARM Processor.
- 256 MB main memory.

- Major portion of debugging time spent on fixing **memory leaks and memory corruption** issues.



Vodafone - Samsung H1



Why we wanted another tool?

Tools we tried,

- Memory leaks
 - GNU libc's mtrace
 - Valgrind – memcheck
 - memprof
- Memory corruptions
 - GNU libc's MALLOC_CHECK_=2
 - DUMA (efence)
 - Valgrind - memcheck
- Issues
 - Huge memory overhead.
 - No support for GUI scenario based testing
 - Separate tools for memory leak and corruption
 - No support of call graph

Tool features

- Less memory overhead
- Provides call graph
- Support for scenario based memory leak testing
- Single tool to detect memory leaks and heap consistency

Memory leak report...

```
block of 18 bytes was not freed. (id: 0x12740)
/usr/lib/libsafemem.so(malloc+0x10)[0x4002c174]
/lib/libc.so.6(+0x25a54)[0x4009fa54]
/lib/libc.so.6(bindtextdomain+0x1c)[0x4009fc9c]
/usr/lib/libgtk-x11-2.0.so.0(+0x44482918)
/usr/lib/libg...args+0x40)[0x44...
/usr/lib/libgtk-x11-2.0.so.0(gtk_init_check+0x8)[0x444...
/usr/lib/libgtk-x11-2.0.so.0(gtk_init+0x8)[0x44482c74]
/usr/bin/window(main+0x28)[0x9388]
/lib/libc.so.6(__libc_start_main+0x118)[0x4008f4c4]
```

backtrace

Memory corruption report...

```
window:2238:Mon Oct 18 16:02:38 2010
memcpy-error: 0x3521c8 Allowed 8B, Needs 33B

Backtrace ...
/usr/lib/libsafemem.so(+0x2898)[0x4002a898]
/usr/lib/libsafemem.so(__bt+0x10)[0x4002a8f8]
/usr/lib/libsafemem.so(memcpy+0xe8)[0x4002afb4]
/usr/bin/window(segmentation_fault_handler+0x48)[0x9194]
/usr/lib/libgobject-2.0.so.0(g_closure_invoke+0x1f0)[0x422b31fc]
/usr/lib/libgobject-2.0.so.0(g_closure_invoke+0x1f0)[0x422b31fc]
/usr/lib/libgobject-2.0.so.0(g_signal_emit_valist+0x744)[0x422c9860]
/usr/lib/libgobject-2.0.so.0(g_signal_emit+0x20)[0x422c9d34]
/usr/lib/libgtk-x11-2.0.so.0(+0x443cf968)
/usr/lib/libgobject-2.0.so.0(g_cclosure_marshal_VOID__VOID+0x6c)[0x422c1224]
/usr/lib/libgobject-2.0.so.0(+0x422b15f4)
/usr/lib/libgobject-2.0.so.0(g_closure_invoke+0x1f0)[0x422b31fc]
/usr/lib/libgobject-2.0.so.0(+0x422c7e48)
```

backtrace

The Idea – Memory Leak

“For every allocated block add *Header* and *Footer*. Add size and caller information in the *Header*”

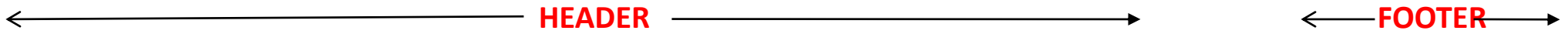
“Erase the *Header & Footer*, before de-allocating the *block*”

”Scan the heap region for yet un-freed blocks and construct the call graph for every block found”

The implementation

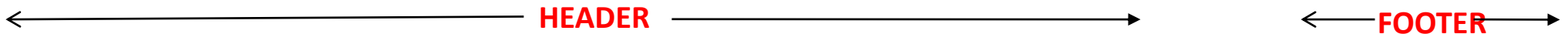
Buffer typedef

Size	Header Sig1	Header Sig2	MODE	# of Frames	Frame1	----	Frame 30	User Data	Footer Sig1	Footer Sig2



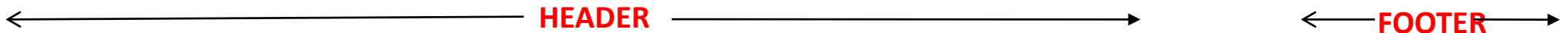
Buffer allocated when memory leak check is ON

Size	Header Sig1	Header Sig2	MODE	# of Frames	Frame1	----	Frame 30	User Data	Footer Sig1	Footer Sig2
28	0xdead beef	0xabc deff	1	12	0x400 01234		NULL		0xcafe babe	0xdeaf feed



Buffer allocated when memory leak check is OFF

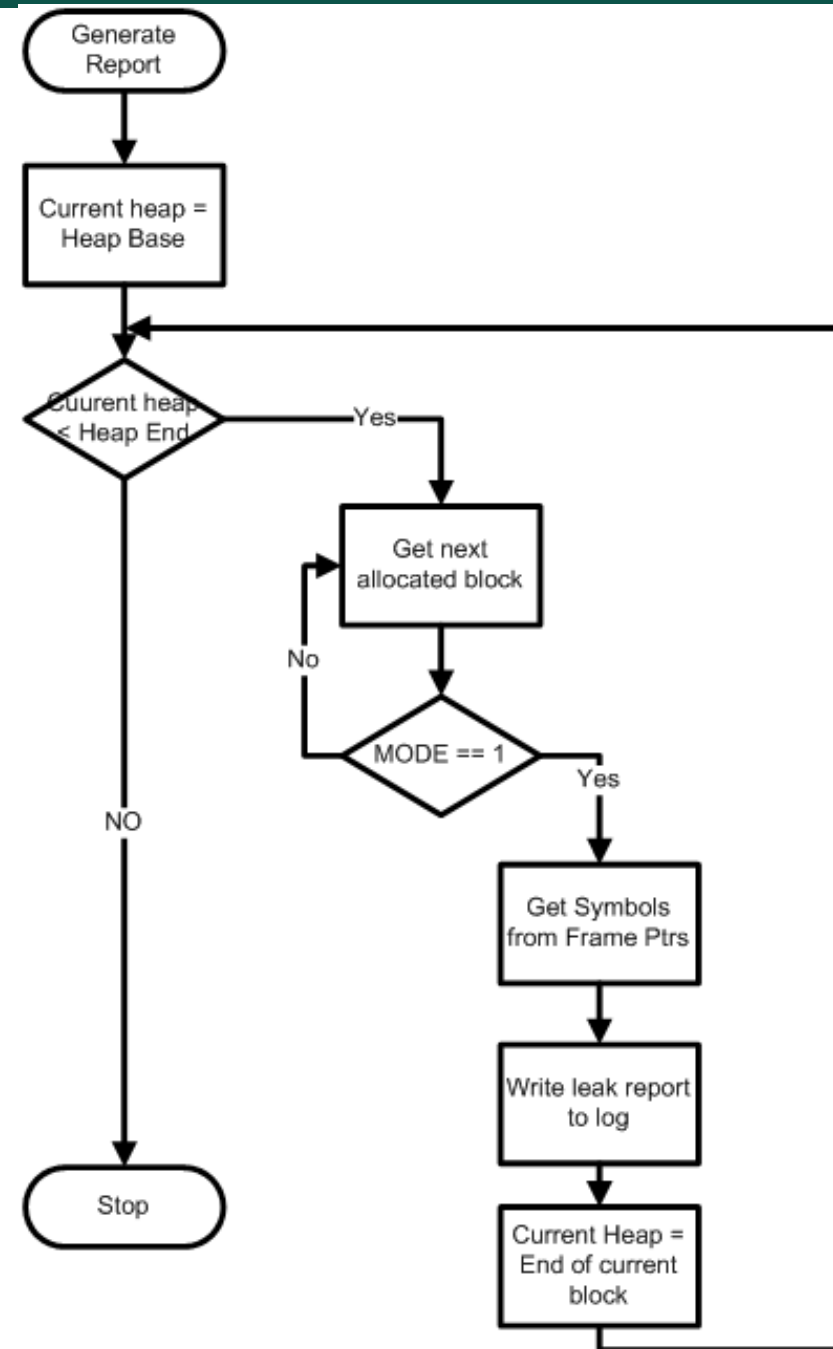
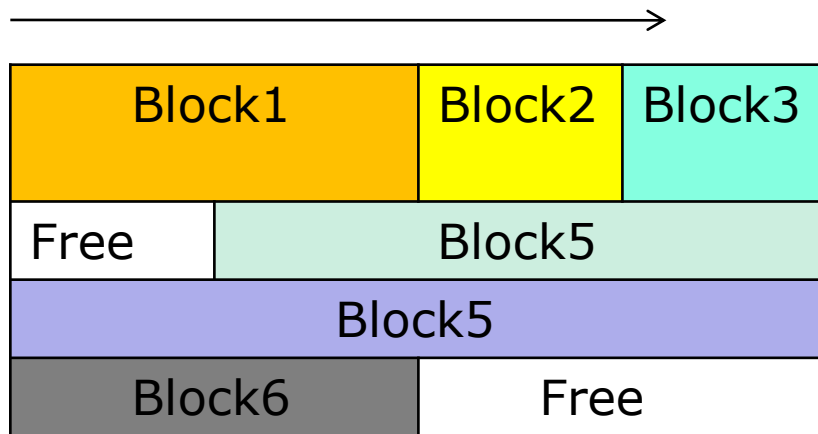
Size	Header Sig1	Header Sig2	MODE	# of Frames	Frame1	----	Frame 30	User Data	Footer Sig1	Footer Sig2
80	0xdead beef	0xabc deff	0	12	0x400 01234		NULL		0xcafe babe	0xdeaf feed



The implementation – Memory leak

Generate memory leak report

Example heap layout



The Idea – Memory corruption

“One of the source of memory corruption is wrong usage of parameters to lib C’s string manipulation functions”

“use LD_PRELOAD to preload DSO of modified functions”

“Given destination buffer, get the size from Header and check for possible memory corruption”

The implementation – Memory corruption

To check heap consistency,

- Preloaded string wrappers check if the destination address is from heap region or not.
 - If from heap
 - Checks the validity of the buffer.
 - Checks if number of bytes > allocated size.
 - If yes, error details will be written to log file and *SIGSEGV* will be raised.
 - If no, proceed normally
 - if not from heap, proceed normally
 - *realloc*, *calloc*, and *free* will check header and footer for integrity.

Limitations

- Shell script sets up the environment variables like LD_PRELOAD, LEAK_MODE, G_SLICE and launches the debugged program.
- “-fno-omit-frame-pointer” is needed for backtrace()
- if no “-rdynamic”, use addr2line to convert VMA to Symbol name.

Download

- Integrated tool with couple of bug fixes is not yet uploaded to public domain. (contact the author to check the latest status)
- Separate tools to detect memory corruption and leaks are available at,
 - `git clone git://git.savannah.nongnu.org/safeheap.git`
 - `git clone git://git.savannah.nongnu.org/memleak.git`

Questions

END...