Memory Analyzer Manual

What Memory Analyzer Does

Gives you real-time update on RSS (Resident Set Size, or total RAM usage) breakdown, which is generated based on smaps and callstack list that Smaps Analyzer generates. All the RSS segment managed under smaps is examined. RSS breakdown (or how to group them) is configurable.

(Former Smaps Analyzer)

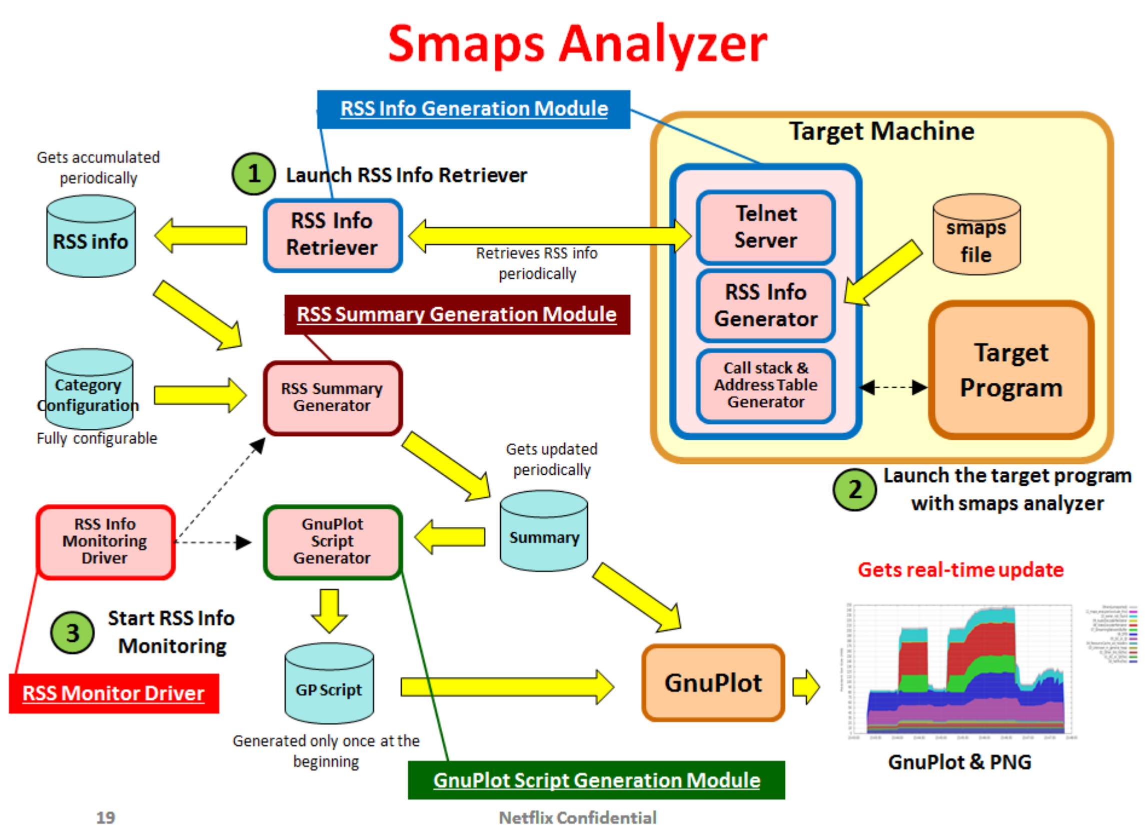
How It Works

- RSS information gets generated on target machine side with pre-loaded shared library (no change in the target program is necessary)
- Collects the RSS information through telnet served by the above shared library
- Analyze the RSS information and generates RSS beakdown
- Feed the RSS breakdown into GnuPlot (It generates the GnuPlot script based on RSS breakdown)
- GnuPlot does the periodic update with updated RSS breakdown

Dependencies

- Requires smaps support in Linux kernel (CONFIG_MMU)
- Requires GNU libc's backtrace and backtrace_symbols
- Requires symbols in the target executable

Memory Analyzer Components


Smaps Analyzer
RSS Info Generation Module
Target Machine
Gets accumulated periodically
1 Launch RSS Info Retriever
RSS info
RSS Info Retriever
Retrieves RSS info periodically
Telnet Server
smaps file
RSS Info Generator
RSS Summary Generation Module
Category Configuration
Fully configurable
RSS Summary Generator
Call stack & Address Table Generator
Target Program
Gets updated periodically
2 Launch the target program with smaps analyzer
RSS Info Monitoring Driver
GnuPlot Script Generator
Summary
Gets real-time update
3 Start RSS Info Monitoring
RSS Monitor Driver
GP Script
GnuPlot
Generated only once at the beginning
GnuPlot & PNG
GnuPlot Script Generation Module
19
Netflix Confidential

RSS Information Generation Module (Target device side)

- Call stack & address table generation module
  - By interposing malloc and mmap, it generates arrays of callstack and returned address
- RSS information generation
  - By going through each line in /proc/[pid]/maps file, uses pre-generated callstack & address information to determine which virtual memory section belongs to whom
  - By using mincore function, it checks how much of the VM is actually in resident
- Telnet server for simple control
  - This allows a user to interact with Maps Analyzer through telnet
  - User can trigger maps analyzer report through telnet

RSS Information Generation Module (Off-target device side)

- Simply retrieves the RSS information through telnet and save it into a file

RSS Summary Generation Module

- Analyze the RSS information and consolidates into RSS breakdown using configuration file, which is highly configurable

GnuPlot Script Generation Module

- Generates GnuPlot script based on RSS summary file

RSS Monitor Driver

- Kicks off RSS summary generation module, GnuPlot script generation module, and GnuPlot

How to build and use it

How to build it

```
cd your_build_directory_for_nrdp14.1
../netflix/configure --gibbon-memanalyzer=static --nrdapp-tools=memanalyzer
--release --no-small
(../netflix/configure --gibbon-memanalyzer=shared --nrdapp-tools=memanalyzer
--release --no-small)
make -j7
```

How to run it

```
cd your_build_directory
cd tools/memanalyzer
./run_memanalyzer.sh
```

That should be all you need! (to run it for device. please refer to run_memanalyzer.sh

Preparation of RSS breakdown configuration file (You can skip this if you are running against ref app)

1. Create a configuration file that looks like below

```
mem_analyzer(exclude) own *
00_netflix(file) maps netflix
01_JSC(file) maps WTF,JavaScriptCore,libQt
03_Unknown_in_general_heap maps possible_holes_in_heap
02_Other_3rd_lib(file) maps *
mem_analyzer(exclude) *
libmemanalyzer.so;libmemanalyzer.so,libmemanalyzer.so;libc,libmemanalyzer.so;l
d-linux
04_JSC(GrowHeap) * GrowHeap
05_JSC(Scavenger) * runScavengerThread
06_WTF(reserveUncommitted) * reserveUncommitted
07_JSC(Other) * ScriptEngine,JSC::DecodedData,JSString,WTF,JSC,libQt
12_VideoDecoder * VideoRendererDFB
11_Streaming *
BufferManagerNative,MediaFragment,SampleWriterNative,Mp4Demultiplexer
13_AudioDecoder * libasound,AudioRenderer,libpuls,AudioDecoder
08_DFB * directfb
15_owner_not_found(includes_resource_cache) none
no_owner_found_in_rss_chunk,possible_holes_in_rss,no_call_stack_avaialble
09_HttpService * HttpService
10_DnsMgr/DrmMgr/TeeApi/crypto/surfaceDecoder/etc *
DiskCache,DrmManager,DnsManager,CollectorThread,TeeApi,libcrypto,SurfaceDecode
rs,DataBuffer
10_DnsMgr/DrmMgr/TeeApi/crypto/surfaceDecoder/etc mmap *
14_NonCategorized * *
```

a. Each line consists of the following 3 fields
    i. 1st field: Category name
    ii. 2nd field: allocation type (pick from maps/malloc/calloc/realloc/mmap/mmap64/own/total or *)
    iii. 3rd field: Target string (typically, function/class name) concatenated with ",". These strings will be grep-ed in the callstack table

2. Edit the monitoring driver file (smaps_analyzer/sa_monitor_driver.sh)

```
# set category configuration file
categoryFile="./summary_generation/configurations/nrdp40_conf.txt"  # <---
modify this part to point to your own configuration file
```

How to run it (manually)

1. Start RSS info retriever on your machine (any machine that is in the same subnet as your target machine)

```
cd smaps_analyzer
./rss_info_generation/ma_rss_info_retriever.py -d your_output_directory
```

2. Run any app you want as follows (make sure that you have symbols not being stripped)

```
MA_LINES_IN_SMAPS=11 LD_PRELOAD=path_to_libmemanalyzer.so ./your_program  #
default value for MA_LINES_IN_SMAPS is 14, which works fine with Ubuntu12.04
```

a. Supported environment variable options

Environment variable	**Default value**	**Description**
SA_LINES_IN_SMAPS	14	Number of lines in between each mapped regions in /proc/(pid)/smaps
SA_PORT	1234	The port that the telnet server in maps_analyzer.c is listening to
SA_MALLOC_SIZE_TH	4KB	maps_analyzer will generate callstack for malloc with any size bigger than this threashold
SA_REPORT_PAGE_NUM_TH	25	maps_analyzer will generate report on any memory mapped region that has RSS size bigger than this threashold * pagesize (4KB) (inclidive)

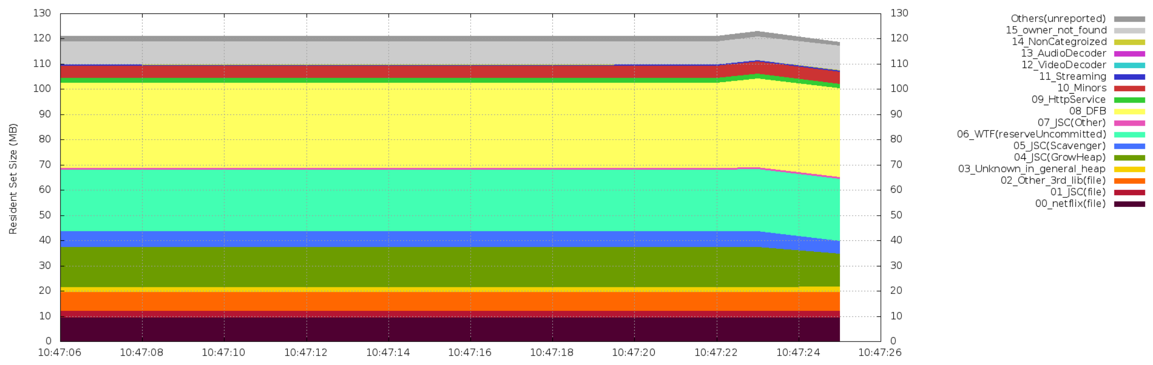
i. For example, SA_LINES_IN_SMAPS needs to be set to 14 if the smaps file looks like below

```
kmiyagi@kmiyagi-desktop:/mnt/hdd1/data/git_repos/4.0/nrd-4.0.2-rele
ase-1911259/build_release_symbol/src/platform/gibbon$ cat
/proc/`pidof netflix`/smaps |more
08048000-0969d000 r-xp 00000000 08:21 56891472
/mnt/hdd1/data/git_repos/4.0/nrd-4.0.2-release-1911259/build/src/pl
atform/gibbon/netflix
Size:              22868 kB
Rss:               11560 kB
Pss:               11560 kB
Shared_Clean:          0 kB
Shared_Dirty:          0 kB
Private_Clean:     11560 kB
Private_Dirty:         0 kB
Referenced:        11560 kB
Anonymous:             0 kB
AnonHugePages:         0 kB
Swap:                  0 kB
KernelPageSize:        4 kB
MMUPageSize:           4 kB
Locked:                0 kB
0969d000-096c8000 rw-p 01655000 08:21 56891472
/mnt/hdd1/data/git_repos/4.0/nrd-4.0.2-release-1911259/build/src/pl
atform/gibbon/netflix
Size:                172 kB
Rss:                 128 kB
```

3. Run monitor script

```
cd netflix/tools/memanalyzer
./ma_rss_monitor_driver.sh your_output_directory
```

4. That's it. You'll see GnuPlot window and realtime update that looks like this. Also, image file (your_output_directory.png) will be saved as well.



Example Outputs (when run with 'your_output_directory' set to 'example')

- RSS Info (example/sa_20140109-143038.log)

```
9969664  maps    netflix
    172032  maps    netflix
    245760  none    no_owner_found_in_rss_chunk
    104480  malloc
libstdc;libstdc;netflix::device::esplayer::DrmManager::initPlayReady;netflix::
device::esplayer::DrmManager::DrmManager;netflix::device::esplayer::DrmManager
::create;
     16384  calloc
libX11.so.6;libdirectfb_x11.so;libdirectfb_x11.so;libdirectfb-1.2.so.9;libdire
ctfb-1.2.so.9;libdirectfb-1.2.so.9;libfusion-1.2.so.9;libdirectfb-1.2.so.9;lib
directfb-1.2.so.9;
     20864  calloc
libxcb.so.1;libxcb.so.1;libxcb.so.1;libX11.so.6;libX11.so.6;libdirectfb_x11.so
;libdirectfb_x11.so;libdirectfb-1.2.so.9;libdirectfb-1.2.so.9;libdirectfb-1.2.
so.9;
     28672  malloc  libstdc;std::vector<netflix::base::Variant,
std::allocator<netflix::base::Variant>
>::_M_insert_aux;netflix::nrdlog::Instrumentation::CollectorThread::bufferEven
t;netflix::nrdlog::Instrumentation::CollectorThread::Run;
     19764  realloc
netflix::base::DataBuffer::reserve;netflix::base::Configuration::resourceConte
nt;netflix::application::Application::resourceContent;netflix::application::gi
bbon::NetworkManager::handleFileRequest;
     61440  malloc
libstdc;netflix::base::pclist::IProducerListView<netflix::device::Mp4Demultipl
exer::DataBlock>::NodePool::NodePool;netflix::device::PlaybackDevice::Playback
Device;
     61440  malloc
libstdc;netflix::base::pclist::IProducerListView<netflix::device::Mp4Demultipl
exer::DataBlock>::NodePool::NodePool;netflix::device::PlaybackDevice::Playback
Device;
     37284  malloc
libcares.so.2;netflix::net::DnsManager::Channel::create;netflix::net::DnsManag
er::create;netflix::NrdLib::NrdLib;netflix::nbp::NrdpBridge::startNrdLib;netfl
ix::nbp::NBP::startupNrdlib;netflix::application::http::HTTPNBP::startupNrdlib
;
      8192  realloc libcrypto.so.1.0.0;
   2998152  maps    possible_holes_in_heap
   1048576  none    no_owner_found_in_rss_chunk
     65536  mmap64
WTF::OSAllocator::reserveAndCommit;WTF::OSAllocator::reserveUncommitted;WTF::P
ageAllocationAligned::allocate;JSC::CopiedSpace::getFreshBlock;JSC::CopiedSpac
e::getFreshBlock;JSC::CopiedSpace::tryAllocateSlowCase;

...
    530432  none    possible_holes_in_rss
    118784  maps    ld-2.15.so
     61440  maps    [stack]
 215027712  total   smaps_total
```

- RSS summary (example/summary/sa_20140109-143038.log.summary)

```
9.42 00_netflix(file)
   2.56 01_JSC(file)
   7.50 02_Other_3rd_lib(file)
   2.05 03_Unknown_in_general_heap
  15.96 04_JSC(GrowHeap)
   6.23 05_JSC(Scavenger)
  24.61 06_WTF(reserveUncommitted)
   0.72 07_JSC(Other)
  35.24 08_DFB
   1.85 09_HttpService
   4.82 10_Minors
   0.43 11_Streaming
   0.00 12_VideoDecoder
   0.02 13_AudioDecoder
   0.10 14_NonCategroized
   9.38 15_owner_not_found
   2.21 Others(unreported)
 123.09 Total
```

- GnuPlot script (example.gp)

```
# set variables
width=1900
height=600
interval=1
outimagename="/mnt/hdd1/data/git_repos/tools/smaps_analyzer/test.png"
finalSummary="/mnt/hdd1/data/git_repos/tools/smaps_analyzer/test/summary/final
Summary.txt"
categories="00_netflix(file) 01_JSC_or_Qt(file) 02_Other_3rd_lib(file)
03_Unknown_in_general_heap 04_ResourceCache_wo_headers 05_JSC_or_Qt 06_DFB
07_StreamingNetworkBuffer 08_VideoDecoderRenderer 09_AudioDecoderRenderer
10_owner_not_found 11_maps_analyzer(exclude_this) Others(unreported)"
# set color schemes
colors="#5555aa #55aa55 #aa5555 #aaaa55 #55aaaa #aa55aa #3333cc #33cc33
#cc3333 #cccc33 #33cccc #cc33cc #cccccc"
# set terminal
set terminal wxt noraise title 'monitored smaps analysis on ' . finalSummary
# set legends
set key outside;
# set axis and grid
set xdata time
set timefmt "%Y%m%d-%H%M%S"
set format x "%H:%M:%S"
set ytics 10
set grid front xtics ytics
set ylabel "Resident Set Size (MB)"
set terminal wxt size 1900, 600# plot
plot \
  finalSummary every ::1 using 1:($2 + $3 + $4 + $5 + $6 + $7 + $8 + $9 + $10
+ $11 + $12 + $13 + $14) t word(categories, 13) w filledcurves x1 lc rgb
word(colors, 13), \
  finalSummary every ::1 using 1:($2 + $3 + $4 + $5 + $6 + $7 + $8 + $9 + $10
+ $11 + $12 + $13) t word(categories, 12) w filledcurves x1 lc rgb
word(colors, 12), \
  finalSummary every ::1 using 1:($2 + $3 + $4 + $5 + $6 + $7 + $8 + $9 + $10
+ $11 + $12) t word(categories, 11) w filledcurves x1 lc rgb word(colors, 11),
\
  finalSummary every ::1 using 1:($2 + $3 + $4 + $5 + $6 + $7 + $8 + $9 + $10
+ $11) t word(categories, 10) w filledcurves x1 lc rgb word(colors, 10), \
  finalSummary every ::1 using 1:($2 + $3 + $4 + $5 + $6 + $7 + $8 + $9 + $10)
t word(categories, 9) w filledcurves x1 lc rgb word(colors, 9), \
  finalSummary every ::1 using 1:($2 + $3 + $4 + $5 + $6 + $7 + $8 + $9) t
word(categories, 8) w filledcurves x1 lc rgb word(colors, 8), \
  finalSummary every ::1 using 1:($2 + $3 + $4 + $5 + $6 + $7 + $8) t
word(categories, 7) w filledcurves x1 lc rgb word(colors, 7), \
  finalSummary every ::1 using 1:($2 + $3 + $4 + $5 + $6 + $7) t
word(categories, 6) w filledcurves x1 lc rgb word(colors, 6), \
  finalSummary every ::1 using 1:($2 + $3 + $4 + $5 + $6) t word(categories,
5) w filledcurves x1 lc rgb word(colors, 5), \
  finalSummary every ::1 using 1:($2 + $3 + $4 + $5) t word(categories, 4) w
filledcurves x1 lc rgb word(colors, 4), \
  finalSummary every ::1 using 1:($2 + $3 + $4) t word(categories, 3) w
filledcurves x1 lc rgb word(colors, 3), \
  finalSummary every ::1 using 1:($2 + $3) t word(categories, 2) w
filledcurves x1 lc rgb word(colors, 2), \
  finalSummary every ::1 using 1:($2) t word(categories, 1) w filledcurves x1
lc rgb word(colors, 1)
# save the image
set terminal png size width, height
set output outimagename
replot
```

```
# pause and reload
pause interval
reread
```

Additional Information

https://nrd.netflix.com/display/DOC41/Memory+Profiling