DEBUGGING LESSONS LEARNED WHILE FIXING NETBSD
ABOUT ME

maya@NetBSD.org

coypu@sdf.org

NetBSD/pkgsrc for the last 3 years
THIS TALK

Mix of a bunch of bugs

Not solo work

Thanks to riastradh, dholland, martin, kamil, many others
EARLY ATTEMPTS

checkout the source code

cvs -danoncvs@anoncvs.NetBSD.org:/cvsroot co src
./build.sh -U -u -O ~/obj -m amd64 tools kernel GENERIC

5-10 minutes round trip time to check

(so slow that I forget what I was testing)
TESTING IN STYLE

[desktop] <==[serial console, ethernet]==> [router]

Enable TFTP (desktop):
uncomment tftp line in /etc/inetd.conf, restart inetd
put kernels in /tftpboot

u-boot side (router):

set serverip=desktop.ip; set ipaddr=router.ip
tftp $loadaddr kernelname; bootm
set bootcmd=...

power reset = loads latest kernel from TFTP
round trip test time of 10 seconds
MIPS HANGS IN EARLY BOOT

serial console: can see last messages before it hangs

message that appears on console is a message printed by the source code. we can search for it.

The hang happens after the last print

```c
printf("%s:%d\n", __func__, __LINE__); everywhere
```
COMMANDS HANG WITH SOME CONNECTION TO MEMORY USAGE

SIGINFO, BSD favourite:

```
[ 510.5488859] load: 0.07  cmd: sleep 1357 [nanoslp] 0.00u 0.00s 0%
^ wchan
```

wchan appears in kernel source code

```
kern/kern_time.c
352:    error = kpause("nanoslp", true, timo, NULL);
```

sufficient to find relevant code!
Alternatively, ddb:

BREAK to enter (or whatever hw.cnmagic is set to)

```
crush> ps/l
    PID  LID S  CPU     FLAGS       STRUCT LWP   *               NAME WAI
       632  1  3   1   80   ffff81f7dbec8320              sleep nan

crush> bt/a ffff81f7dbec8320
trace: pid 632 lid 1 at 0xfffff8201393a6e50
sleepq_block() at sleepq_block+0x115
kpause() at kpause+0xed
nanosleep1() at nanosleep1+0xc6
sys___nanosleep50() at sys___nanosleep50+0x4a
syscall() at syscall+0x173
--- syscall (number 430) ---
79367043e6ba:
```
<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>useg</td>
<td>user memory, mapped</td>
</tr>
<tr>
<td>kseg0</td>
<td>kernel, unmapped</td>
</tr>
<tr>
<td>kseg1</td>
<td></td>
</tr>
<tr>
<td>kseg2</td>
<td>kernel virtual</td>
</tr>
</tbody>
</table>
SSH ON WIFI DOESN'T WORK?

ssh -vvv

ping -s [1,1000]
dmesg > before
ping -s 500 www.NetBSD.org
dmesg > after
diff -u before after | grep '^+'
bwfm_pci_intr_disable:2067
bwfm_pci_ring_rx:1377
bwfm_pci_ring_read_avail:1315
bwfm_pci_ring_update_wptr:1212
bwfm_pci_ring_rx:1377
bwfm_pci_ring_read_avail:1315
bwfm_pci_ring_update_wptr:1212
bwfm_pci_msg_rx:1406
bwfm_pci_pktid_free:993
bwfm_pci_ring_read_commit:1336
bwfm_pci_ring_write_rptr:1226
bwfm_pci_ring_rx:1377
bwfm_pci_ring_read_avail:1315
bwfm_pci_ring_update_wptr:1212
bwfm_pci_intr_enable:2056
bwfm_pci_intr:2023
configure:4671: checking minix/config.h usability
configure:4671: gcc -c -O2 -D_FORTIFY_SOURCE=2 -I/usr/include/krb5 conftest.c:55:26: fatal error: minix/config.h: No such file or directory
  
#include <minix/config.h>

^ compilation terminated.
configure:4671: $? = 1
configure: failed program was:
| #include <minix/config.h>
double rounding_alpha_simple_even = 9223372036854775808.000000; /* 
uint64_t unsigned_even = rounding_alpha_simple_even; 
assert(unsigned_even % 2 == 0);

surely that's a compiler bug...

GCC alpha person: can't reproduce on linux
-mfp-trap-mode=sui

| cvttq/svic | $f10, $f11 |
| cvttq/svc  | $f10, $f11 |
VAX FLOAT

no infinity

no NaN

no subnormals

traps instead
GETTING GRAPHICS: NIGHTMARE

SETUP

No network booting

Monitor becomes black

options DDB_COMMANDONENTER="bt; reboot"

Fortunately, reboot saves dmesg buffer
"MUTEX IS NOT INITIALIZED"

[initialization]  ->  [use]
BUG IN INITIALIZATION?

db_stacktrace();

print the memory allocated at initialization and use
can confirm all callers are allocate correctly
worst bug: can see the effect, not the cause
13TH ALLOCATION IS THE OFFENDING ONE

What can we do with this?

```
static int i = 0;
++i;
if (i == 13) {
    /* do something to offending allocation */
}
```

Put a debug register on the 13th allocation
Nothing goes well- didn't get backtrace from DDB_COMMANDONENTER

fatal page fault in supervisor mode
trap type 6 code 0 rip 0xfffffffff8077d472 cs 0x8
rflags 0x10286 cr2 0x18 ilevel 0 rsp 0xfffff8b0139de6e30
curlwp 0xfffff882ade2f7b20 pid 19253.648 lowest kstack 0xfffff8b0139de

gdb> disas 0xfffffffff8077d472 ---> kmem_free
Still know it's the 13th allocation

```c
if (i == 13) {
    corrupted_start = allocation
    corrupted_size = size;
}

kmem_free(...) {
    if (initialized_memory;)
        if (memory in [allocation, allocation+size])
            db_stacktrace();
            panic("corrupting range!");
}
```
## MIPS BASICS

<table>
<thead>
<tr>
<th>Registers</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a0-a3</td>
<td>Function input</td>
</tr>
<tr>
<td>v0-v1</td>
<td>Function output</td>
</tr>
<tr>
<td>s0-s9</td>
<td>Local registers (can't trash)</td>
</tr>
<tr>
<td>t0-t9</td>
<td>Local registers (can trash)</td>
</tr>
</tbody>
</table>
assembler: "No .cprestore pseudo-op used in PIC code"

JaegerTrampoline:
- lui $28,%hi(_gp_disp)
- addiu $28,$28,%lo(_gp_disp)
- addu $28,$28,$25
+ .cpload $25
PIC code

<table>
<thead>
<tr>
<th>Executable</th>
<th>Fixed memory 0x80000...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Library A</td>
<td>???</td>
</tr>
<tr>
<td>Library B</td>
<td>???</td>
</tr>
</tbody>
</table>

All the code can't assume fixed memory
x86, others: code can just use PC-relative addressing
MIPS: not so easy, dedicate a register: GP
"WOW, THAT'S INEFFICIENT"

MIPS is an ABI clusterfuck

netbsd/mips64

• n64 kernel
• default n32 userland
• can run o32, n32, n64
Want to run o32 code

(code written when MIPS was more popular)
a0-a3 to pass arguments
if they're 32bit, how to pass 64bit argument?
How to pass very many arguments?
syscall ABI compat:

- syscall table is auto-generated
- sy_flags says which argument is 64bit
- combine the result from two registers to match calling convention