Agenda

• Introduction
• Kernel Oops
• Hangs, Magic SysRq
• Demos, Examples
Introduction

• Classification of bugs
  • Kernel Oops
  • Kernel Panics
  • Lockups a.k.a hangs (Soft/Hard)
  • Unexpected behavior
Introduction

• Steps involved in fixing bugs
  • Localizing the bug
  • Understanding what's wrong
  • Fixing bug
Oops - defined

- Triggered when kernel detects serious abnormal conditions
- Dumps useful debug information
  - CPU state, Kernel stack etc.
- Tries to prevent kernel going out of control or causing data corruption
- How to Capture Oops
It looks like this..

Unable to handle kernel paging request at virtual address ffffffffb  
printing eip: c104c406
*pde = 00002067
Oops: 0000 [#1]
last sysfs file: /class/vc/vcsa7/dev
Modules linked in: nfs lockd nfs_acl rpcsec_gss_krb5 auth_rpcgss sunrpc ...
CPU: 0
EIP: 0060:[<c104c406>]  Tainted: G U VLI
EFLAGS: 00010246 (2.6.15-kdb-smp #1)
EIP is at nameidata_to_filp+0x9/0x2e
eax: d83b5f30   ebx: fffffff3   ecx: 00008441   edx: 00008441
esi: d83b5f30   edi: b7f1fff4   ebp: d83b5f28   esp: d83b5f24
ds: 007b   es: 007b   ss: 0068
Process bash (pid: 14600, threadinfo=d83b4000 task=d26ea030)
Stack: <0>00008441 d83b5f88 c104c45a d8303c94 dff6fa40 cd043ee3 00000006
d49b5008 00003000 00000000 00000000 00000002 df48c900 00001000 00001000
00000003 0000000d 00008000 00000003 d2af3dc0 00008442 000001b6 fffffff3
00008441
(Contd.)
It looks like this.. (Contd.)

Call Trace:

[<c100407b>] show_stack_log_lvl+0xaa/0xb5
[<c100419d>] show_registers+0x117/0x17d
[<c1004470>] die+0x12e/0x1ad
[<c117c97d>] do_page_fault+0x38b/0x547
[<c1003bcf>] error_code+0x4f/0x60
[<c104c45a>] do_filp_open+0x2f/0x36
[<c104c545>] do_sys_open+0x3f/0xb8
[<c104c5ea>] sys_open+0x16/0x18
[<c10029eb>] sysenter_past_esp+0x54/0x79

Code: 60 01 00 b8 e9 ff ff ff eb 10 6a 00 89 f1 50 89 da 89 f8 e8 33 fe ff ff 5e 5f 8d 65 f4 5b 5e 5f 5d c3 55 89 d1 89 e5 53 8b 58 4c <83> 7b 08 00 75 13 8b 50 04 6a 00 53 8b 00 e8 0d fe ff ff 89 c3
InterpretingOops

• Fault
• EIP = function base address + instruction offset
• Oops counter, No. of CPUs
• EFLAGS
• Registers (general purpose, segment, control registers)
• Call trace – return addresses
Approach

- Get EIP from Oops output
- Find the nearest matching address (less than EIP) and corresponding function in System.map
- Disassemble using objdump
- Match the function offset to the instruction
- Match the assembler instructions to the C source
- Identity the issue
- Fix it and Be Happy! :)

© Novell Inc, Confidential & Proprietary
Kernel Oops - Demo
Points to note

• Don't trust Oopsed kernel
• Frame pointers support – better Stack tracebacks
• Always check syslog in case of strange behavior
• Linus Torvalds says:
  • “I’m afraid that I’ve seen too many people fix bugs by looking at debugger output, and that almost inevitably leads to fixing the symptoms rather than the underlying problems.”
  • “Use the source Luke”
Lockups

• System just freezes, no messages, no responses

• Types
  • Lockups with interrupts enabled
  • Lockups without interrupts enabled
    – NMI watchdog
  • Hardware lockups
    – Mostly due to hardware problem
    – Hardware abuse because of poorly written driver
Lockups with interrupts enabled

• Common reasons
  • Spinning in a loop
  • Waiting on a lock
  • Deadlocks

• Symptoms
  • Toggle keyboard lights (Caps-lock, Scroll lock keys will blink)
  • Machine will react to pings
  • Keyboard inputs may/may not be echoed
  • But, Process won't progress
Using Magic SysRq (System Request)

• Simple keystrokes which allow commands to be sent directly to the Kernel
• Kernel support for Magic SysRq
• How to – Select Magic System Request (SysRq) keys under “Kernel Hacking” Menu of Kernel Configuration
• Enable temporarily – as root enter
  • echo 1 > /proc/sys/kernel/sysrq
• Permanently using Yast
Magic SysRq (contd.)

• Keys - Alt + SysRq + <key> on i386
• Output to /var/log/messages
• Frequently used keys
  • ’p’ - Will dump the current registers and flags to your console
  • ’t’ - Will dump a list of current tasks and their information to your console
  • ’m’ - Will dump current memory info to your console.
  • Other keys

• Other keys
  • ’h’ - The most important key - will display help
  • ’b’ - Will immediately reboot the system without syncing or unmounting your disks
Magic SysRq (Contd.)

- 'o' - Will shut your system off (if configured and supported)
- 'u' - Will attempt to remount all mounted file systems read-only.
- 'e' - Send a SIGTERM to all processes, except for init.
- 'i' - Send a SIGKILL to all processes, except for init.
- 'I' - Send a SIGKILL to all processes, INCLUDING init. (Your system will be non-functional after this.)
- '0'-'9' - Sets the console log level, controlling which kernel messages will be printed to your console.
SysRq + p

SysRq : Show Regs (SysRq + p)

Pid: 2894, comm: X
EIP: 0060:<c020a7a2> CPU: 0
EIP is at read_chan+0x5/0x5b1
EFLAGS: 00003282 Not tainted (2.6.15-kdb-smp)
EAX: dd885000 EBX: dd88500c ECX: bf9416dc EDX: d94a4d80
ESI: dd885000 EDI: c020a79d EBP: dbdc5f44 DS: 007b ES: 007b
CR0: 80050033 CR2: b6bb4108 CR3: 1fd79000 CR4: 000006d0
[c01023c5>] show_regs+0x10a/0x115
[c021467e>] sysrq_handle_showregs+0xe/0x10
[c02147f7>] __handle_sysrq+0x7a/0xf1
[c0214881>] handle_sysrq+0x13/0x16
[c0210040>] kbd_keycode+0x131/0x2f6
[c021027e>] kbd_event+0x79/0xa7
SysRq + p

[c022b08e>] input_event+0x3d6/0x3f9
[c022e3a3>] atkbd_report_key+0x5e/0x7e
[c022e7d0>] atkbd_interrupt+0x40d/0x4dd
[c02187c0>] serio_interrupt+0x35/0x6e
[c02191d0>] i8042_interrupt+0x1d8/0x1ea
[c0140727>] handle_IRQ_event+0x27/0x52
[c01407df>] __do_IRQ+0x8d/0xe2
[c01062b9>] do_IRQ+0x49/0x5a
[c0104e5a>] common_interrupt+0x1a/0x20
[c020648a>] tty_read+0x63/0xb3
[c015bf3c>] vfs_read+0xac/0x15b
[c015c262>] sys_read+0x3b/0x60
[c0103d9b>] sysenter_past_esp+0x54/0x79
Example

• Running connectathon tests from multiple (180) clients, Client process hung.
  • Process in D (uninterruptible) State in kernel mode
How to debug

• Use Magic SysRq (or)
• ps n -o pid,user,wchan -C <process>
• ps -o pid,user,wchan -C <process> will translate the address (EIP) into corresponding function
• If address found in System.map, use /proc/kallsyms or disassemble module with starting address
Questions?
Novell
Unpublished Work of Novell, Inc. All Rights Reserved.
This work is an unpublished work and contains confidential, proprietary, and trade secret information of Novell, Inc. Access to this work is restricted to Novell employees who have a need to know to perform tasks within the scope of their assignments. No part of this work may be practiced, performed, copied, distributed, revised, modified, translated, abridged, condensed, expanded, collected, or adapted without the prior written consent of Novell, Inc. Any use or exploitation of this work without authorization could subject the perpetrator to criminal and civil liability.

General Disclaimer
This document is not to be construed as a promise by any participating company to develop, deliver, or market a product. Novell, Inc., makes no representations or warranties with respect to the contents of this document, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. Further, Novell, Inc., reserves the right to revise this document and to make changes to its content, at any time, without obligation to notify any person or entity of such revisions or changes. All Novell marks referenced in this presentation are trademarks or registered trademarks of Novell, Inc. in the United States and other countries. All third-party trademarks are the property of their respective owners.