The History and Future of Core Dumps in FreeBSD

Sam W. Gwydir, Texas A&M University sam@samgwydir.com

March 25, 2017

Who Am I?

My name is Sam Gwydir

- Texas A&M University - Graduating in May
- Computer Engineering/Computer Science & Mathematics
- I've used *NIX for about 12 years
- OpenBSD, and later FreeBSD for about 4

Overview

- ~~Who Am I?~~
- Background
  - What is a Coredump?
  - Procedure
  - Tutorial
- The Past
  - Timelines
- The Present
  - FreeBSD
  - Illumos
  - Mac OS X
  - Backtrace.io
- The Future
  - Core Dump Extensions
    - Modular Dump Code
    - netdump
    - `minidumpsz`
    - Compressed Dump

What is a Core Dump?

Core Dump A machine readable form of the state of a machine at some point in time, usually after a `panic(9)`.

Useful for debugging both userspace programs and the kernel – we will focus on the kernel today.

Written using various methods to various over the years

General Dump Procedure (4.1 BSD - FreeBSD CURRENT)

- Started by a `panic(9)`, `reboot -d`
  - `sysctl debug.kdb.panic=1`
  - `dtrace -w -n 'BEGIN{ panic();}'`
- `dumpsys()` lands all/part of memory on swap in a particular format
- On reboot, `savecore(8)` writes dump to dumpdir for analysis

How to take a Core Dump in FreeBSD

- You are purposely panicking your machine.
- Do this in a VM!

```
root@:~ # sysrc dumpdev="AUTO" dumpdir="/var/crash"
root@:~ # mkdir /var/crash # create the dumpdir
root@:~ # chmod 700 /var/crash  # fix permissions
root@:~ # # for text dump: sysctl debug.ddb.textdump.pending=1
root@:~ # sysctl debug.kdb.panic=1
```

Overview

- ~~Who Am I?~~
- ~~Background~~
- The Past
  - Timelines
- The Present
  - FreeBSD
  - Illumos
  - Mac OS X
  - Backtrace.io
- The Future
  - Core Dump Extensions
    - Modular Dump Code
    - netdump
    - `minidumpsz`
    - Compressed Dump

The History

- The Odyssey of `doadump()`
- Starts at 6th Edition Research UNIX's `crash(8)`
- Ends at FreeBSD 12-CURRENT's Encrypted Dump
- Turn to the Appendix for a more in depth history
  - Includes architecture support
  - Feature changes and larger bug fixes
- For even more depth, go to the org-mode file on github.
  - Includes commits, mailing list emails, copious notes.

Core Dump Output Format Time Line

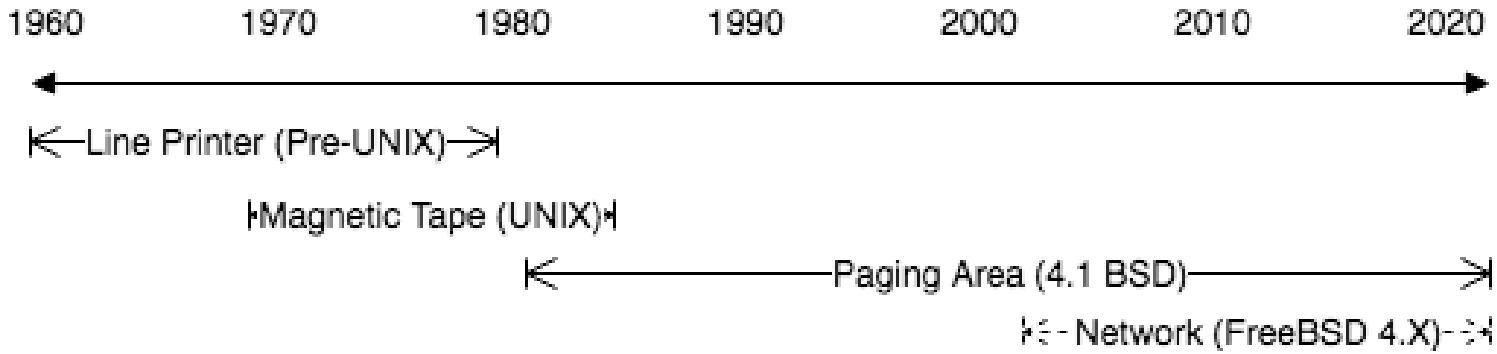

Figure 1: Core Dump Types

"Well in 1979 I can remember doing a crash dump on a Harris S/210 24 bit machine to the line printer in octal, it only took 2 hours to print. . . ." - rgrimes

FreeBSD Core Dump Extension

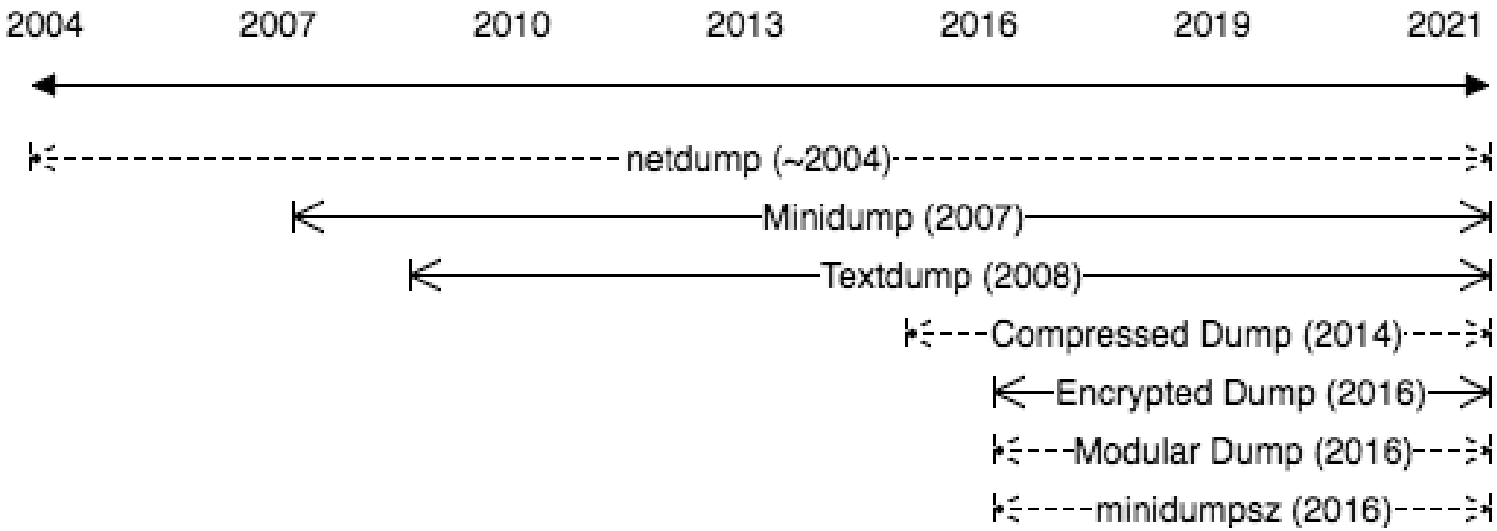

Figure 2: Core Dump Extension Timeline

Overview

- ~~Who Am I?~~
- ~~Background~~
- ~~The Past~~
- The Present
  - FreeBSD
  - Illumos
  - Mac OS X
  - Backtrace.io
- The Future
  - Core Dump Extensions
    - Modular Dump Code
    - netdump
    - `minidumpsz`
    - Compressed Dump

What is a Core Dump?

Two Types:

Full Dump Full contents of memory. Current procedure dates to 4.1 BSD. If you need help with your PDP-11 instructions are in the appendix.

Minidump Only active kernel pages are dumped. Added By Peter Wemm in FreeBSD 6.2. The default as of FreeBSD 7.0.

What is inside?

`info`	Metadata about dump (time, panic string, hostname)
`core.txt`	System info (backtrace, ps, vmstat, netstat, fstat)
`vmcore`	core itself

Full Dump On-Disk Format

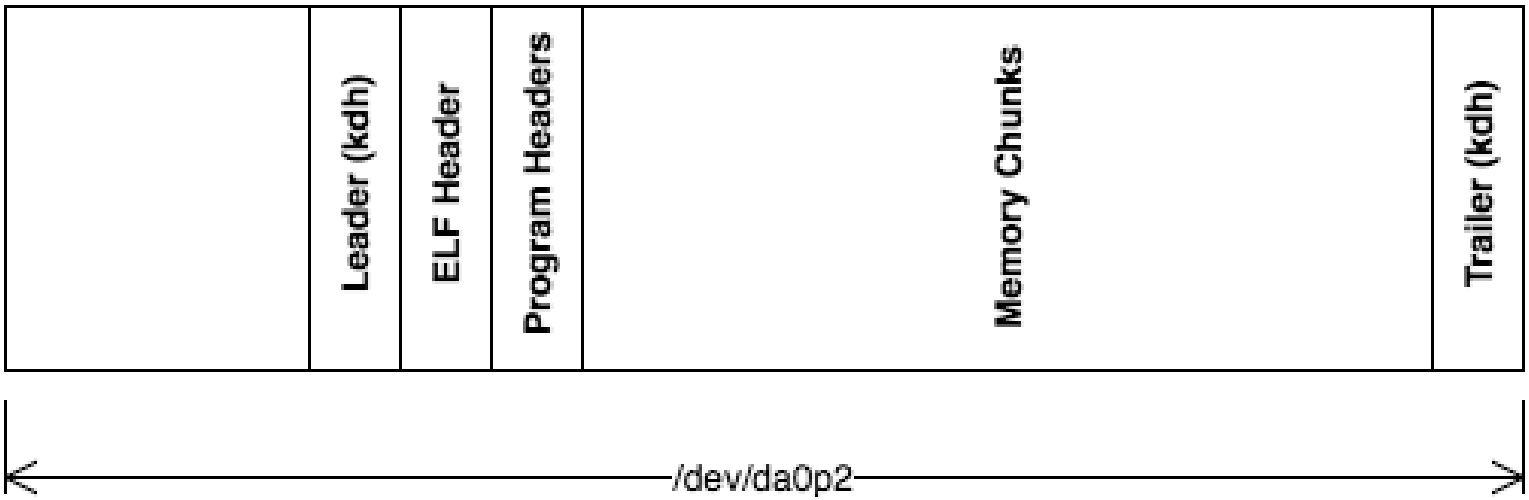

Figure 3: Full Dump Format

- Full Dump (FreeBSD 6.0)
  - A classic core dump – the full contents of memory at the time of a crash
  - ELF Format (a.out previous to FreeBSD 6.0)

Minidump On-Disk Format

Figure 4: Mini Dump Format

- Minidump (FreeBSD 6.2) - Peter Wemm
  - Contains only memory pages in use by kernel
  - Much smaller than the full contents of memory, modern dumps can still be fairly large
  - Custom “minidump” Format

What is a Text Dump?

`textdump(4)` "The textdump facility allows the capture of kernel debugging information to disk in a human-readable rather than the machine-readable form normally used with kernel memory dumps and minidumps."

Added by Robert Watson in FreeBSD 7.1.

What is inside?

- Other files are easily added with a small patch

`version.txt`	Kernel version string
`panic.txt`	Kernel panic message
`msgbuf.txt`	Kernel message buffer
`config.txt`	Kernel configuration
`ddb.txt`	Captured DDB output

Text Dump On-Disk Format

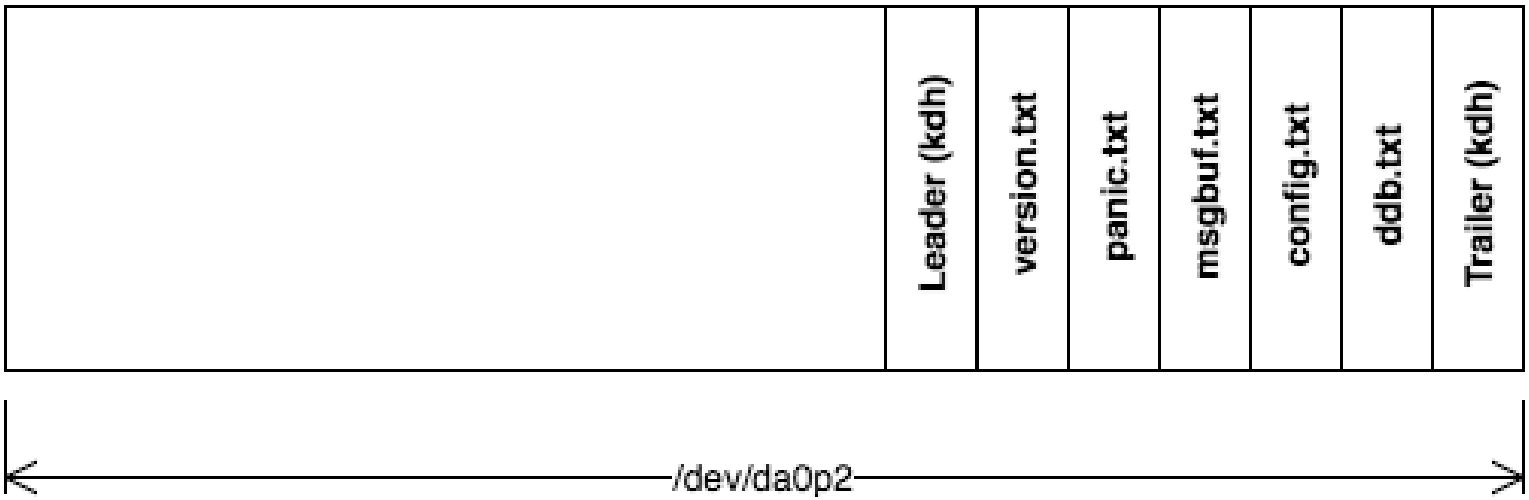

Figure 5: Text Dump Format

- Text Dump (FreeBSD 7.1) - Robert Watson
  - Custom ddb scripting in lieu of a dump
  - Written backwards because size is unknown a priori.
  - USTAR format

Core Dumps vs Textdumps

Both

- Useful when crashes aren't predicted i.e. production
- Operators can debug crashes offline
- Allows archiving of crash data for later comparison

Core Dumps

- Do not need to know what you are looking for ahead of time
- Need source tree, debug symbols and built kernel for analysis

Text Dumps

- Less Complete but much smaller (A few MB vs Many GB)
- Sometimes easier to extract information using DDB over kgdb.

Illumos

Not a BSD but the features are alluring

- Online dump size estimation
  - Includes different calculations for settings, e.g. compression
- Compressed Dump
  - gzip compression
- Dump to Swap on zvol
  - Versatility of zvols vs partitions
- Live Dump
  - Useful for production machines where interactive debugging is not possible
  - Especially for debugging hangs

Mac OS X

- Very different from the BSD dump procedure
  - Mach-O
  - Local or remote (network or Firewire)
- netdump - `kdumpd(8)`
  - Using a modified `tftpd(8)` from FreeBSD!
- Compressed Dump
  - gzip compression
  - Both local and using `kdumpd(8)`.
- Full Procedure in paper

Backtrace.io

- Backtrace.io curates kernel and userspace cores
- Snapshots allow for debugging on a laptop instead on crashed machine
  - Snapshots use automation to choose relevant sections of dump
- Allows for asking questions like:
  - Which panic is most common?
  - Correlated by datacenter? Storage Controller? Hard Drive Model? Timestamp?

Overview

- ~~Who Am I?~~
- ~~Background~~
- ~~The Past~~
- ~~The Present~~
- The Future
  - Core Dump Extensions
    - Modular Dump Code
    - netdump
    - `minidumpsz`
    - Compressed Dump

Extant Core Dump Extensions

- Modular Dump Code
  - Embedded systems
  - Mix and match features (compression but no netdump)
  - rgrimes@ for info
- netdump
  - Started at Duke by Darrell Anderson
  - Holding on since FreeBSD 4.x (~2004)
  - Continued at Sandvine and then Isilon
  - Almost part of FreeBSD 9.0
  - markj@ for info
- `minidumpsz` - minidump size estimation
- Compressed Dump

minidumpsz

```
% sudo sh minidumpsize_10.1.sh
debug.mini_dump_size: 138127282176 # 138.1 GB(!!!!)
```

- Dump Size Estimation
- A "no op" version of the minidump code as a kernel module
  - `minidumpsz` for FreeBSD 10 and 11
  - Should be upstreamed soon
  - Email for binary (rgrimes@ or sam@samgwydir.com)

Compressed Dump

- "Save Compression"
  - gzip dump on the fly before landing in swap
- Compression Ratio 6:1 to 14:1
- A 32 GB Core becomes 5.34 GB!
- Fixing the patch so it applies to FreeBSD 12 after encrypted dump will take some work

Proposed Core Dump Extensions

- Dump to swap on zvol
  - gibbs@ offered to mentor me
- Live Dump
  - Gauge interest

How to use the appendix for Research

- Use the org-mode file
  - Includes many of the commit messages, emails, and code referenced
    - Bonus email: jkh@ calling this topic esoteric :)
  - Includes information on versions not referenced in paper
    - UNIX v5 and other incomplete sections
  - Includes notes of various levels of detail
    - Code is often included where applicable

Links

- Thanks to
  - Deb Goodkin for bringing me into the FreeBSD Community
  - Rodney Grimes for help reading PDP-11 assembly
  - Michael Dexter for coming up with this idea and for asking me to thank him.
  - You for coming!
- github.com/gwydirsam/bsd-coredump-history
- github.com/dspinellis/unix-history-repo
- people.freebsd.org/~rgrimes/index.html#kerneldump