

# Example Debugging Session Using DTrace and mdb

# Want to Follow Along?



- Download SmartOS and install node and npm install dtrace-provider
- Or, go to [www.joyent.com](http://www.joyent.com) and provision a node SmartOS instance
- Or, Set up an account on [www.joyent.com/products/manta](http://www.joyent.com/products/manta) and:
  - `npm install -g manta`
  - `mput -f leak2.js ~/stor/leak2.js`
  - `mlogin -s /$MANTA_USER/stor/leak2.js`
- `leak2.js` is available at [github.com/max123/nodeconfeu](https://github.com/max123/nodeconfeu)

# The Problem: Memory Leak



```
maxb@manta # node leak2.js &
node leak2.js &
[1] 77649
maxb@manta # STARTING
maxb@manta # prstat -c -p 77649
Please wait...
  PID USERNAME  SIZE  RSS STATE PRI NICE   TIME    CPU PROCESS/NLWP
  77649 root      58M   44M cpu5   59    0   0:00:06  0.2% node/3
Total: 1 processes, 3 lwps, load averages: 0.05, 0.02, 0.01
  PID USERNAME  SIZE  RSS STATE PRI NICE   TIME    CPU PROCESS/NLWP
  77649 root      59M   45M sleep  59    0   0:00:07  0.2% node/3
Total: 1 processes, 3 lwps, load averages: 0.05, 0.02, 0.01
  PID USERNAME  SIZE  RSS STATE PRI NICE   TIME    CPU PROCESS/NLWP
  77649 root      62M   48M sleep  59    0   0:00:07  0.2% node/3
Total: 1 processes, 3 lwps, load averages: 0.05, 0.02, 0.01'
(^C)
maxb@manta #
```

# Take a Couple of Dumps



```
maxb@manta # gcore 77649
gcore: core.77649 dumped
maxb@manta # mv core.77649 core.0
maxb@manta #
```

<wait a minute or so...>

```
maxb@manta # gcore 77649
gcore: core.77649 dumped
maxb@manta # mv core.77649 core.1
maxb@manta #
```

# Run mdb(1) on the Dumps



```
maxb@manta # mdb core.0
Loading modules: [ libc.so.1 ld.so.1 ]
> ::load v8.so
V8 version: 3.14.5.8
Autoconfigured V8 support from target
C++ symbol demangling enabled
> ::findjsobjects
...
82825be1      2045          4 Error: stack, arguments, type, name
8459f8d1      2399          4 Object: value, writable, enumerable, ...
828c4181      5248          7 Array
84b0ad95      4934         12 PropertyDescriptor: value_, hasValue_, ...
> 84b0ad95::findjsobjects | ::jsprint
...
{
  value_: {
    leak: "hello leaky world",
  },
  hasValue_: true,
  writable_: true,
  hasWritable_: true,
  enumerable_: true,
  hasEnumerable_: true,
  configurable_: true,
  hasConfigurable_: true,
  get_: undefined,
  hasGetter_: false,
  set_: undefined,
  hasSetter_: false,
}
...
```

# Run mdb(1) on the Second Dump



```
maxb@manta # mdb core.1
Loading modules: [ libc.so.1 ld.so.1 ]
> ::load v8.so
V8 version: 3.14.5.8
Autoconfigured V8 support from target
C++ symbol demangling enabled
> ::findjsobjects
...
828c4191      1400      12 Array
82825be1      5641       4 Error: stack, arguments, type, name
828c4181      8897       7 Array
84b0ad95      8000      12PropertyDescriptor: value_, hasValue_, ...
> 84b0ad95::findjsobjects | ::jsprint !more
...
{
  value_: {
    leak: "hello leaky world",
  },
  hasValue_: true,
  writable_: true,
  hasWritable_: true,
  enumerable_: true,
  hasEnumerable_: true,
  configurable_: true,
  hasConfigurable_: true,
  get_: undefined,
  hasGetter_: false,
  set_: undefined,
  hasSetter_: false,
}
...
```

# How is Memory Allocated?



- Either on the “heap” via `brk/sbrk` (2) System Call
- Or via `mmap/mmap64` (2) system calls
- To “see” address space:

```
maxb@manta # pmap -x 77649
```

```
77649:  node leak2.js
```

Address	Kbytes	RSS	Anon	Locked	Mode	Mapped File
08043000	20	20	20	-	rw---	[ stack ]
08050000	7920	7920	-	-	r-x--	node
0881B000	56	56	16	-	rwx--	node
08829000	<b>3436</b>	3436	3436	-	rwx--	[ heap ]
80B00000	36	36	36	-	rw---	[ anon ]
80B0A000	980	980	980	-	rwx--	[ anon ]
81000000	36	36	36	-	rw---	[ anon ]
8100A000	980	980	980	-	rwx--	[ anon ]
81500000	1024	1024	1024	-	rw---	[ anon ]
81700000	1024	1024	1024	-	rw---	[ anon ]
81B00000	1024	1024	1024	-	rw---	[ anon ]

```
...
```

```
total Kb  374676 372288 360784
```

```
maxb@manta #
```

# Run pmap(2) Again a little later



```
maxb@manta # pmap -x 77649
```

```
77649:  node leak2.js
```

Address	Kbytes	RSS	Anon	Locked	Mode	Mapped File
08043000	20	20	20	-	rw---	[ stack ]
08050000	7920	7920	-	-	r-x--	node
0881B000	56	56	16	-	rwx--	node
08829000	<b>3436</b>	3436	3436	-	rwx--	[ heap ]
80B00000	36	36	36	-	rw---	[ anon ]
80B0A000	980	980	980	-	rwx--	[ anon ]

```
...
```

```
total Kb 380812 377612 366108
```

```
maxb@manta #
```



# Node is Using mmap(2) to Grow Space



- “heap” isn’t growing
- Node is using:
  - `addr = mmap64(NULL, size, prot, ...MAP_ANON..., -1, 0);`

# DTrace mmap(2) calls by node



## •mmap.d

```
#!/usr/sbin/dtrace -s
```

```
syscall::mmap*:entry
```

```
/$target == pid/
```

```
{
```

```
    printf("%s: mapping %d bytes on fd: %x\n", probefunc, arg1,  
        arg4);
```

```
    jstack();
```

```
}
```

# mmap.d Output



```
maxb@manta # ./mmap.d -p 77649
dtrace: description 'syscall::mmap*:entry' matched 3 probes
CPU      ID                      FUNCTION:NAME
  2    11461                      mmap:entry mmap: mapping 2097152 bytes on fd: ffffffff

                                libc.so.1`mmap+0x15

node`_ZN2v88internal15MemoryAllocator20ReserveAlignedMemoryEjjPNS0_13VirtualMemoryE+0x2a

node`_ZN2v88internal15MemoryAllocator21AllocateAlignedMemoryEjjNS0_13ExecutabilityEPNS0_13
VirtualMemoryE+0x37
...
    << internal code >>
    << internal >>
    (anon) as (anonymous function) at vm.js position 3316
    << adaptor >>
    run at /leak2.js line 13
    (anon) as (anon) at /leak2.js line 27
    listOnTimeout at timers.js position 5196
    << adaptor >>
    << internal >>
    << entry >>

...
node`uv__run_timers+0x70
node`uv_run+0x85
node`_ZN4node5StartEiPPc+0x17b
```

# Taking a look at leak2.js



```
1
2  new Error().stack;
3  var
4  vm      = require("vm"),
5  total   = 1000000,
6  result  = null;
7
8  console.log("STARTING");
9  process.nextTick(function run() {
10
11      var script = vm.createScript('setInterval(function() {}, 0);', 'test.js');
12      var sandbox = { setInterval: setInterval, foo: {leak: "hello leaky world"} };
13      script.runInNewContext(sandbox);
14
15      total--;
16      if (total) {
17          /*      process.nextTick(run); */
18          setTimeout(function() {
19              var foo = {"bar": "hello world"};
20              run();
21              }, 1000);
22      } else {
23          console.log("COMPLETE");
24      }
25  });
```

# References



- <https://github.com/chrisa/node-dtrace-provider>
- <http://dtrace.org/blogs/blog/category/node-js/>
- <http://dtrace.org/blogs/bmc/2010/08/30/dtrace-node-js-and-the-robinson-projection/>
- <http://dtrace.org/blogs/dap/2012/01/05/where-does-your-node-program-spend-its-time/>
- <http://dtrace.org/blogs/brendan/2011/09/26/observing-observer-a-cloud-analytics-case-study/>

# Acknowledgements



- Thanks to nodeconfeu, Joyent Engineering (Bryan Cantrill, Mark Cavage, TJ Fontaine, Robert Mustacchi, Dave Pacheco, and others)
- Thanks for listening!
- Materials available at [git@github.com:max123/nodeconfeu.git](https://git@github.com:max123/nodeconfeu.git)
- [max@joyent.com](mailto:max@joyent.com), [@mrbruning](https://twitter.com/mrbruning)

