Operating System Design and Implementation

Getting started with kernel and kernel debugging

Charles Tsao

Outline

- How to develop an operating system
  - hardware, compile, assembler, linker
- How to compile kernel codes
  - make
- How to debug kernel codes
  - gdb, kgdb, ICE
- How to maintain kernel codes
  - Version control, CVS, SVN, GIT
  - patch
- A case study of Linux

Hardware

- How to develop an operating system for a new processor
  - Simulator vs. emulator vs. virtual machine
- Simulator
  - A program to reproduce the behavior of a computer system based on an abstract model
- Emulator
  - Hardware or software or both that duplicates (or emulates) the functions of one computer system (the guest) in another computer system (the host), different from the first one, so that the emulated behavior closely resembles the behavior of the real system (the guest)[1]
- Virtual machine
  - A virtual machine (VM) is a software implementation of a machine (i.e. a computer) that executes programs like a physical machine[2]

[1,2] *http://en.wikipedia.org/*

Compiler

- Compiler vs. cross compiler
  – How to develop a compiler for new processor?
- Assembler
  – How to develop an assembler?
- Linker
  – Why?

Linking

code/link/main.c

```c
/* main.c */
void swap();

int buf[2] = {1, 2};

int main()
{
    swap();
    return 0;
}
```

code/link/main.c

(a) `main.c`

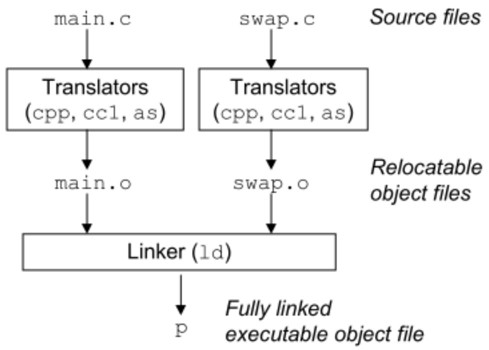
code/link/swap.c

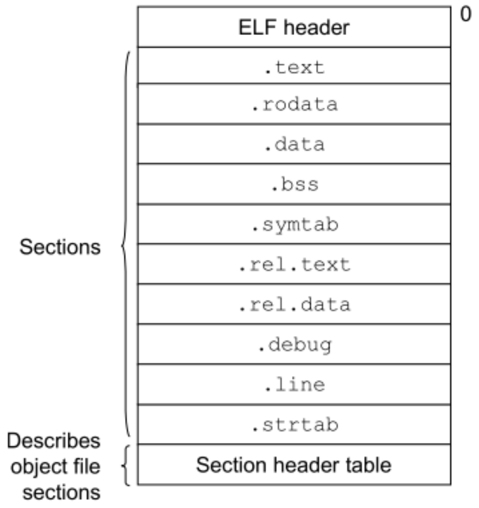
```c
/* swap.c */
extern int buf[];

int *bufp0 = &buf[0];
int *bufp1;

void swap()
{
    int temp;

    bufp1 = &buf[1];
    temp = *bufp0;
    *bufp0 = *bufp1;
    *bufp1 = temp;
}
```

code/link/swap.c

(b) `swap.c`

Linking

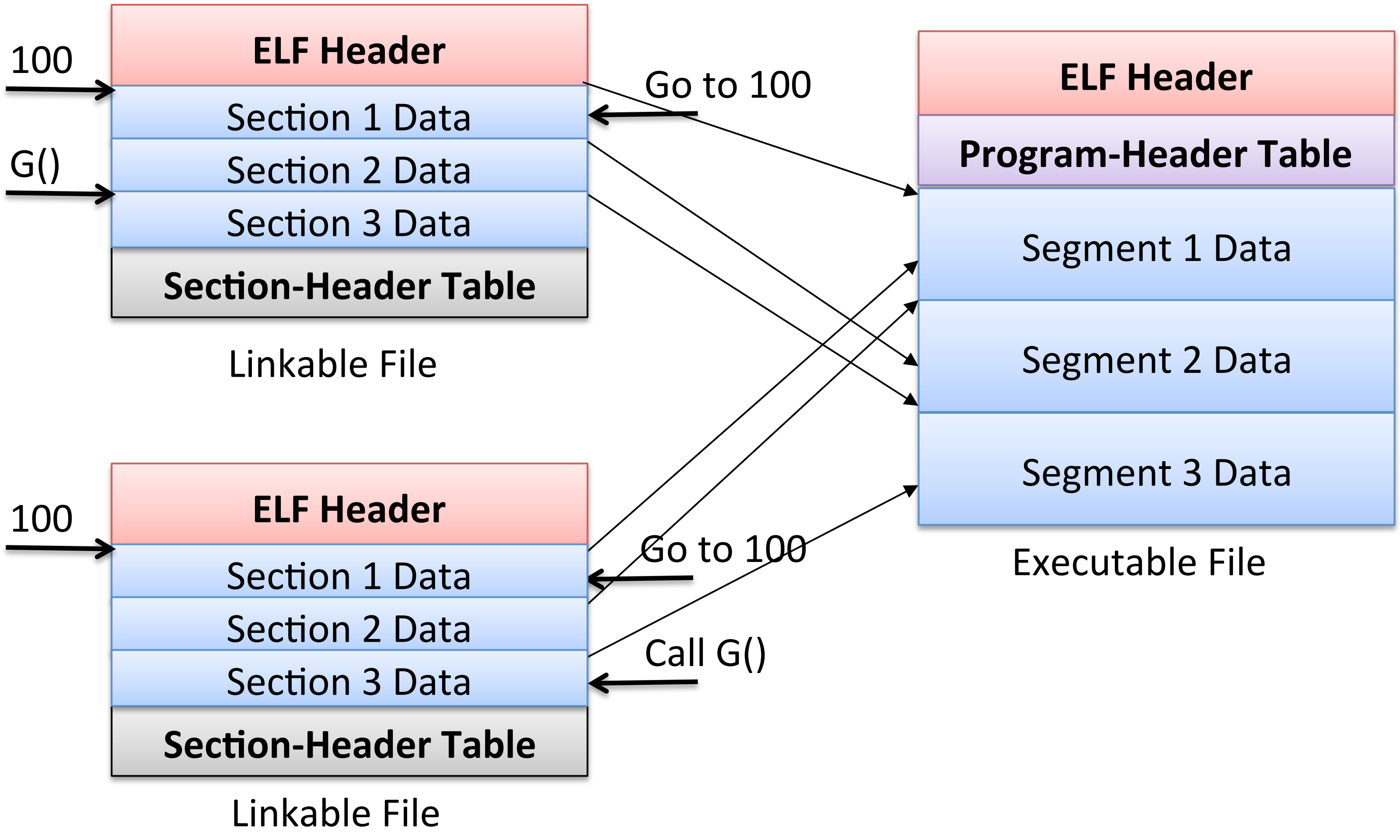

main.c
swap.c
Source files
Translators (cpp, cc1, as)
Translators (cpp, cc1, as)
main.o
swap.o
Relocatable object files
Linker (ld)
p
Fully linked executable object file



```
Num:    Value  Size Type    Bind   Ot  Ndx Name
  8:        0     8 OBJECT  GLOBAL  0    3 buf
  9:        0    17 FUNC    GLOBAL  0    1 main
 10:        0     0 NOTYPE  GLOBAL  0  UND swap
```

Computer Systems: A Programmer's Perspective, 2/E (CS:APP2e)
Randal E. Bryant and David R. O'Hallaron, Carnegie Mellon University

Linking



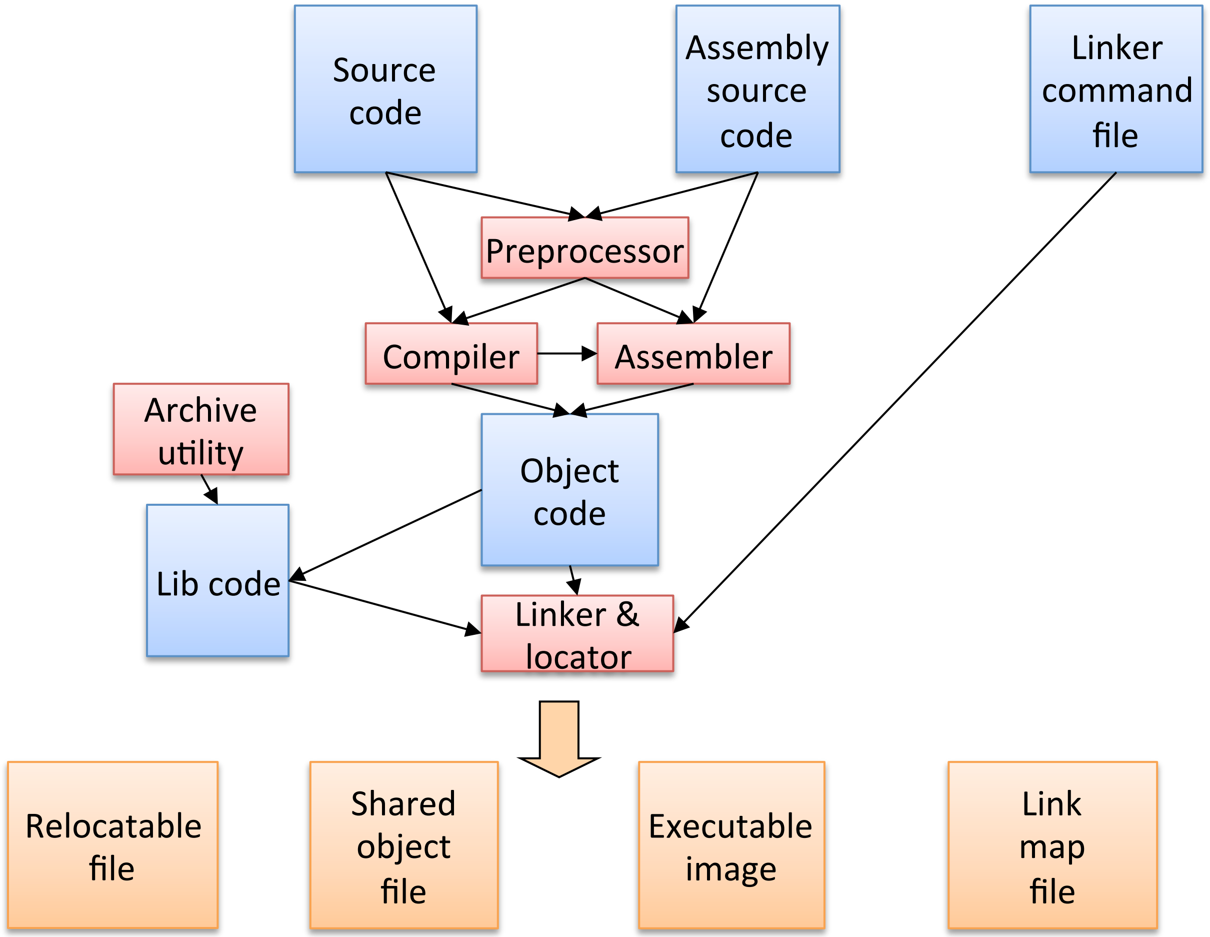
100
ELF Header
Section 1 Data
G()
Section 2 Data
Section 3 Data
Section-Header Table
Linkable File
Go to 100
ELF Header
Program-Header Table
Segment 1 Data
Segment 2 Data
Segment 3 Data
Executable File
100
ELF Header
Section 1 Data
Go to 100
Section 2 Data
Section 3 Data
Call G()
Section-Header Table
Linkable File

Linking

- Please review system programming and compiler if you are not familiar with below terms
  - Static linking
  - Dynamic linking
  - Relocations
  - Symbol table
  - Share library
  - Linking and loading

How to compile kernel codes

- Creating an executable image



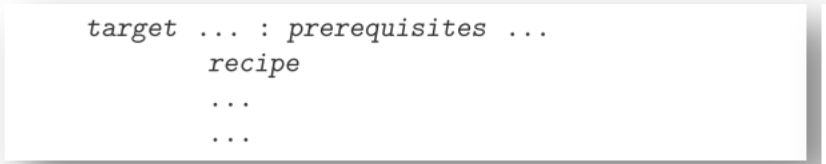
How to compile kernel codes

- 17,090 – The number of files in Linux 2.6.11
- 37,626 – The number of files in Linux 3.2
- How can I find and compile my network interface card driver among 100 network interface card drivers?
- Shall I recompile again when I modify one file?
- How can we produce kernel image ?

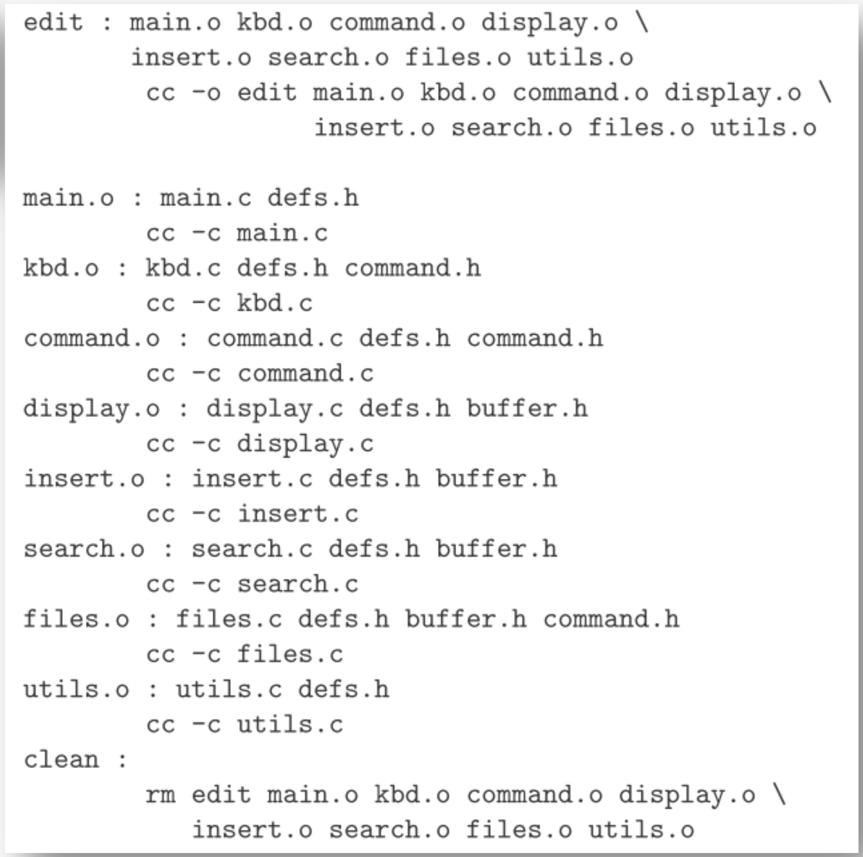
Make and Makefile

- Make: utility to provide a convenient facility to build, install, and uninstall projects
- Makefile: script file for make to compile and link programs

Make and Makefile



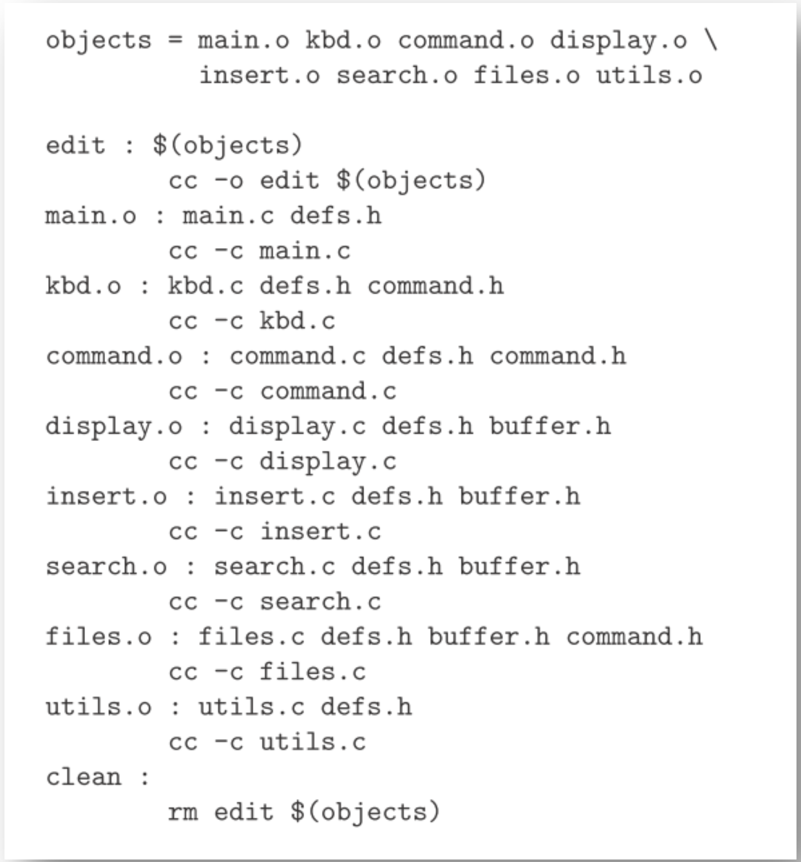
```
     target ... : prerequisites ...
             recipe
             ...
             ...
```



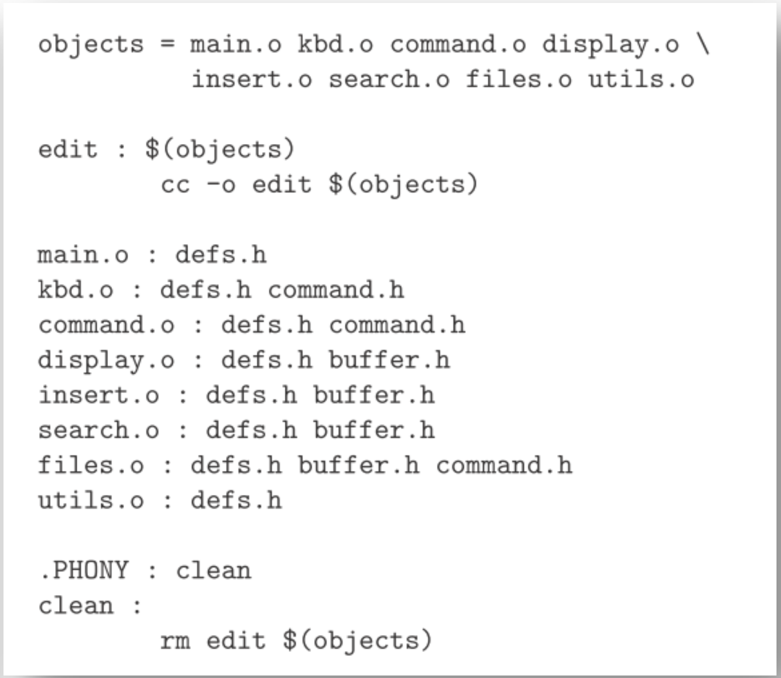
```
edit : main.o kbd.o command.o display.o \
       insert.o search.o files.o utils.o
        cc -o edit main.o kbd.o command.o display.o \
                   insert.o search.o files.o utils.o

main.o : main.c defs.h
        cc -c main.c
kbd.o : kbd.c defs.h command.h
        cc -c kbd.c
command.o : command.c defs.h command.h
        cc -c command.c
display.o : display.c defs.h buffer.h
        cc -c display.c
insert.o : insert.c defs.h buffer.h
        cc -c insert.c
search.o : search.c defs.h buffer.h
        cc -c search.c
files.o : files.c defs.h buffer.h command.h
        cc -c files.c
utils.o : utils.c defs.h
        cc -c utils.c
clean :
        rm edit main.o kbd.o command.o display.o \
           insert.o search.o files.o utils.o
```

http://www.gnu.org/software/make/manual/make.pdf

Make and Makefile



```
objects = main.o kbd.o command.o display.o \
          insert.o search.o files.o utils.o

edit : $(objects)
        cc -o edit $(objects)
main.o : main.c defs.h
        cc -c main.c
kbd.o : kbd.c defs.h command.h
        cc -c kbd.c
command.o : command.c defs.h command.h
        cc -c command.c
display.o : display.c defs.h buffer.h
        cc -c display.c
insert.o : insert.c defs.h buffer.h
        cc -c insert.c
search.o : search.c defs.h buffer.h
        cc -c search.c
files.o : files.c defs.h buffer.h command.h
        cc -c files.c
utils.o : utils.c defs.h
        cc -c utils.c
clean :
        rm edit $(objects)
```



```
objects = main.o kbd.o command.o display.o \
          insert.o search.o files.o utils.o

edit : $(objects)
        cc -o edit $(objects)

main.o : defs.h
kbd.o : defs.h command.h
command.o : defs.h command.h
display.o : defs.h buffer.h
insert.o : defs.h buffer.h
search.o : defs.h buffer.h
files.o : defs.h buffer.h command.h
utils.o : defs.h

.PHONY : clean
clean :
        rm edit $(objects)
```

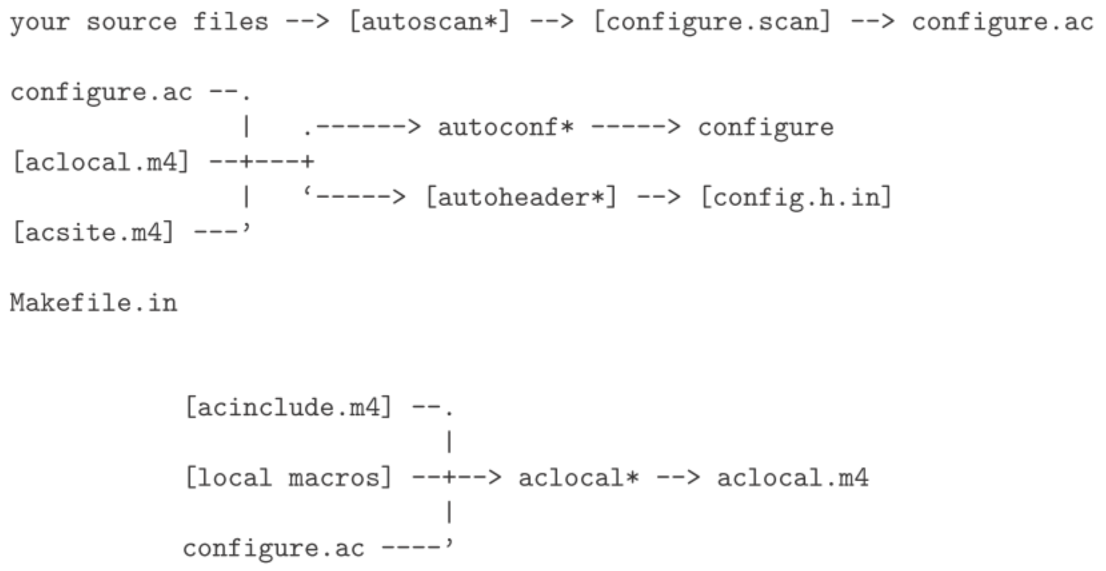
http://www.gnu.org/software/make/manual/make.pdf

Make and Makefile

- Variables and settings
  - make config
  - make menuconfig
- Phony targets
  - make all
  - make clean
  - make depend
  - make install
  - make uninstall

The portability problem

- Hardware differences
- OS differences
- Compiler differences



```
your source files --> [autoscan*] --> [configure.scan] --> configure.ac

configure.ac --.
               |   .------> autoconf* -----> configure
[aclocal.m4] --+---+
               |   `-----> [autoheader*] --> [config.h.in]
[acsite.m4] ---'

Makefile.in


             [acinclude.m4] --.
                              |
             [local macros] --+--> aclocal* --> aclocal.m4
                              |
             configure.ac ----'
```

Autoconf Manual - The GNU Operating System

The portability problem

```
          configure.ac --.
                         +--> automake* --> Makefile.in
          Makefile.am ---'


                        .-------------> [config.cache]
configure* ------------+-------------> config.log
                        |
[config.h.in] -.        v            .-> [config.h] -.
               +--> config.status* -+                +--> make*
Makefile.in ---'                     `-> Makefile ---'
```

Autoconf Manual - The GNU Operating System

How to debug kernel codes

- Kernel logs (discontinues logs)
  - Printk
  - Oops and Kallsyms
- Kernel debug supports
  - Kexec, kdump, SysRq
- Kernel hacking options
- Kernel debug tools
  - gdb, kgdb, kdb
- Profile
  - OProfile
- Trace
  - KFT, LTT/LTTng
  - Gprof
- Lock detection
  - Lockmeter
- Memory leaking
- Test equipment

Debugging and profiling device drivers

- printk()
  - Loglevels

Loglevel	Description
KERN_EMERG	An emergency condition; the system is probably dead
KERN_ALERT	A problem that requires immediate attention
KERN_CRIT	A critical condition
KERN_ERR	An error
KERN_WARNING	A warning
KERN_NOTICE	A normal, but perhaps noteworthy, condition
KERN_INFO	An informational message
KERN_DEBUG	A debug messagetypically superfluous

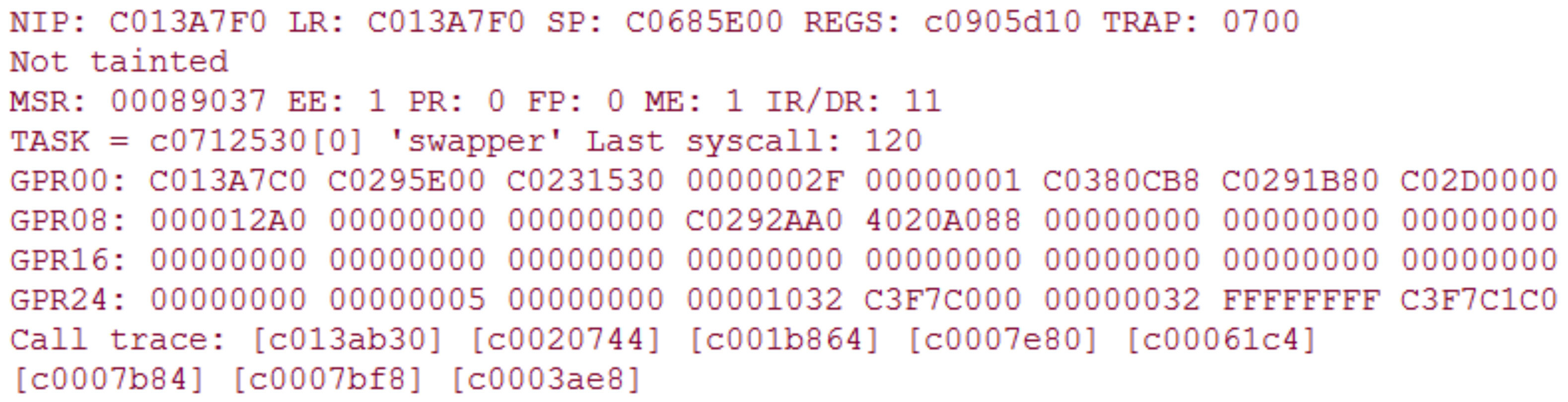
printk()

- Log buffer
- Klogd
  - /proc/kmsg or syslog()
- Syslogd
  - Appeds all the messages it receives to a file
  - /var/log/messages
  - /etc/syslog.conf
- Not for booting stage debugger (early_printk())
- Not for debugging non-console case (via serial port)
- Not easy to detect racing condition

Kernel debugging options

- Turn on in kernel hacking/linuxconfig
- BUG()/ BUG_ON() case oops (stack trace, error message dump to kernel)
- Panic() prints error messages and halts the kernel
- dump_stack()

ksymoops



```
NIP: C013A7F0 LR: C013A7F0 SP: C0685E00 REGS: c0905d10 TRAP: 0700
Not tainted
MSR: 00089037 EE: 1 PR: 0 FP: 0 ME: 1 IR/DR: 11
TASK = c0712530[0] 'swapper' Last syscall: 120
GPR00: C013A7C0 C0295E00 C0231530 0000002F 00000001 C0380CB8 C0291B80 C02D0000
GPR08: 000012A0 00000000 00000000 C0292AA0 4020A088 00000000 00000000 00000000
GPR16: 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
GPR24: 00000000 00000005 00000000 00001032 C3F7C000 00000032 FFFFFFFF C3F7C1C0
Call trace: [c013ab30] [c0020744] [c001b864] [c0007e80] [c00061c4]
[c0007b84] [c0007bf8] [c0003ae8]
```

- Ksymoops + system.map+module information
- Linux 2.6.X uses kallsyms

ksymoops

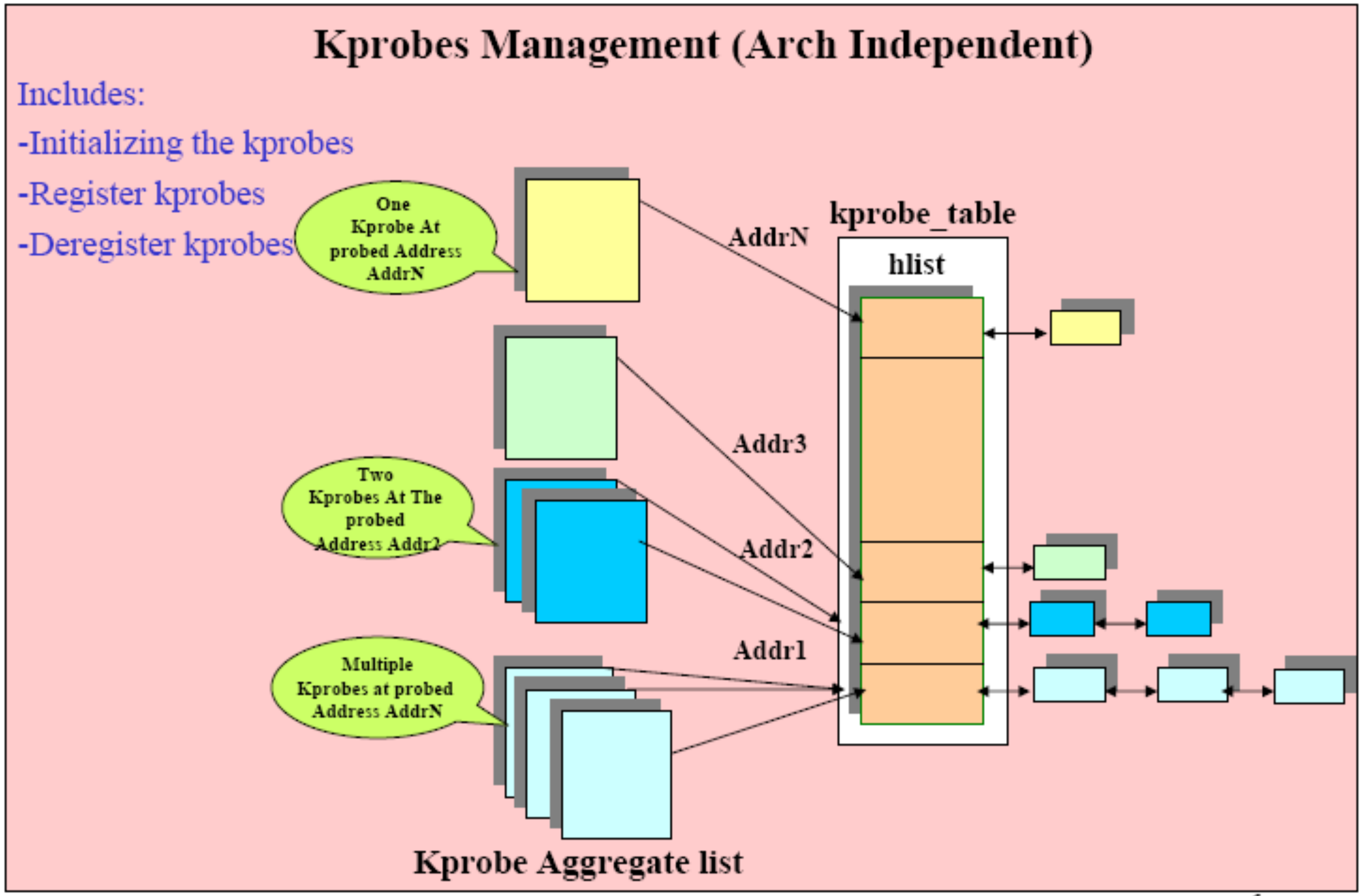
```
Oops: Exception in kernel mode, sig: 4
Unable to handle kernel NULL pointer dereference at virtual address 00000001

NIP: C013A7F0 LR: C013A7F0 SP: C0685E00 REGS: c0905d10 TRAP: 0700
Not tainted
MSR: 00089037 EE: 1 PR: 0 FP: 0 ME: 1 IR/DR: 11
TASK = c0712530[0] 'swapper' Last syscall: 120
GPR00: C013A7C0 C0295E00 C0231530 0000002F 00000001 C0380CB8 C0291B80 C02D0000
GPR08: 000012A0 00000000 00000000 C0292AA0 4020A088 00000000 00000000 00000000
GPR16: 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
GPR24: 00000000 00000005 00000000 00001032 C3F7C000 00000032 FFFFFFFF C3F7C1C0
Call trace:
[c013ab30] tulip_timer+0x128/0x1c4
[c0020744] run_timer_softirq+0x10c/0x164
[c001b864] do_softirq+0x88/0x104
[c0007e80] timer_interrupt+0x284/0x298
[c00033c4] ret_from_except+0x0/0x34
[c0007b84] default_idle+0x20/0x60
[c0007bf8] cpu_idle+0x34/0x38
[c0003ae8] rest_init+0x24/0x34
```

SysRq

Key Command	**Description**
SysRq-b	Reboot the machine
SysRq-e	Send a SIGTERM to all processes except init
SysRq-h	Display SysRq help on the console
SysRq-i	Send a SIGKILL to all processes except init
SysRq-k	Secure Access Key: kill all programs on this console
SysRq-l	Send a SIGKILL to all processes including init
SysRq-m	Dump memory information to console
SysRq-o	Shut down the machine
SysRq-p	Dump registers to console
SysRq-r	Turn off keyboard raw mode
SysRq-s	Sync all mounted file systems to disk
SysRq-t	Dump task information to console
SysRq-u	Unmount all mounted file systems

Kprobes

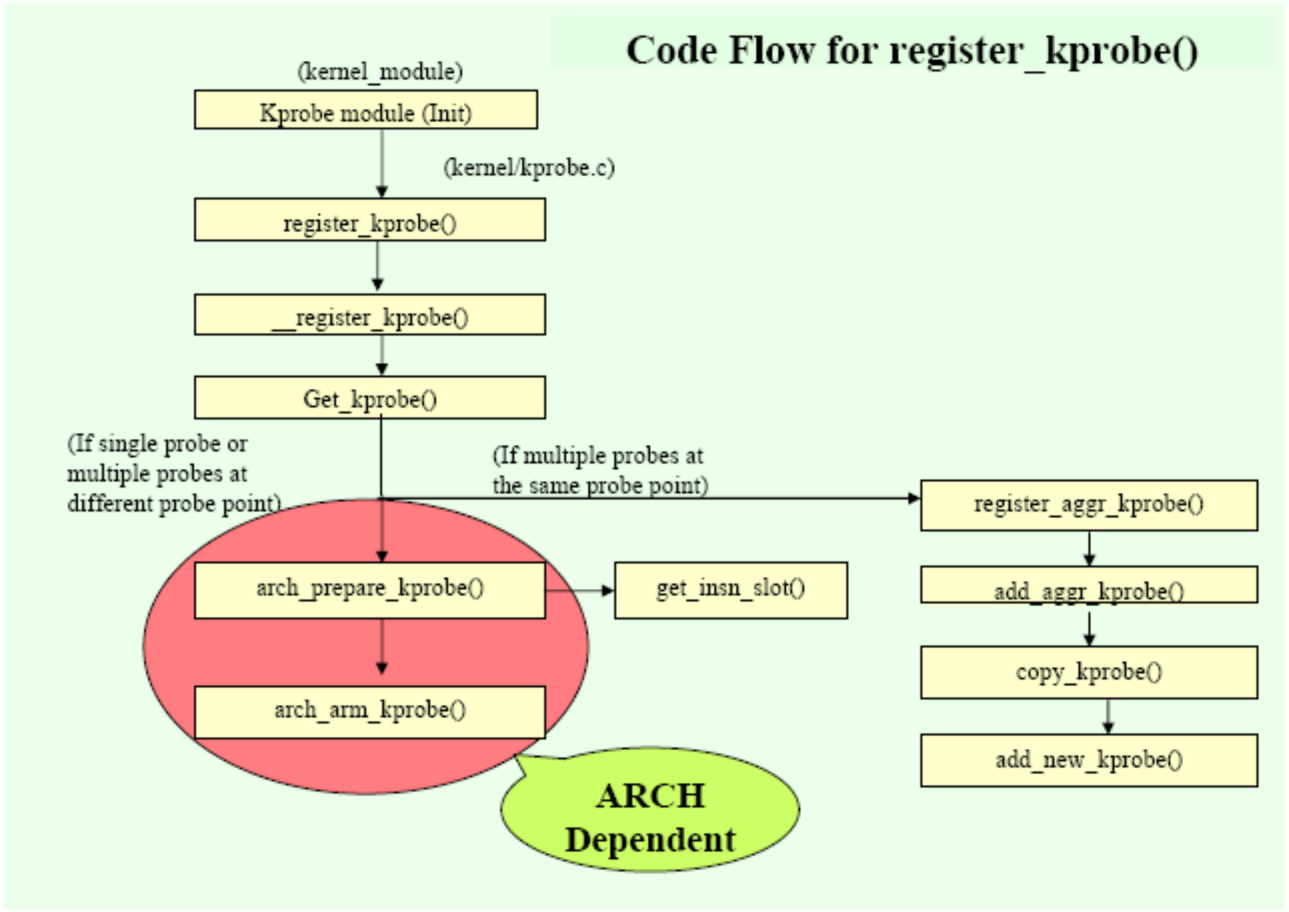
- Turn on CONFIG_KPROBES (Instrumentation Support Kprobes) in the kernel configuration menu



Kprobes Management (Arch Independent)
Includes:
-Initializing the kprobes
-Register kprobes
-Deregister kprobes
One Kprobe At probed Address AddrN
Two Kprobes At The probed Address Addr2
Multiple Kprobes at probed Address AddrN
AddrN
Addr3
Addr2
Addr1
kprobe_table
hlist
Kprobe Aggregate list

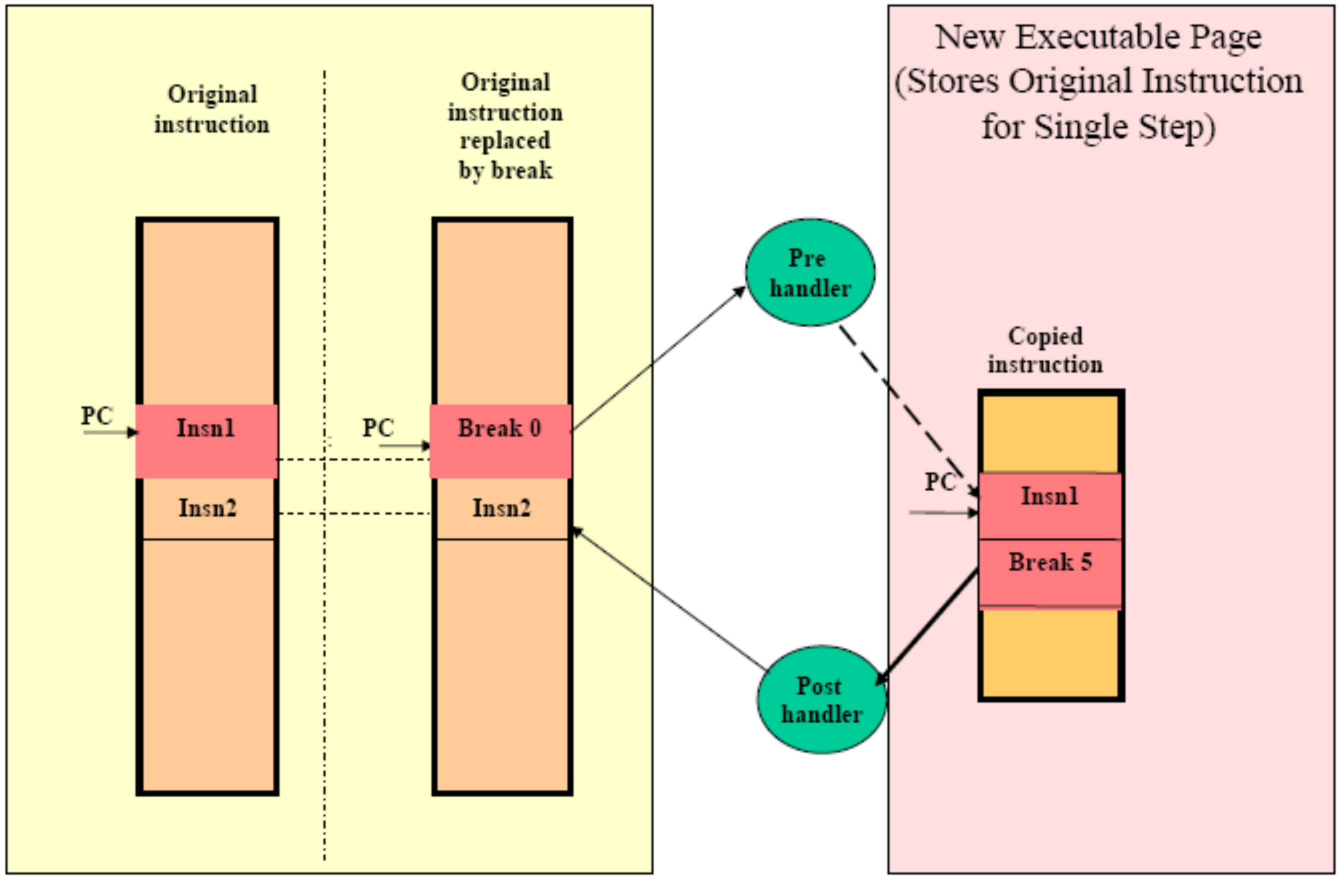
Source: Lubna, Vikas, Madhvesh. "Kprobes Support for MIPS"

Kprobes



Source: Lubna, Vikas, Madhvesh. "Kprobes Support for MIPS"

Kprobes


Original instruction
Original instruction replaced by break
New Executable Page (Stores Original Instruction for Single Step)
PC
Insn1
Insn2
PC
Break 0
Insn2
Pre handler
Post handler
Copied instruction
PC
Insn1
Break 5

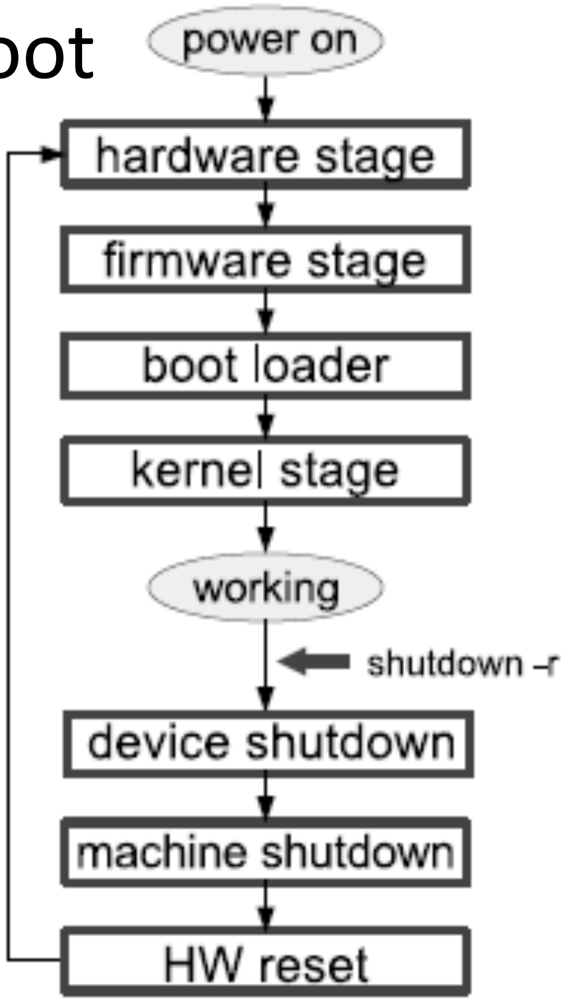
Source: Lubna, Vikas, Madhvesh. "Kprobes Support for MIPS"

Kexec and Kdump

- Kexec uses exec() to spawn a new kernel over a running kernel without the overhead of boot
- save several seconds of reboot time
- capturing a dump after a kernel crash
- CONFIG_KEXEC (Processor Type and Features Kexec System Call) in the kernel configuration menu

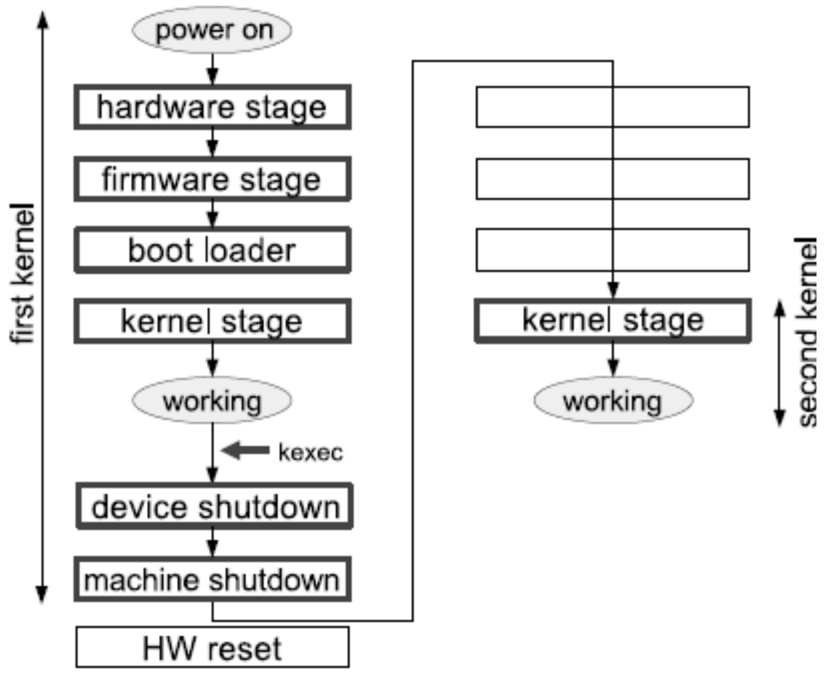
Kexec and Kdump

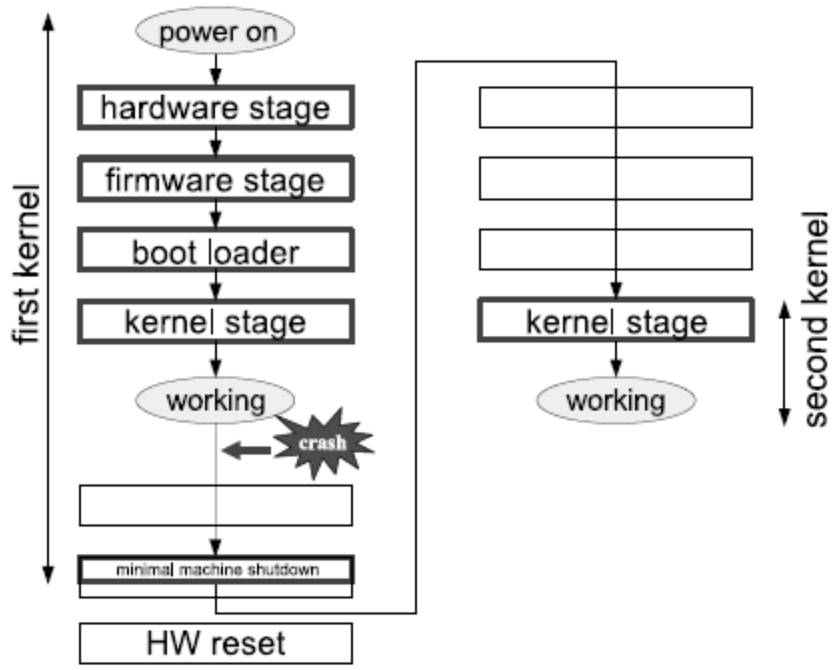
- Normal Linux boot



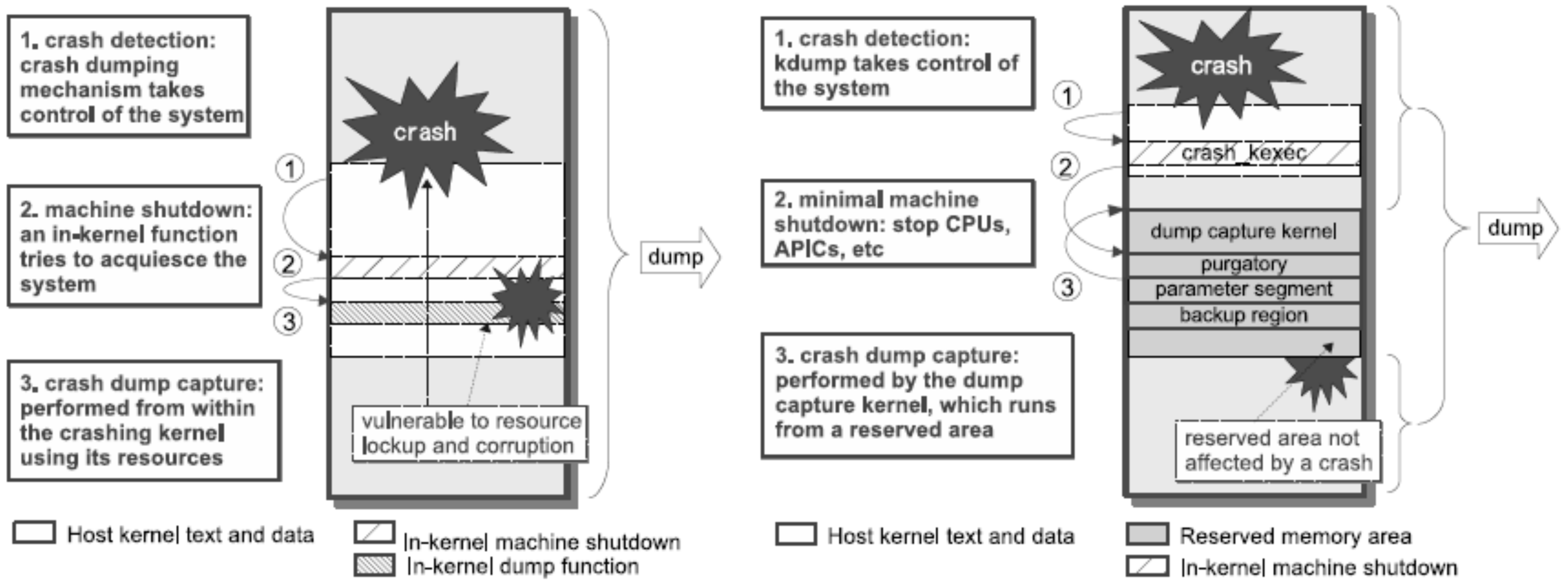
Kexec and Kdump

- Kexec boot　　　　　　kdump boot





Kdump


1. crash detection: crash dumping mechanism takes control of the system
2. machine shutdown: an in-kernel function tries to acquiesce the system
3. crash dump capture: performed from within the crashing kernel using its resources
1
2
3
crash
vulnerable to resource lockup and corruption
dump
Host kernel text and data
In-kernel machine shutdown
In-kernel dump function
1. crash detection: kdump takes control of the system
2. minimal machine shutdown: stop CPUs, APICs, etc
3. crash dump capture: performed by the dump capture kernel, which runs from a reserved area
1
2
3
crash
crash_kexec
dump capture kernel
purgatory
parameter segment
backup region
reserved area not affected by a crash
dump
Host kernel text and data
Reserved memory area
In-kernel machine shutdown

(a) In-kernel crash dumping

(b) Kdump-based crash dumping

Kernel hacking options

- Some kernel hacking options are architecture-dependent
- CONFIG_PRINTK_TIME: show Timing information on printks
- CONFIG_DEBUG_SLAB: debug slab memory allocations
- CONFIG_DEBUG_SPINLOCK: finds lock-related problems
- CONFIG_MAGIC_SYSRQ: Magic SysRq key
- CONFIG_DETECT_SOFTLOCKUP: detect tight loops in kernel code that last for more than 10 seconds

Kernel hacking options

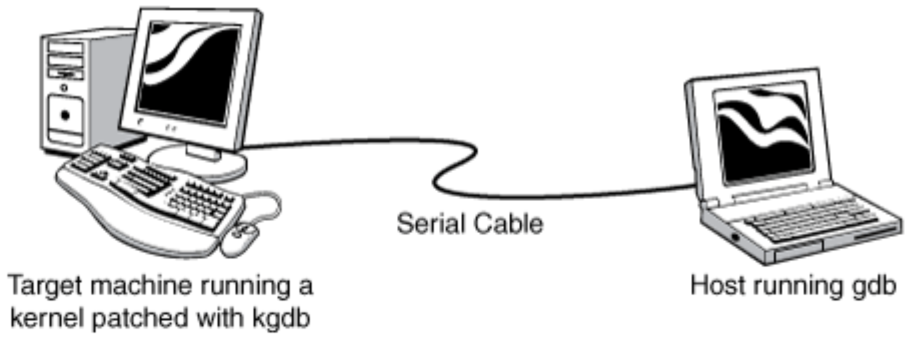
- CONFIG_DEBUG_SLAB/CONFIG_DEBUG_HIMEM/ CONFIG_DEBUG_PAGE_ALLOC : help debug memory management problems
- CONFIG_DEBUG_STACKOVERFLOW: warnings if the available stack space falls below a threshold
- CONFIG_DEBUG_STACK_USAGE): adds stack space instrumentation to the magic Sysrq key output
- CONFIG_DEBUG_BUGVERBOSE: verbose BUG() reporting
- CONFIG_KALLSYMS: debug an "oops" message

gdb

- Compile kernel with –g flag
- gdb vmlinux /proc/kcore
- Cannot modify the kernel data
- Cannot single-step
- Cannot set breakpoint

kgdb

- http://kgdb.linsyssoft.com/
- Remote debug
- Kernel patched + gdb (over serial line)
- Full gdb functions



kdb

- http://oss.sgi.com/projects/kdb/
- built-in kernel debugger (not remote debugger)
- kernel patch
- support variable modification, breakpoints, and single-stepping, …

kdb vs kgdb

	KDB	*KGDB*
Debugger environment	It is a debugger that needs to be built inside the kernel. All it requires is a console using which commands can be entered and output displayed on the console.	It requires a development machine to run the debugger as a normal process that communicates with the target using the GDB protocol over a serial cable. Recent versions of KGDB support the Ethernet interface.
Kernel support/ patches required	KDB requires two patches: a common kernel patch that implements the architecture-independent functionality and an architecture-dependent patch.	KGDB makes use of a single patch that has three components: ■ GDB stub that implements the GDB protocol on the target side, ■ Changes to the serial (or Ethernet) driver for sending and receiving the messages between the target and the development machine, ■ The changes to the exception handlers for giving control to the debugger when an exception happens.
Support for source-level debugging	No support for source-level debugging	Support for source-level debugging provided the kernel is compiled with the `-g` flag on the development machine and the kernel source tree is available. On the development machine where the debugger application runs, `-g` option tells gcc to generate debugging information while compiling, which in conjunction with source files provides source-level debugging.
Debugging features offered	The most commonly used debugging features of KDB are: ■ displaying and modifying memory and registers ■ applying breakpoints ■ stack backtrace Along with the user-applied breakpoints, KDB is invoked when the kernel hits an irrecoverable error condition such as panic or OOPS. The user can use the output of KDB to diagnose the problem.	Supports GDB execution control commands, stack trace, and KGDB-specific watchpoints among a host of other features such as thread analysis.
Kernel module debugging	KDB provides support for kernel module debugging.	Debugging modules using KGDB is tricky because the module is loaded on the target machine and the debugger (GDB) runs on a different machine; so the KGDB debugger needs to be informed of the module load address. KGDB 1.9 is accompanied by a special GDB that can automatically detect module loading and unloading. For KGDB versions equal to or less than 1.8, the developer has to make use of an explicit GDB command `add-symbol-file` to load the module object into GDB's memory along with the module load address.
Web sites for download	http://oss.sgi.com/projects/kdb/	http://kgdb.linsyssoft.com/

Some tricks

- Debug process-related kernel code
  - Use UID
- Rate kernel print

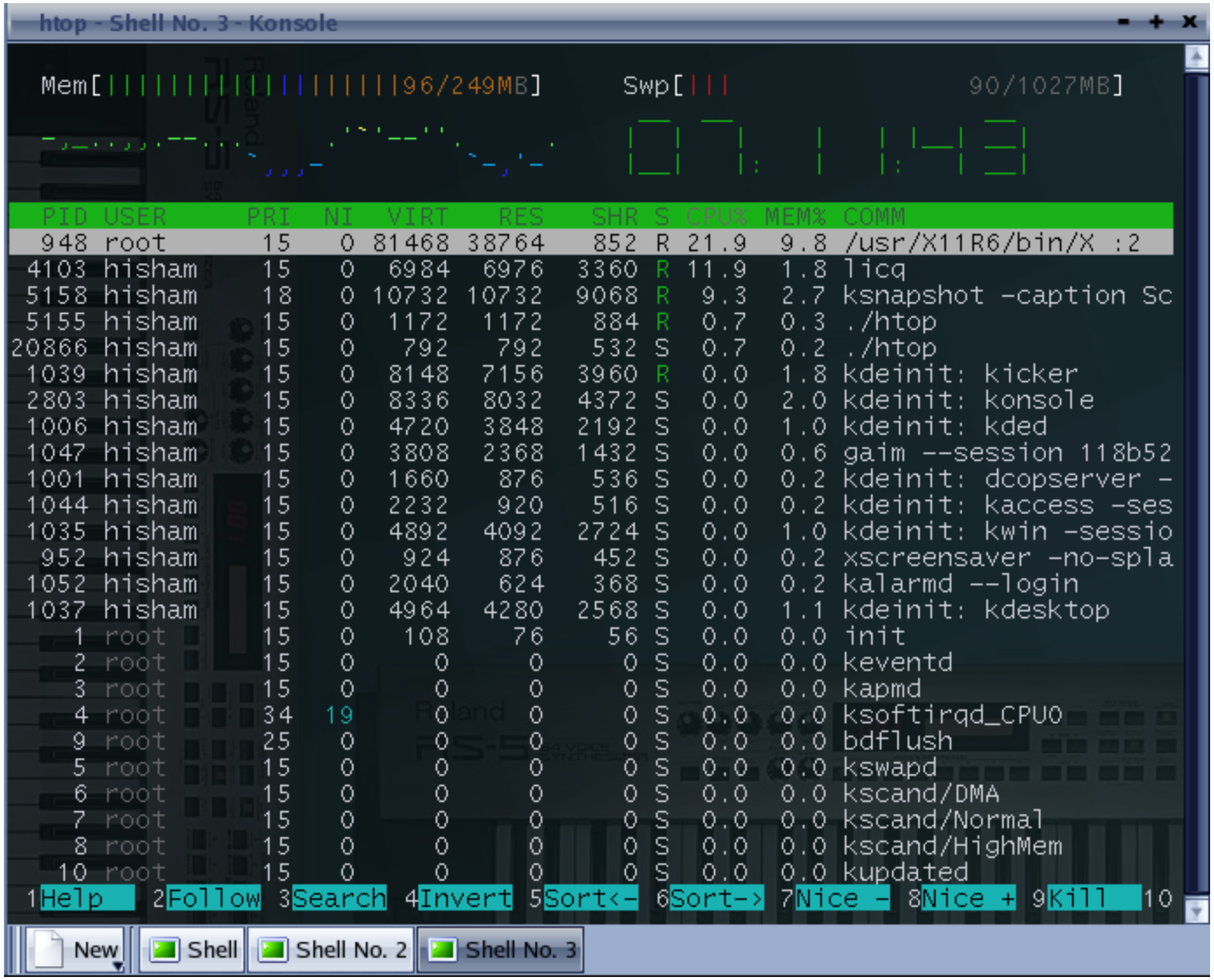
```
static unsigned long prev_jiffy = jiffies;    /* rate limiting */

if (time_after(jiffies, prev_jiffy + 2*HZ)) {
    prev_jiffy = jiffies;
    printk(KERN_ERR "blah blah blah\n");
}
```

```
static unsigned long limit = 0;

if (limit < 5) {
    limit++;
    printk(KERN_ERR "blah blah blah\n");
}
```

Linux Profile

- Linux-built-in vs. 3rd party package
- Instrument vs. non-instrument
- Trace-based vs. Counting-based vs. Sampling based
- Kernel profiling vs. AP profiling

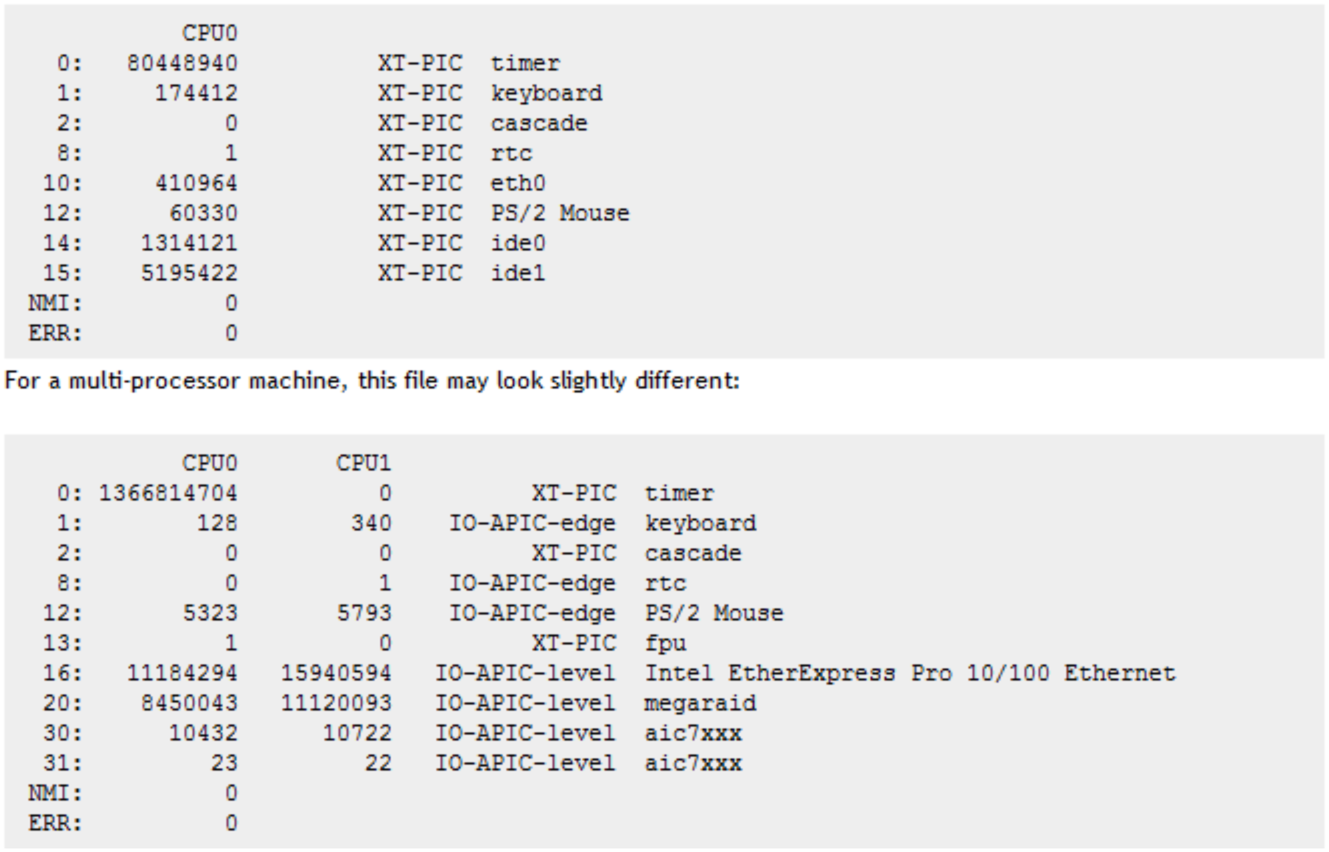
System Load Monitoring



```
Mem[||||||||||||||||||||96/249MB]     Swp[|||                    90/1027MB]

  PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM% COMM
  948 root       15   0 81468 38764   852 R 21.9  9.8 /usr/X11R6/bin/X :2
 4103 hisham     15   0  6984  6976  3360 R 11.9  1.8 licq
 5158 hisham     18   0 10732 10732  9068 R  9.3  2.7 ksnapshot -caption Sc
 5155 hisham     15   0  1172  1172   884 R  0.7  0.3 ./htop
20866 hisham     15   0   792   792   532 S  0.7  0.2 ./htop
 1039 hisham     15   0  8148  7156  3960 R  0.0  1.8 kdeinit: kicker
 2803 hisham     15   0  8336  8032  4372 S  0.0  2.0 kdeinit: konsole
 1006 hisham     15   0  4720  3848  2192 S  0.0  1.0 kdeinit: kded
 1047 hisham     15   0  3808  2368  1432 S  0.0  0.6 gaim --session 118b52
 1001 hisham     15   0  1660   876   536 S  0.0  0.2 kdeinit: dcopserver -
 1044 hisham     15   0  2232   920   516 S  0.0  0.2 kdeinit: kaccess -ses
 1035 hisham     15   0  4892  4092  2724 S  0.0  1.0 kdeinit: kwin -sessio
  952 hisham     15   0   924   876   452 S  0.0  0.2 xscreensaver -no-spla
 1052 hisham     15   0  2040   624   368 S  0.0  0.2 kalarmd --login
 1037 hisham     15   0  4964  4280  2568 S  0.0  1.1 kdeinit: kdesktop
    1 root       15   0   108    76    56 S  0.0  0.0 init
    2 root       15   0     0     0     0 S  0.0  0.0 keventd
    3 root       15   0     0     0     0 S  0.0  0.0 kapmd
    4 root       34  19     0     0     0 S  0.0  0.0 ksoftirqd_CPU0
    9 root       25   0     0     0     0 S  0.0  0.0 bdflush
    5 root       15   0     0     0     0 S  0.0  0.0 kswapd
    6 root       15   0     0     0     0 S  0.0  0.0 kscand/DMA
    7 root       15   0     0     0     0 S  0.0  0.0 kscand/Normal
    8 root       15   0     0     0     0 S  0.0  0.0 kscand/HighMem
   10 root       15   0     0     0     0 S  0.0  0.0 kupdated
 1Help  2Follow 3Search 4Invert 5Sort<- 6Sort-> 7Nice - 8Nice + 9Kill  10
```

System Load Monitoring

- /proc/interrupts

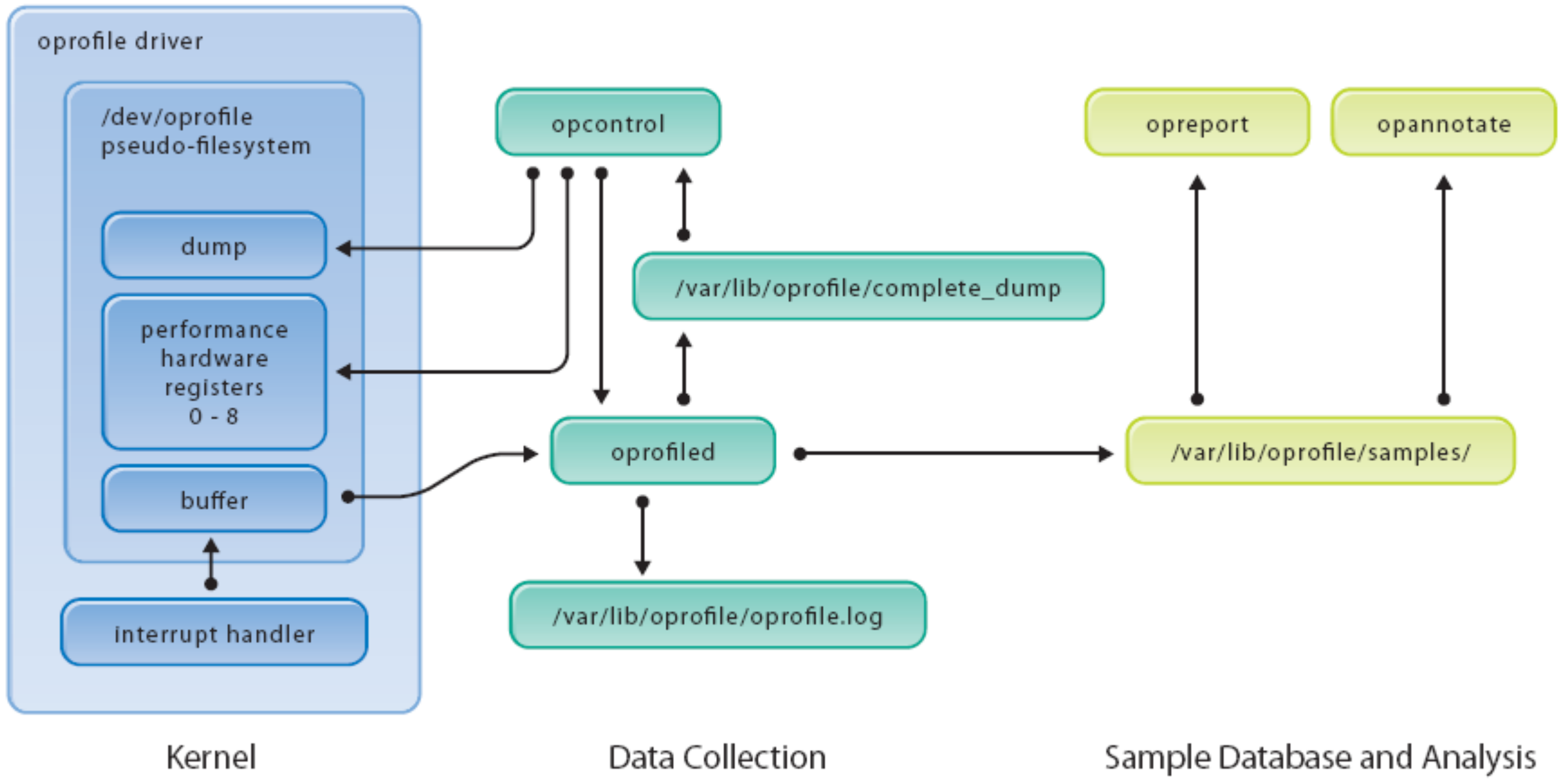


```
           CPU0
  0:   80448940          XT-PIC  timer
  1:     174412          XT-PIC  keyboard
  2:          0          XT-PIC  cascade
  8:          1          XT-PIC  rtc
 10:     410964          XT-PIC  eth0
 12:      60330          XT-PIC  PS/2 Mouse
 14:    1314121          XT-PIC  ide0
 15:    5195422          XT-PIC  ide1
NMI:          0
ERR:          0
```

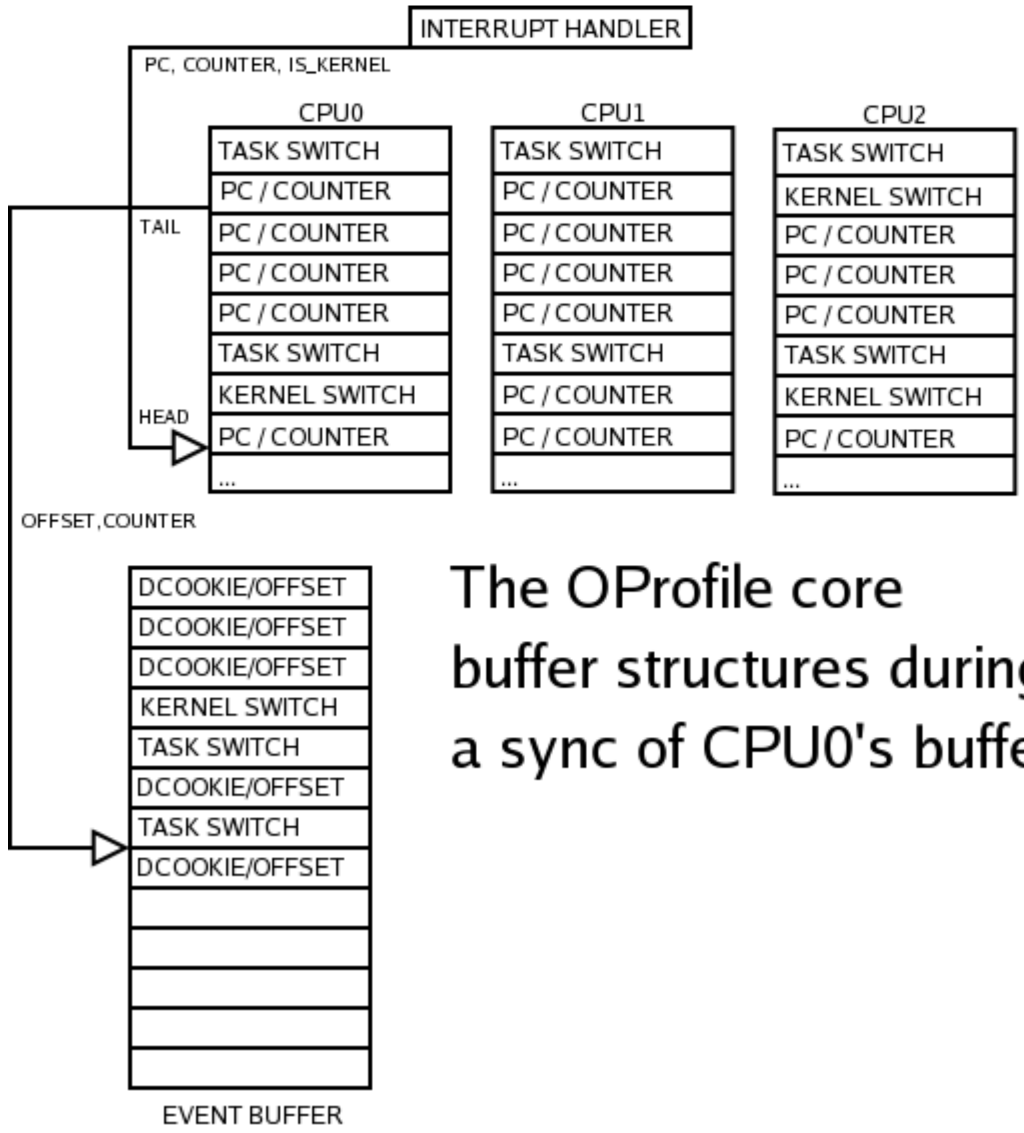
For a multi-processor machine, this file may look slightly different:

```
           CPU0       CPU1
  0: 1366814704          0          XT-PIC  timer
  1:        128        340    IO-APIC-edge  keyboard
  2:          0          0          XT-PIC  cascade
  8:          0          1    IO-APIC-edge  rtc
 12:       5323       5793    IO-APIC-edge  PS/2 Mouse
 13:          1          0          XT-PIC  fpu
 16:   11184294   15940594   IO-APIC-level  Intel EtherExpress Pro 10/100 Ethernet
 20:    8450043   11120093   IO-APIC-level  megaraid
 30:      10432      10722   IO-APIC-level  aic7xxx
 31:         23         22   IO-APIC-level  aic7xxx
NMI:          0
ERR:          0
```

OProfile Architecture

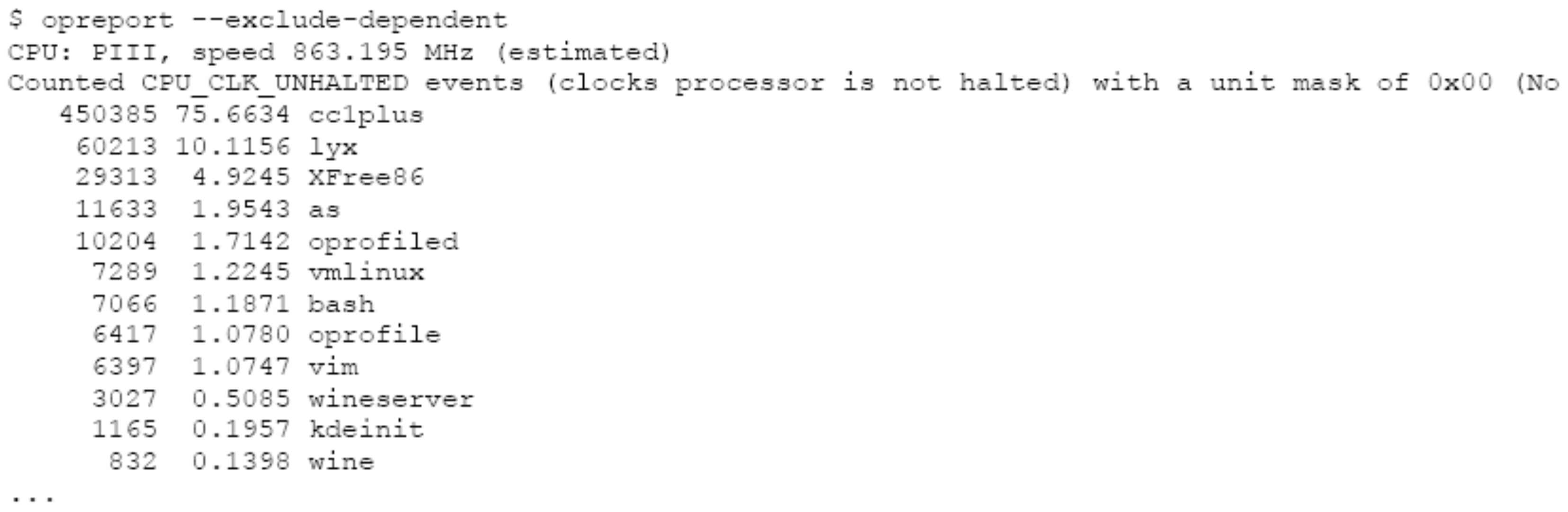

oprofile driver
/dev/oprofile pseudo-filesystem
dump
performance hardware registers 0 - 8
buffer
interrupt handler
opcontrol
/var/lib/oprofile/complete_dump
oprofiled
/var/lib/oprofile/oprofile.log
opreport
opannotate
/var/lib/oprofile/samples/
Kernel
Data Collection
Sample Database and Analysis

Oprofile Buffer



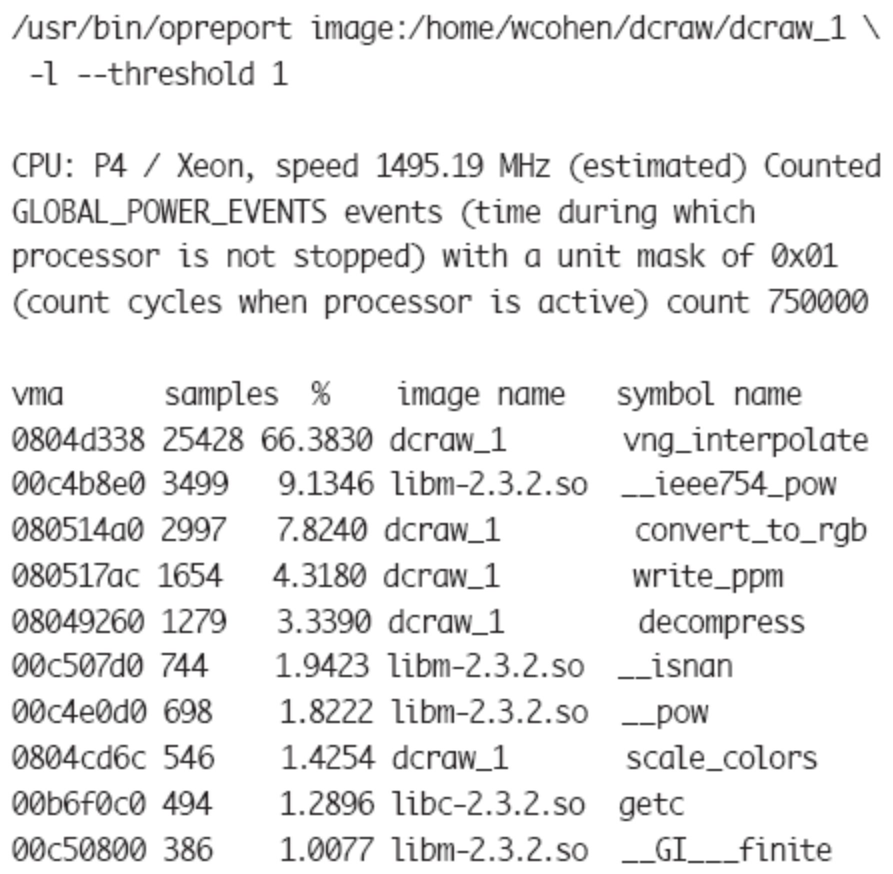
The OProfile core buffer structures during a sync of CPU0's buffer

System-wide binary image summary



```
$ opreport --exclude-dependent
CPU: PIII, speed 863.195 MHz (estimated)
Counted CPU_CLK_UNHALTED events (clocks processor is not halted) with a unit mask of 0x00 (No
   450385 75.6634 cc1plus
    60213 10.1156 lyx
    29313  4.9245 XFree86
    11633  1.9543 as
    10204  1.7142 oprofiled
     7289  1.2245 vmlinux
     7066  1.1871 bash
     6417  1.0780 oprofile
     6397  1.0747 vim
     3027  0.5085 wineserver
     1165  0.1957 kdeinit
      832  0.1398 wine
...
```

Function breakdown



```
/usr/bin/opreport image:/home/wcohen/dcraw/dcraw_1 \
 -l --threshold 1

CPU: P4 / Xeon, speed 1495.19 MHz (estimated) Counted
GLOBAL_POWER_EVENTS events (time during which
processor is not stopped) with a unit mask of 0x01
(count cycles when processor is active) count 750000

vma       samples  %     image name    symbol name
0804d338 25428 66.3830 dcraw_1         vng_interpolate
00c4b8e0 3499    9.1346 libm-2.3.2.so  __ieee754_pow
080514a0 2997    7.8240 dcraw_1         convert_to_rgb
080517ac 1654    4.3180 dcraw_1         write_ppm
08049260 1279    3.3390 dcraw_1         decompress
00c507d0 744     1.9423 libm-2.3.2.so  __isnan
00c4e0d0 698     1.8222 libm-2.3.2.so  __pow
0804cd6c 546     1.4254 dcraw_1        scale_colors
00b6f0c0 494     1.2896 libc-2.3.2.so  getc
00c50800 386     1.0077 libm-2.3.2.so  __GI___finite
```

Listing 2. Per function breakdown of samples for initial program

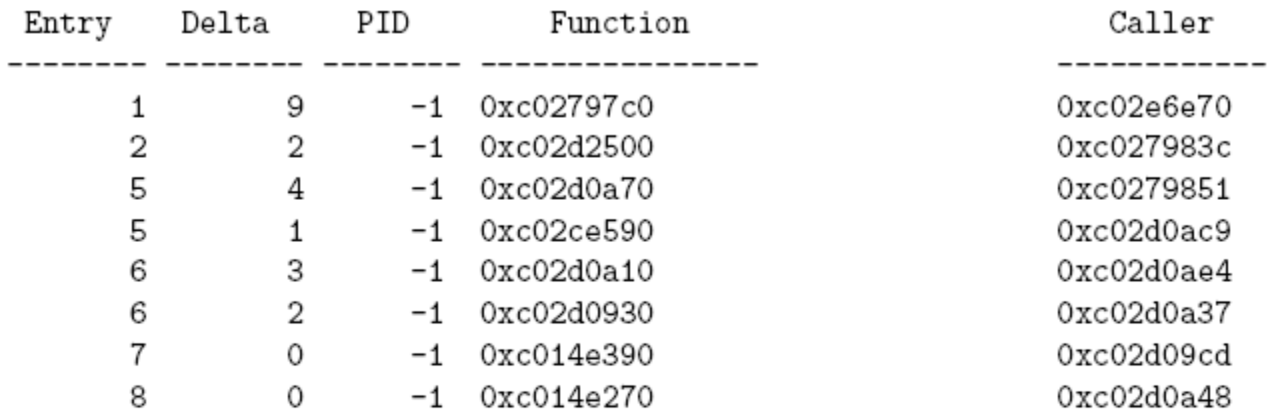
KFT

- Be careful of interpreting results
  - Duration
    - Do not subtract interrupts and thread switching
  - Delta
    - The problem may be caused by child functions
- Good for debugging straight-line code
  - No block or lock by mutex/semaphores

How to implement it

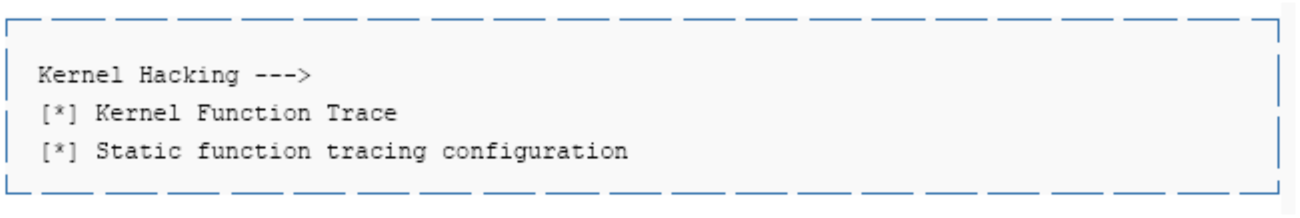
```
static inline void
__noinstrument do_func_entry(struct kft_run* run, void *this_fn,
        void *call_site)
{
        * check for log full condition
        * acquire lock on trace log so that multiple CPUs are serialized
        * allocate space for the new entry
        * unlock the log
}


static inline void
__noinstrument do_func_exit(struct kft_run* run, void *this_fn,
        void *call_site)
{
        collect the pid
        get the lock on trace log
        find matching entry in log - searching backwards from current log end
        when we find the entry point calculate the runtime (delta)
        check if it fits the filter criteria (if not don't log it)
        record CPU and PID
        and unlock the log file
}
```

How to implement it



```
 Entry    Delta     PID       Function                                 Caller
-------- -------- -------- ----------------                         ------------
       1        9       -1  0xc02797c0                               0xc02e6e70
       2        2       -1  0xc02d2500                               0xc027983c
       5        4       -1  0xc02d0a70                               0xc0279851
       5        1       -1  0xc02ce590                               0xc02d0ac9
       6        3       -1  0xc02d0a10                               0xc02d0ae4
       6        2       -1  0xc02d0930                               0xc02d0a37
       7        0       -1  0xc014e390                               0xc02d09cd
       8        0       -1  0xc014e270                               0xc02d0a48
```

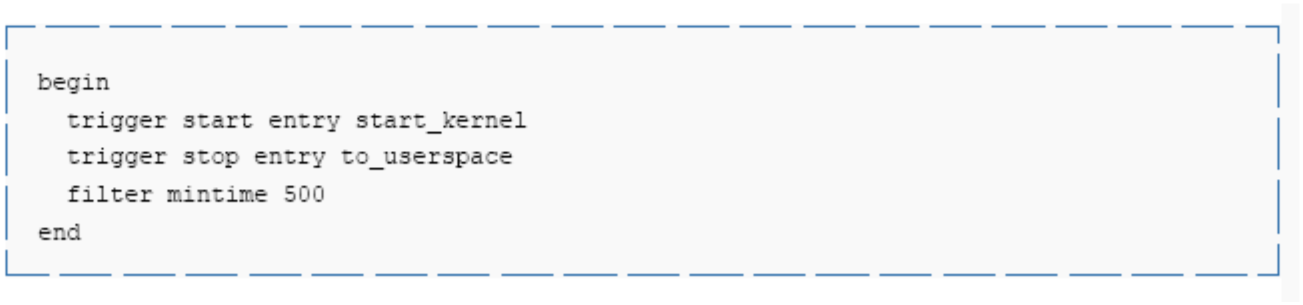
How to install it



```
Kernel Hacking --->
[*] Kernel Function Trace
[*] Static function tracing configuration
```

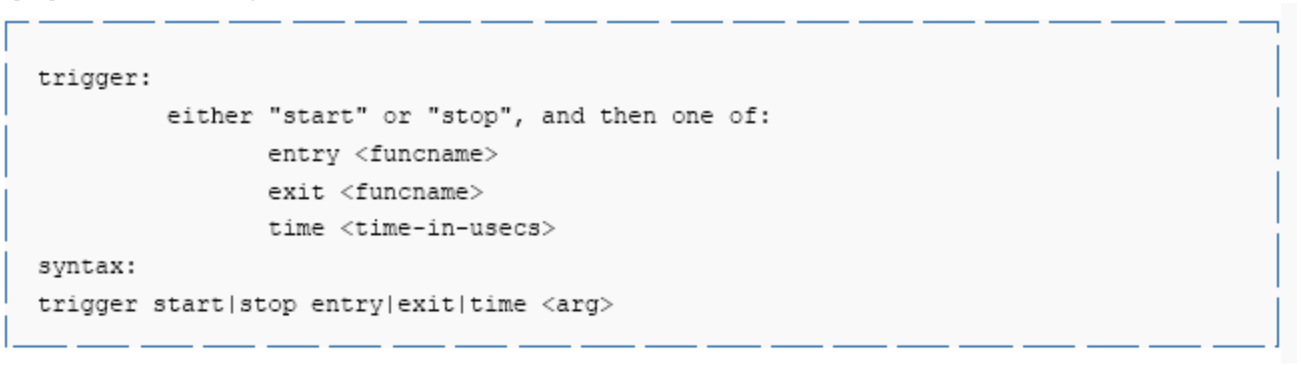
- Static: kernel/kftstatic.conf
- Dynamic: cat /trace.config > /proc/kft
- Save system.map

How to configure the trace run



```
begin
  trigger start entry start_kernel
  trigger stop entry to_userspace
  filter mintime 500
end
```

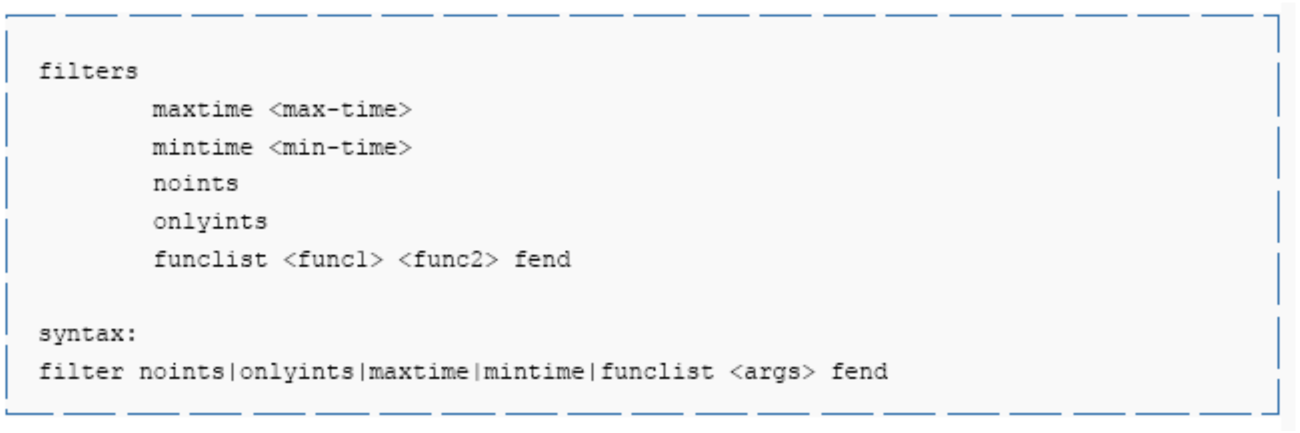
- Triggers
  - trigger start entry start_kernel
  - trigger stop exit do_fork
  - trigger start time 10000000
  - trigger stop time 5000



```
trigger:
         either "start" or "stop", and then one of:
                entry <funcname>
                exit <funcname>
                time <time-in-usecs>
syntax:
trigger start|stop entry|exit|time <arg>
```

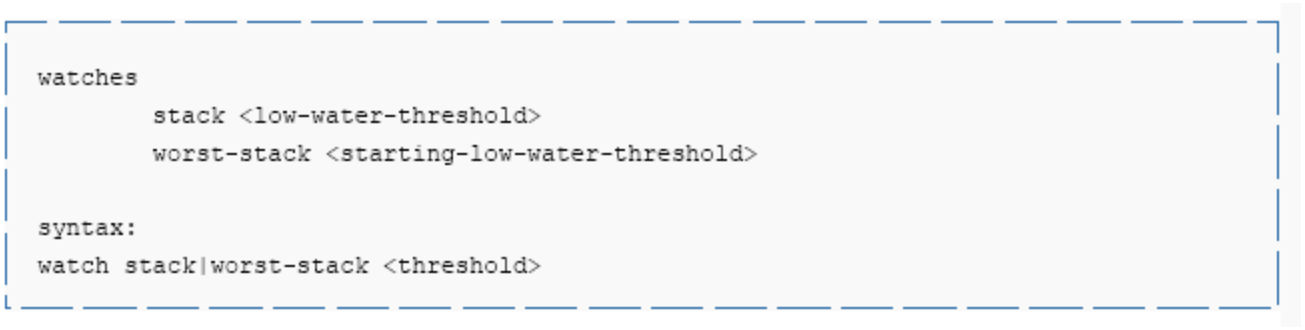
Filters

- Filters
  - filter mintime 100
  - filter maxtime 5000000
  - filter noints
  - filter onlyints
  - filter funclist do_fork sys_read fend



```
filters
        maxtime <max-time>
        mintime <min-time>
        noints
        onlyints
        funclist <func1> <func2> fend

syntax:
filter noints|onlyints|maxtime|mintime|funclist <args> fend
```

Watch



```
watches
        stack <low-water-threshold>
        worst-stack <starting-low-water-threshold>

syntax:
watch stack|worst-stack <threshold>
```

How to configure the trace run

- Static kernel/kftstatic.conf
- Dynamic
  - Edit trace.config
  - Sym2addr trace.config system.map > trace.config2
  - Cat trace.config2 > /proc/kft

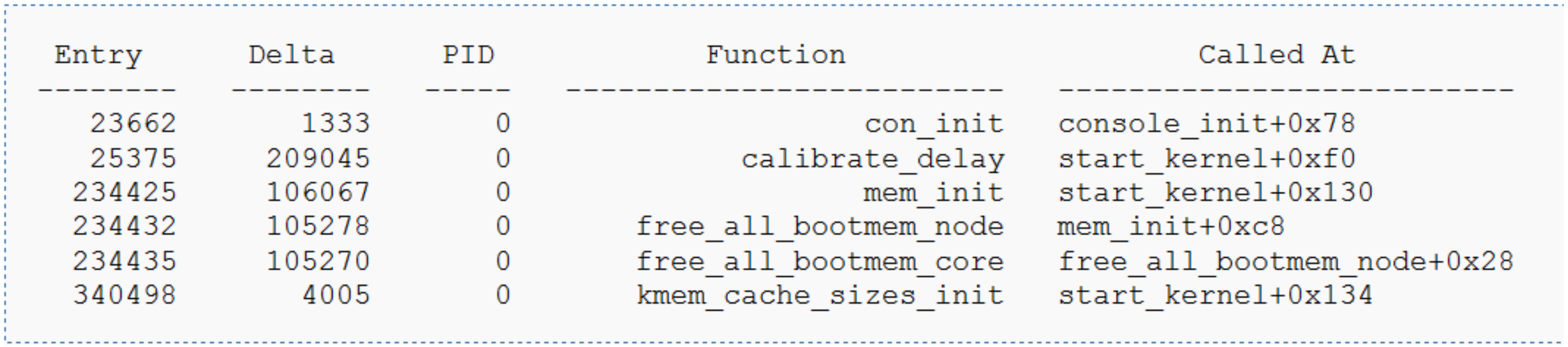
```
new
begin
  trigger start entry 0xc001d804
  trigger stop time 5000000
  filter mintime 500
  filter maxtime 0
  filter noints
end
```

```
new
begin
  trigger start entry do_fork
  trigger stop exit do_fork
  filter mintime 10
  filter maxtime 400
  filter noints
  logentries 500
end
```

```
new
begin
  trigger start time 5000000
  trigger stop time 5000
  filter onlyints
end
```

How to read and process the trace results

- Cat /proc/kft_data > kft.log
- addr2sym kft.log –m system.map > kft.lst



```
  Entry      Delta     PID            Function                   Called At
 --------  --------  -----  -------------------------  -------------------------
    23662      1333      0                   con_init  console_init+0x78
    25375    209045      0            calibrate_delay  start_kernel+0xf0
   234425    106067      0                   mem_init  start_kernel+0x130
   234432    105278      0      free_all_bootmem_node  mem_init+0xc8
   234435    105270      0      free_all_bootmem_core  free_all_bootmem_node+0x28
   340498      4005      0      kmem_cache_sizes_init  start_kernel+0x134
```

Post processing by kd

- [show all functions sorted by time]
- $ ./kd kftsample.lst | less

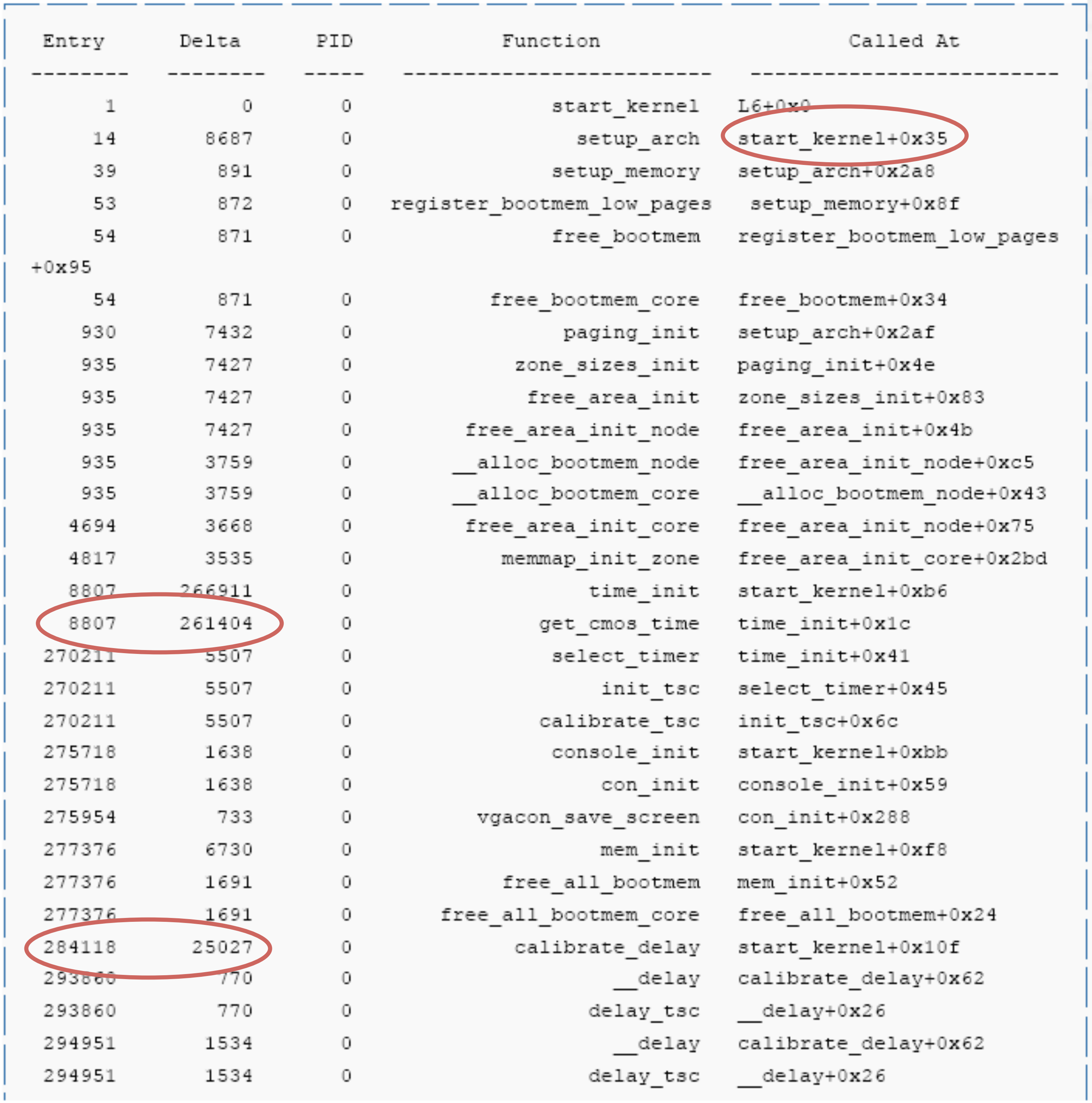
- [show only 10 top time-consuming functions]
- $ ./kd -n 10 kftsample.lst

- [show only functions lasting longer than 100 milliseconds]
- $ ./kd -t 100000 kftsample.lst

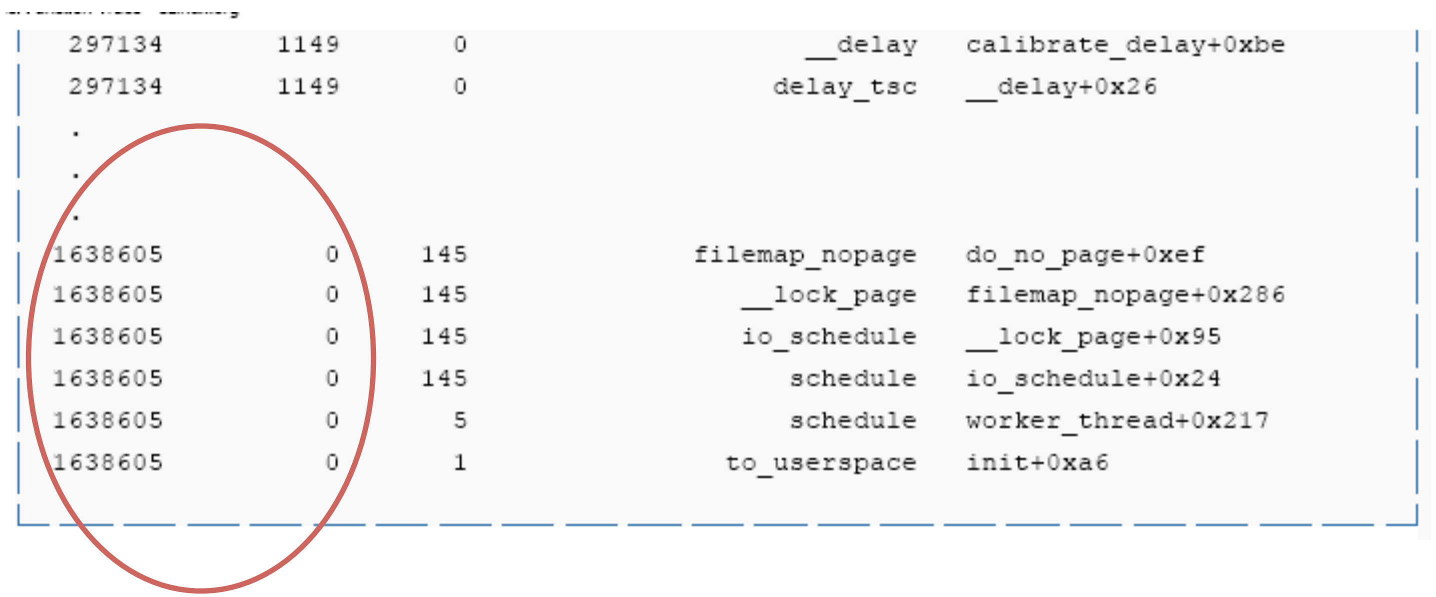
- [show each function's most time-consuming child, and the number of times it was called.]
- $ ./kd -f Fcatlmn kftsample.lst

- [show call traces]
- $ ./kd -c kftsample.lst

- [show call traces with timing data, and functions interlaced]
- $ ./kd -c -l -i kftsample.lst



```
  Entry      Delta     PID           Function                    Called At
 --------   --------   -----   -------------------------   -------------------------
        1          0      0                 start_kernel   L6+0x0
       14       8687      0                   setup_arch   start_kernel+0x35
       39        891      0                 setup_memory   setup_arch+0x2a8
       53        872      0    register_bootmem_low_pages   setup_memory+0x8f
       54        871      0                 free_bootmem   register_bootmem_low_pages
 +0x95
       54        871      0            free_bootmem_core   free_bootmem+0x34
      930       7432      0                  paging_init   setup_arch+0x2af
      935       7427      0              zone_sizes_init   paging_init+0x4e
      935       7427      0               free_area_init   zone_sizes_init+0x83
      935       7427      0          free_area_init_node   free_area_init+0x4b
      935       3759      0          __alloc_bootmem_node   free_area_init_node+0xc5
      935       3759      0          __alloc_bootmem_core   __alloc_bootmem_node+0x43
     4694       3668      0          free_area_init_core   free_area_init_node+0x75
     4817       3535      0             memmap_init_zone   free_area_init_core+0x2bd
     8807     266911      0                    time_init   start_kernel+0xb6
     8807     261404      0                get_cmos_time   time_init+0x1c
   270211       5507      0                 select_timer   time_init+0x41
   270211       5507      0                     init_tsc   select_timer+0x45
   270211       5507      0                calibrate_tsc   init_tsc+0x6c
   275718       1638      0                 console_init   start_kernel+0xbb
   275718       1638      0                     con_init   console_init+0x59
   275954        733      0           vgacon_save_screen   con_init+0x288
   277376       6730      0                     mem_init   start_kernel+0xf8
   277376       1691      0              free_all_bootmem   mem_init+0x52
   277376       1691      0         free_all_bootmem_core   free_all_bootmem+0x24
   284118      25027      0              calibrate_delay   start_kernel+0x10f
   293860        770      0                      __delay   calibrate_delay+0x62
   293860        770      0                    delay_tsc   __delay+0x26
   294951       1534      0                      __delay   calibrate_delay+0x62
   294951       1534      0                    delay_tsc   __delay+0x26
```



```
  297134        1149        0            __delay   calibrate_delay+0xbe
  297134        1149        0          delay_tsc   __delay+0x26
  .
  .
  .
 1638605           0      145     filemap_nopage   do_no_page+0xef
 1638605           0      145        __lock_page   filemap_nopage+0x286
 1638605           0      145        io_schedule   __lock_page+0x95
 1638605           0      145           schedule   io_schedule+0x24
 1638605           0        5           schedule   worker_thread+0x217
 1638605           0        1       to_userspace   init+0xa6
```

```
$ ~/work/kft/kft/kd -n 30 kftboot-9.lst
Function                   Count Time     Average  Local
-------------------------- ----- -------- -------- --------
schedule                     192  5173790    26946  5173790
do_basic_setup                 1  1159270  1159270       14
do_initcalls                   1  1159256  1159256      627
__delay                      156   619322     3970        0
delay_tsc                    156   619322     3970   619322
__const_udelay               146   608427     4167        0
probe_hwif                     8   553972    69246      126
do_probe                      31   553025    17839       68
ide_delay_50ms               103   552588     5364        0
isapnp_init                    1   383138   383138       18
isapnp_isolate                 1   383120   383120   311629
ide_init                       1   339778   339778       22
probe_for_hwifs                1   339756   339756      103
ide_scan_pcibus                1   339653   339653       13
init_setup_piix                2   339640   169820        0
ide_scan_pcidev                2   339640   169820        0
piix_init_one                  2   339640   169820        0
ide_setup_pci_device           2   339640   169820      242
probe_hwif_init                4   339398    84849       40
```

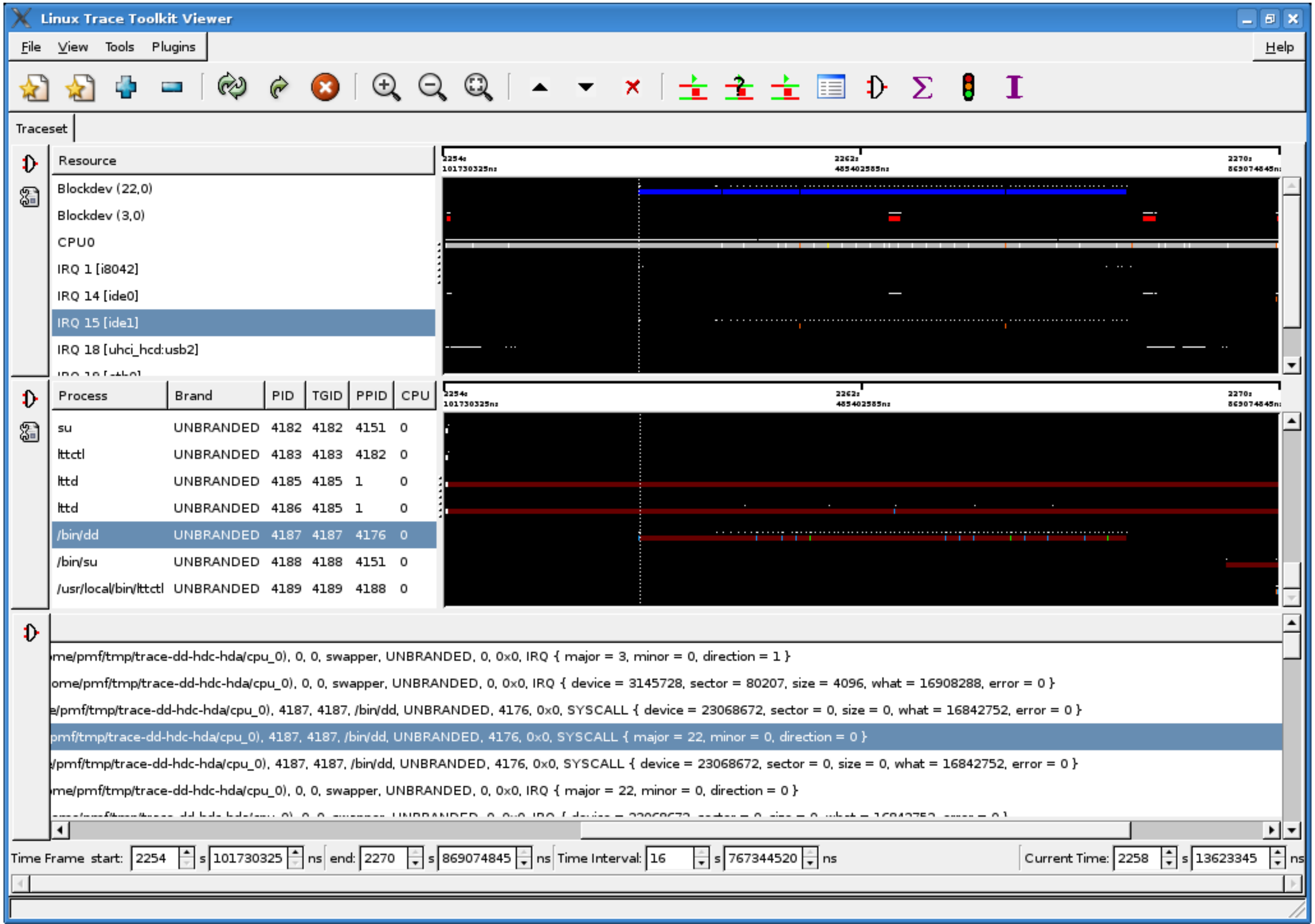
```
     Entry       Duration    Local        Pid    Trace
       4        20428         209        33 do_fork
       7            6           6        33 |   alloc_pidmap
      18         2643          84        33 |   copy_process
      21          114          19        33 |   |   dup_task_struct
      24            8           6        33 |   |   |   prepare_to_copy
      27            2           2        33 |   |   |   |   sub_preempt_count
      35           22           9        33 |   |   |   kmem_cache_alloc
      38            2           2        33 |   |   |   |   __might_sleep
      43           11           9        33 |   |   |   |   cache_alloc_refill
      49            2           2        33 |   |   |   |   |   sub_preempt_count
      60           65           6        33 |   |   |   __get_free_pages
      63           59          14        33 |   |   |   |   __alloc_pages
      65            3           3        33 |   |   |   |   |   __might_sleep
      71            3           3        33 |   |   |   |   |   zone_watermark_ok
      77           37          17        33 |   |   |   |   |   buffered_rmqueue
      80            4           4        33 |   |   |   |   |   |   __rmqueue
      86            3           3        33 |   |   |   |   |   |   sub_preempt_count
      92            3           3        33 |   |   |   |   |   |   bad_range
      98            2           2        33 |   |   |   |   |   |   __mod_page_state
     103            8           5        33 |   |   |   |   |   |   prep_new_page
     106            3           3        33 |   |   |   |   |   |   |   set_page_refs
     117            2           2        33 |   |   |   |   |   zone_statistics
     141           25           4        33 |   |   do_posix_clock_monotonic_gettime
     143           21           6        33 |   |   |   do_posix_clock_monotonic_get
     146           15           6        33 |   |   |   |
do_posix_clock_monotonic_gettime_parts
     149            9           6        33 |   |   |   |   |   getnstimeofday
     152            3           3        33 |   |   |   |   |   |   do_gettimeofday
     169            3           3        33 |   |   copy_semundo
     174           41          17        33 |   |   copy_files
```

Gprof

- cc -g -c myprog.c utils.c -pg
- cc -o myprog myprog.o utils.o –pg

```
Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls  ms/call  ms/call  name
 33.34      0.02     0.02     7208     0.00     0.00  open
 16.67      0.03     0.01      244     0.04     0.12  offtime
 16.67      0.04     0.01        8     1.25     1.25  memccpy
 16.67      0.05     0.01        7     1.43     1.43  write
 16.67      0.06     0.01                             mcount
  0.00      0.06     0.00      236     0.00     0.00  tzset
  0.00      0.06     0.00      192     0.00     0.00  tolower
  0.00      0.06     0.00       47     0.00     0.00  strlen
  0.00      0.06     0.00       45     0.00     0.00  strchr
  0.00      0.06     0.00        1     0.00    50.00  main
  0.00      0.06     0.00        1     0.00     0.00  memcpy
  0.00      0.06     0.00        1     0.00    10.11  print
  0.00      0.06     0.00        1     0.00     0.00  profil
  0.00      0.06     0.00        1     0.00    50.00  report
```

Linux Trace Toolkit



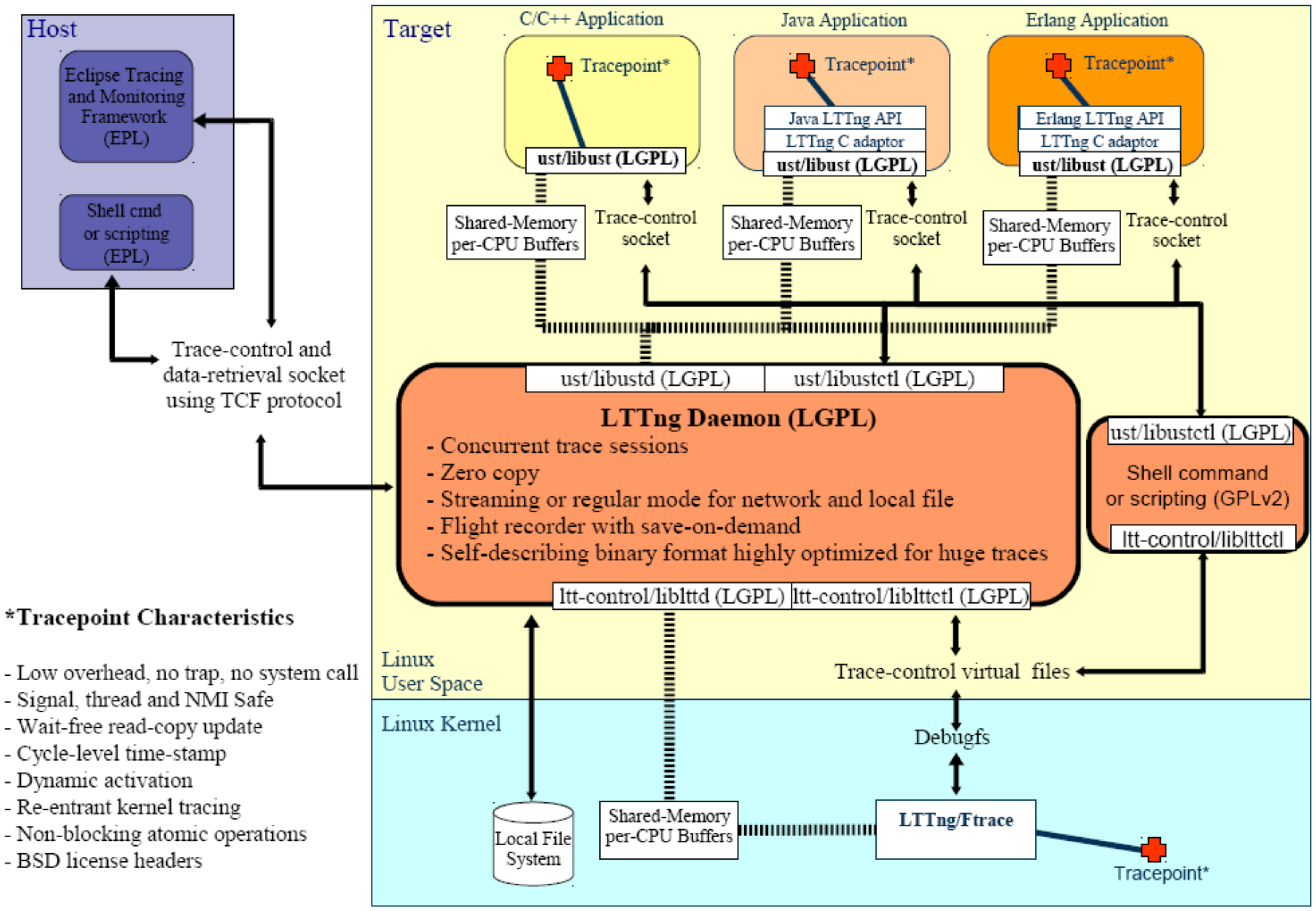
Resource
Blockdev (22,0)
Blockdev (3,0)
CPU0
IRQ 1 [i8042]
IRQ 14 [ide0]
IRQ 15 [ide1]
IRQ 18 [uhci_hcd:usb2]

Process	Brand	PID	TGID	PPID	CPU
su	UNBRANDED	4182	4182	4151	0
lttctl	UNBRANDED	4183	4183	4182	0
lttd	UNBRANDED	4185	4185	1	0
lttd	UNBRANDED	4186	4185	1	0
/bin/dd	UNBRANDED	4187	4187	4176	0
/bin/su	UNBRANDED	4188	4188	4151	0
/usr/local/bin/lttctl	UNBRANDED	4189	4189	4188	0

```
me/pmf/tmp/trace-dd-hdc-hda/cpu_0), 0, 0, swapper, UNBRANDED, 0, 0x0, IRQ { major = 3, minor = 0, direction = 1 }
ome/pmf/tmp/trace-dd-hdc-hda/cpu_0), 0, 0, swapper, UNBRANDED, 0, 0x0, IRQ { device = 3145728, sector = 80207, size = 4096, what = 16908288, error = 0 }
e/pmf/tmp/trace-dd-hdc-hda/cpu_0), 4187, 4187, /bin/dd, UNBRANDED, 4176, 0x0, SYSCALL { device = 23068672, sector = 0, size = 0, what = 16842752, error = 0 }
pmf/tmp/trace-dd-hdc-hda/cpu_0), 4187, 4187, /bin/dd, UNBRANDED, 4176, 0x0, SYSCALL { major = 22, minor = 0, direction = 0 }
/pmf/tmp/trace-dd-hdc-hda/cpu_0), 4187, 4187, /bin/dd, UNBRANDED, 4176, 0x0, SYSCALL { device = 23068672, sector = 0, size = 0, what = 16842752, error = 0 }
me/pmf/tmp/trace-dd-hdc-hda/cpu_0), 0, 0, swapper, UNBRANDED, 0, 0x0, IRQ { major = 22, minor = 0, direction = 0 }
```

Time Frame start: 2254 s 101730325 ns end: 2270 s 869074845 ns Time Interval: 16 s 767344520 ns Current Time: 2258 s 13623345 ns

LTTng Low-Overhead Tracing Architecture


Host
Eclipse Tracing and Monitoring Framework (EPL)
Shell cmd or scripting (EPL)
Trace-control and data-retrieval socket using TCF protocol
Target
C/C++ Application
Java Application
Erlang Application
Tracepoint*
Java LTTng API
Erlang LTTng API
LTTng C adaptor
ust/libust (LGPL)
Shared-Memory per-CPU Buffers
Trace-control socket
ust/libustd (LGPL)
ust/libustctl (LGPL)
LTTng Daemon (LGPL)
- Concurrent trace sessions
- Zero copy
- Streaming or regular mode for network and local file
- Flight recorder with save-on-demand
- Self-describing binary format highly optimized for huge traces
ltt-control/liblttd (LGPL)
ltt-control/liblttctl (LGPL)
Shell command or scripting (GPLv2)
ltt-control/liblttctl
Linux User Space
Trace-control virtual files
Linux Kernel
Debugfs
Local File System
LTTng/Ftrace

***Tracepoint Characteristics**

- Low overhead, no trap, no system call
- Signal, thread and NMI Safe
- Wait-free read-copy update
- Cycle-level time-stamp
- Dynamic activation
- Re-entrant kernel tracing
- Non-blocking atomic operations
- BSD license headers

Other useful information

- Linux Test Project
  - http://ltp.sourceforge.net/
  - a suite consisting of around 3,000 tests designed to exercise different parts of the kernel
- User Mode Linux
  - http://user-mode-linux.sourceforge.net/
  - lets you debug the kernel without "oops"ing the machine

lockmeter

- Lockmeter is a tool for instrumenting the spin locks in a multiprocessor Linux kernel
- http://oss.sgi.com/projects/lockmeter/

```
System: Linux testlinux.austin.ibm.com 2.3.99-pre6 #147 SMP Tue Aug 22 15:18:05 CDT 2000 i686

All (4) CPUs

Start time: Fri Aug 25 16:09:05 2000
End   time: Fri Aug 25 16:11:26 2000
Delta Time: 141.33 sec.
Hash table slots in use:      356.

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
SPINLOCKS        HOLD                 WAIT
   UTIL   CON    MEAN( MAX ) MEAN ( MAX   ) TOTAL  NAME
. . .
  21.86% 25.12% 3.3us( 87us) 4.4us(311us) 9480279 runqueue_lock                          ←Note 1
   1.62% 25.14% 3.3us( 13us) 1.1us(136us)  696844  __wake_up+0x110
   0.00%  0.00% 0.2us(0.2us)   0us               1  __wake_up_sync+0xfc
   0.00% 23.60% 5.1us( 19us) 4.3us(112us)     322  process_timeout+0x1c
   0.00%  8.71% 0.5us(1.3us) 0.2us(7.1us)     551  release+0x28
   0.00% 61.11% 2.0us(5.1us) 1.2us(5.9us)      36  schedule_tail+0x48
  14.69% 19.63% 5.5us( 87us) 1.7us(311us) 3806754  schedule+0xd0
   1.71% 56.33% 2.1us( 12us)  14us(295us) 1150208  schedule+0x444
   0.51% 14.35% 4.3us( 28us) 0.5us( 51us)  165306  schedule+0x710
   0.68% 12.89% 0.8us(4.5us) 0.7us(178us) 1224206  send_sig_info+0x2a0
   0.00% 40.57% 4.8us( 10us) 0.9us( 11us)     801  setscheduler+0x68
   0.78% 46.40% 0.9us(6.1us)  15us(265us) 1210679  sys_sched_yield+0xc
   1.88%  5.56% 2.2us( 14us) 0.4us(164us) 1224571  wake_up_process+0x18

   0.95%  3.87% 0.6us( 17us) 0.1us(9.4us) 2380970 timerlist_lock                         ←Note 2
   0.13%  4.66% 0.4us(4.1us) 0.1us(9.4us)  405952   add_timer+0x14
   0.33%  4.50% 0.5us(4.3us) 0.1us(9.2us) 1004500   del_timer+0x14
   0.00%  0.18% 0.5us(1.5us) 0.0us(2.1us)     565   del_timer_sync+0x20
   0.31%  3.35% 0.6us(4.2us) 0.0us(8.2us)  777490   mod_timer+0x18
   0.03%  1.86% 3.0us( 17us) 0.0us(4.8us)   14134   timer_bh+0x12c
   0.16%  0.99% 1.2us(5.1us) 0.0us(6.3us)  178329   timer_bh+0x2b4
. . .
   3.47%  0.00% 7.0us(142us) 0.0us(7.1us) 702015 __wake_up+0x24                          ←Note 3
   0.22%  0.56% 0.4us( 36us) 0.0us(5.9us) 811287 dev_queue_xmit+0x30
   0.01%  0.00% 1.1us(5.9us)   0us         14558 do_IRQ+0x40
   0.01%  0.00% 4.7us( 31us)   0us          1521 do_brk+0x108
   0.00%  0.00% 0.2us(0.9us)   0us           429 do_exit+0x240
   0.00%  0.00% 0.2us(0.6us)   0us           338 do_fork+0x6fc
. . .
```

```
RWLOCK READERS HOLD    MAX   RDR BUSY PERIOD    WAIT
  UTIL   CON    MEAN  READERS MEAN    (   MAX   ) MEAN ( MAX )   TOTAL NAME
. . .
  52.91% 0.00% 105.8us   5   114.8us (8274.8us) 0.0us(3.3us) 1402747 tasklist_lock        ←Note 4
         0.00%                                    0us                28  count_active_tasks+0x10
         0.23%                                  0.0us(2.3us)        429  exit_notify+0x1c
         0.00%                                    0us                 5  exit_notify+0xb8
         0.00%                                    0us               576  get_pid_list+0x18
         0.00%                                  0.0us(3.0us) 1224079  kill_something_info+0xb8
         0.00%                                    0us             11002  proc_pid_lookup+0x4c
         0.00%                                    0us                 7  proc_root_lookup+0x30
         0.00%                                    0us            165306  schedule+0x6d0
         0.00%                                    0us                25  session_of_pgrp+0x14
         1.12%                                  0.0us(3.3us)        801  setscheduler+0x78
         0.00%                                    0us                18  sys_setpgid+0x38
         0.00%                                    0us                 1  sys_setsid+0x10
         0.00%                                    0us               461  sys_wait4+0x158
         0.00%                                    0us                 9  will_become_orphaned_
                                                                          pgrp+0x14
. . .
 0.33%   0.06%   0.5us   2     0.5us  (   6.5us) 0.0us(528us)  856780 xtime_lock
         0.06%                                  0.0us(528us)  856780  do_gettimeofday+0x10
. . .

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
RWLOCK WRITERS   HOLD        WAIT (ALL)    WAIT (WW)        SPIN SPIN
UTIL     CON  MEAN ( MAX ) MEAN ( MAX   ) MEAN (   MAX ) TOTAL ALL  WW  NAME

0.00% 10.53% 0.8us(2.7us) 9.8us(1515us) 0.8us( 1.7us)  1691 173   5 tasklist_lock
0.00% 10.68% 1.2us(2.6us) 7.3us(1515us) 0.8us( 1.7us)   833  84   5  do_fork+0x8a4        ←Note 5
0.00%  2.80% 0.2us(0.8us) 0.2us(  13us)   0us           429  12   0  exit_notify+0x284
0.00% 17.95% 0.7us(2.7us)  25us( 486us)   0us           429  77   0  release+0x78
. . .
0.12%  1.29% 6.2us(802us) 0.0us(7.7us)  0.9us( 4.3us) 28352 281  84 xtime_lock
0.03%  1.68% 2.8us(802us) 0.0us(7.7us)  1.0us( 3.1us) 14193 180  59  timer_bh+0x14
0.10%  0.89% 9.6us( 24us) 0.0us(6.4us)  0.8us( 4.3us) 14159 101  25  timer_interrupt+0x14
```

Memory leaking

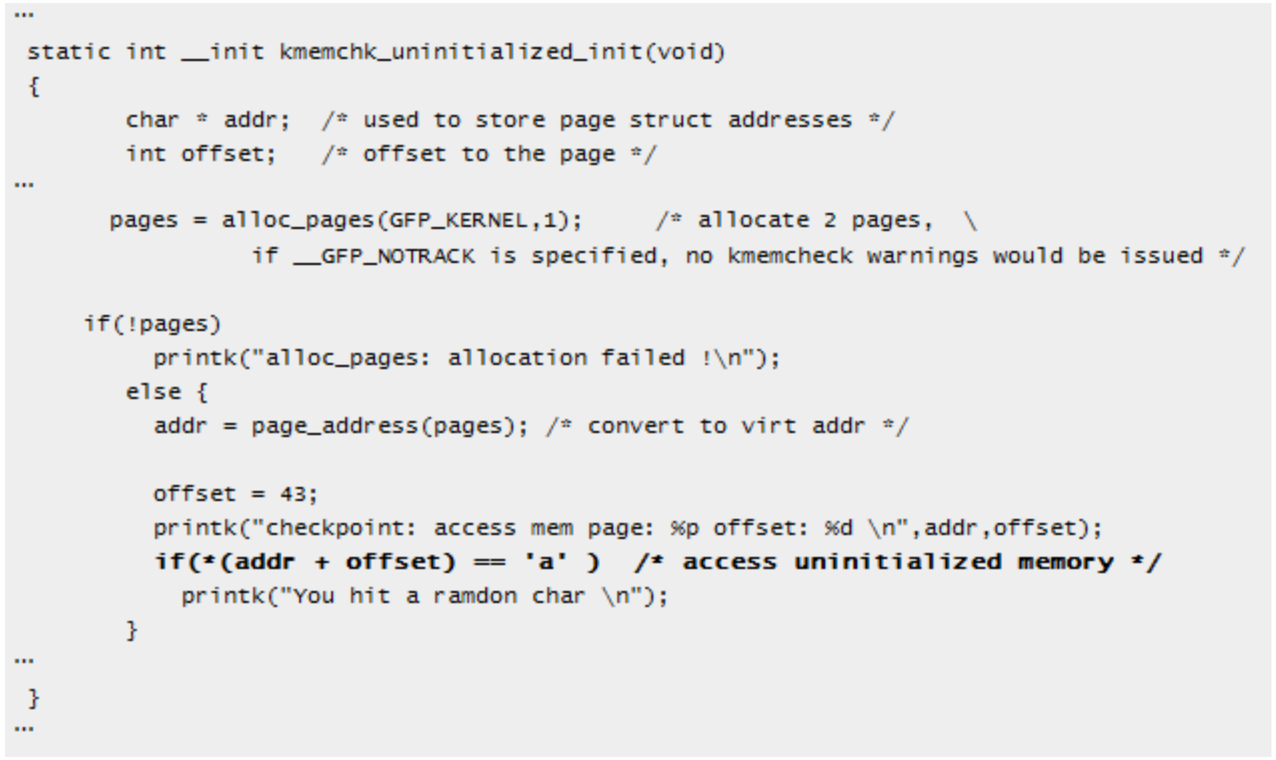
- User space: dmalloc, …

```
/* dmalloc_test.c */

#include <stdio.h>
#include <stdlib.h>

#ifdef USE_DMALLOC
#include <dmalloc.h>
#endif

int main()
{
  char *test[5];
  unsigned int i;

  for (i=0; i < 5; i++)
  {
    unsigned int size = rand()%1024;
    test[i] = (char *)malloc(size);
    printf ("Allocated memory of size %d\n",size);
  }
  for (i=0; i<2; i++)
    free(test[i*2]);
}
```

```
calling dmalloc malloc
Allocated memory of size 359
calling dmalloc malloc
Allocated memory of size 966
calling dmalloc malloc
Allocated memory of size 105
calling dmalloc malloc
Allocated memory of size 115
calling dmalloc malloc
Allocated memory of size 81
bash>cat dlog
1094293908: 8: Dmalloc version '5.3.0' from 'http://dmalloc.com/'
1094293908: 8: flags = 0x3, logfile 'dlog'
1094293908: 8: interval = 0, addr = 0, seen # = 0, limit = 0
1094293908: 8: starting time = 1094293908
1094293908: 8: process pid = 4709
1094293908: 8: Dumping Chunk Statistics:
1094293908: 8: basic-block 4096 bytes, alignment 8 bytes, heap
               grows up
1094293908: 8: heap address range: 0x80c3000 to 0x80ca000, 28672
               bytes
1094293908: 8:     user blocks: 3 blocks, 12217 bytes (42%)
1094293908: 8:    admin blocks: 4 blocks, 16384 bytes (57%)
1094293908: 8: external blocks: 0 blocks, 0 bytes (0%)
1094293908: 8:    total blocks: 7 blocks, 28672 bytes
1094293908: 8: heap checked 0
1094293908: 8: alloc calls: malloc 5, calloc 0, realloc 0, free 3
1094293908: 8: alloc calls: recalloc 0, memalign 0, valloc 0
1094293908: 8: alloc calls: new 0, delete 0
1094293908: 8:   current memory in use: 1081 bytes (2 pnts)
1094293908: 8:  total memory allocated: 1626 bytes (5 pnts)
1094293908: 8:  max in use at one time: 1626 bytes (5 pnts)
1094293908: 8: max alloced with 1 call: 966 bytes
1094293908: 8: max unused memory space: 294 bytes (15%)
1094293908: 8: top 10 allocations:
1094293908: 8:  total-size  count in-use-size  count  source
1094293908: 8:        1626      5        1081  2  ra=0x8048a46
1094293908: 8:        1626      5        1081  2  Total of 1
1094293908: 8: Dumping Not-Freed Pointers Changed Since Start:
1094293908: 8:  not freed: '0x80c6c00|s1' (966 bytes) from
                'ra=0x8048a46'
1094293908: 8:  not freed: '0x80c8f00|s1' (115 bytes) from
                'ra=0x8048a46'
1094293908: 8:  total-size  count  source
1094293908: 8:        1081      2  ra=0x8048a46
1094293908: 8:        1081      2  Total of 1
1094293908: 8: ending time = 1094293908, elapsed since start =
               0:00:00
```

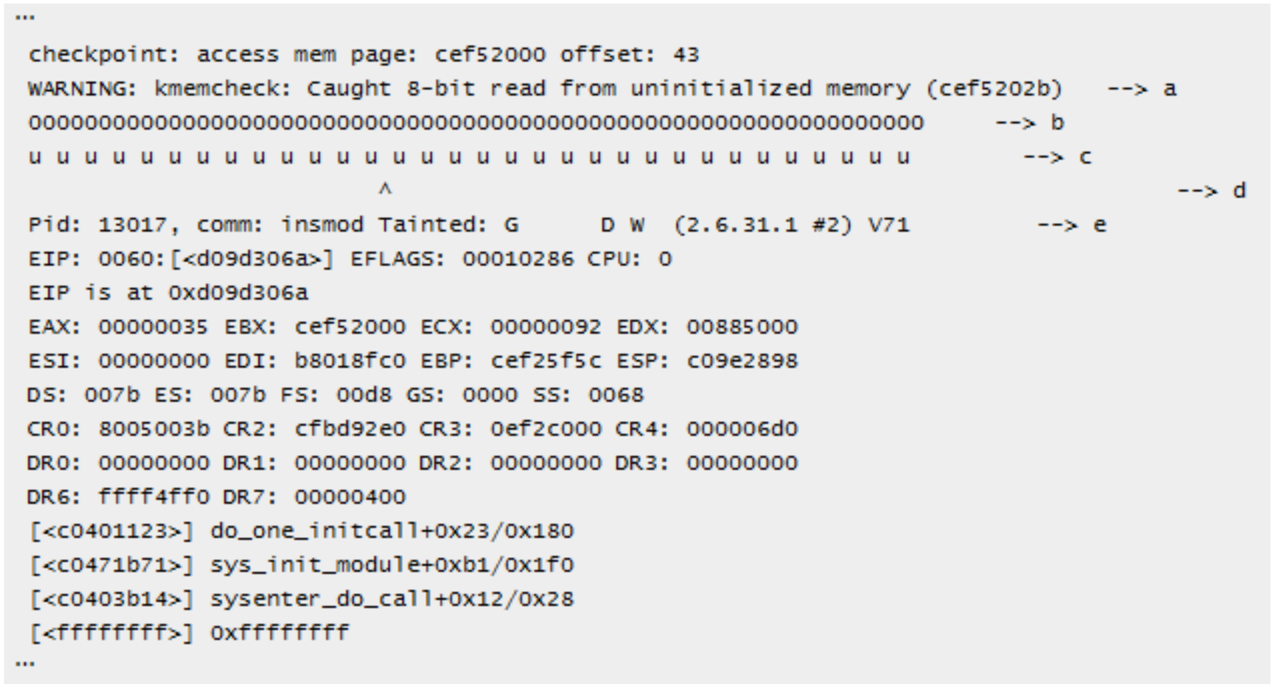
Memory leaking

- Kernel: kmemcheck



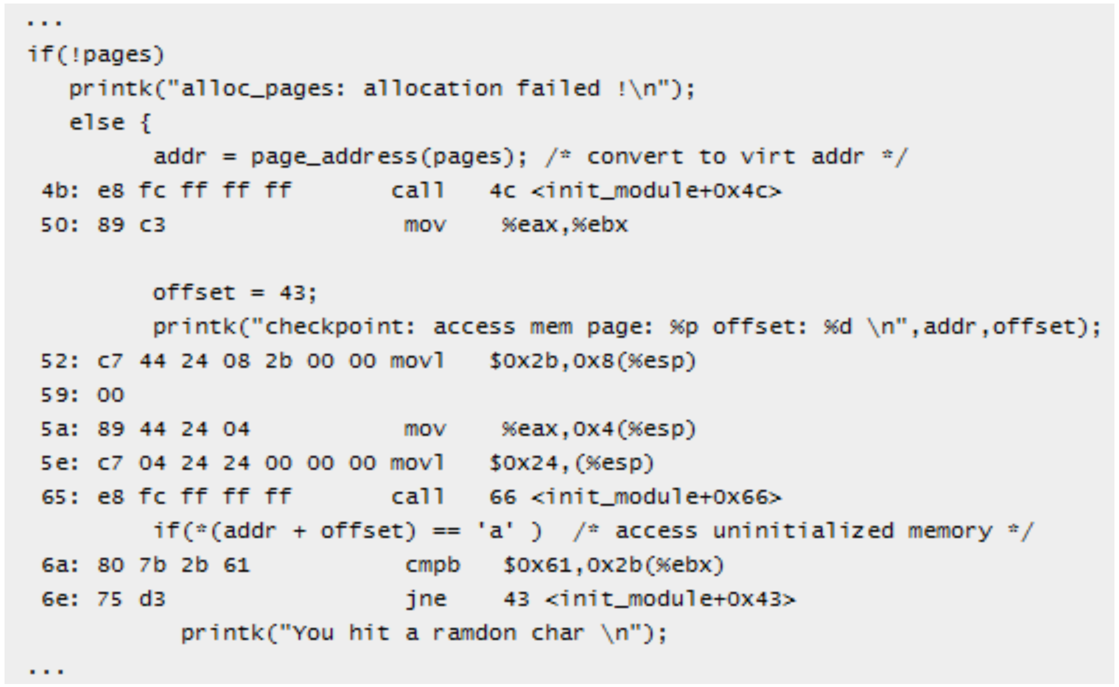
```
…
 static int __init kmemchk_uninitialized_init(void)
 {
        char * addr;  /* used to store page struct addresses */
        int offset;   /* offset to the page */
…
       pages = alloc_pages(GFP_KERNEL,1);     /* allocate 2 pages,  \
                 if __GFP_NOTRACK is specified, no kmemcheck warnings would be issued */

     if(!pages)
          printk("alloc_pages: allocation failed !\n");
        else {
          addr = page_address(pages); /* convert to virt addr */

          offset = 43;
          printk("checkpoint: access mem page: %p offset: %d \n",addr,offset);
          if(*(addr + offset) == 'a' )  /* access uninitialized memory */
            printk("You hit a ramdon char \n");
        }
…
 }
…
```

Kmemcheck WARNING



```
…
 checkpoint: access mem page: cef52000 offset: 43
 WARNING: kmemcheck: Caught 8-bit read from uninitialized memory (cef5202b)   --> a
 0000000000000000000000000000000000000000000000000000000000000000     --> b
 u u u u u u u u u u u u u u u u u u u u u u u u u u u u u u u u        --> c
                          ^                                                        --> d
 Pid: 13017, comm: insmod Tainted: G      D W  (2.6.31.1 #2) V71         --> e
 EIP: 0060:[<d09d306a>] EFLAGS: 00010286 CPU: 0
 EIP is at 0xd09d306a
 EAX: 00000035 EBX: cef52000 ECX: 00000092 EDX: 00885000
 ESI: 00000000 EDI: b8018fc0 EBP: cef25f5c ESP: c09e2898
 DS: 007b ES: 007b FS: 00d8 GS: 0000 SS: 0068
 CR0: 8005003b CR2: cfbd92e0 CR3: 0ef2c000 CR4: 000006d0
 DR0: 00000000 DR1: 00000000 DR2: 00000000 DR3: 00000000
 DR6: ffff4ff0 DR7: 00000400
 [<c0401123>] do_one_initcall+0x23/0x180
 [<c0471b71>] sys_init_module+0xb1/0x1f0
 [<c0403b14>] sysenter_do_call+0x12/0x28
 [<ffffffff>] 0xffffffff
…
```

objdump



```
...
if(!pages)
   printk("alloc_pages: allocation failed !\n");
   else {
         addr = page_address(pages); /* convert to virt addr */
 4b: e8 fc ff ff ff       call   4c <init_module+0x4c>
 50: 89 c3                 mov    %eax,%ebx

         offset = 43;
         printk("checkpoint: access mem page: %p offset: %d \n",addr,offset);
 52: c7 44 24 08 2b 00 00 movl   $0x2b,0x8(%esp)
 59: 00
 5a: 89 44 24 04           mov    %eax,0x4(%esp)
 5e: c7 04 24 24 00 00 00 movl   $0x24,(%esp)
 65: e8 fc ff ff ff       call   66 <init_module+0x66>
         if(*(addr + offset) == 'a' )  /* access uninitialized memory */
 6a: 80 7b 2b 61           cmpb   $0x61,0x2b(%ebx)
 6e: 75 d3                 jne    43 <init_module+0x43>
           printk("You hit a ramdon char \n");
...
```

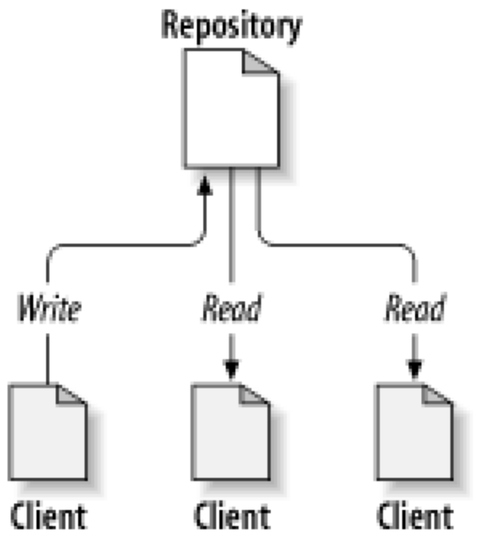
Test equipment

- JTAG/ICE debugger
- Logical analyzer
- NIC
  - Sniffer
- USB
  - Protocol Analyzer
- …

How to maintain kernel codes

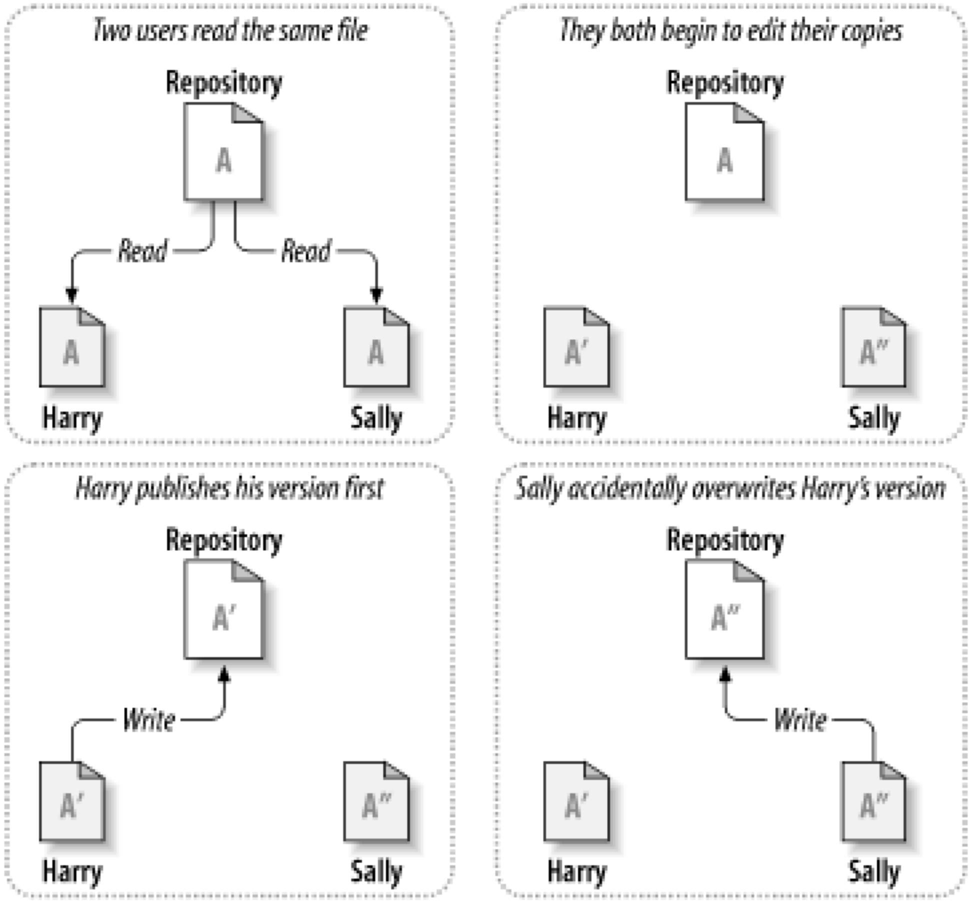
- Why version control?
  - Single developer
  - Multiple developers

How to maintain kernel codes



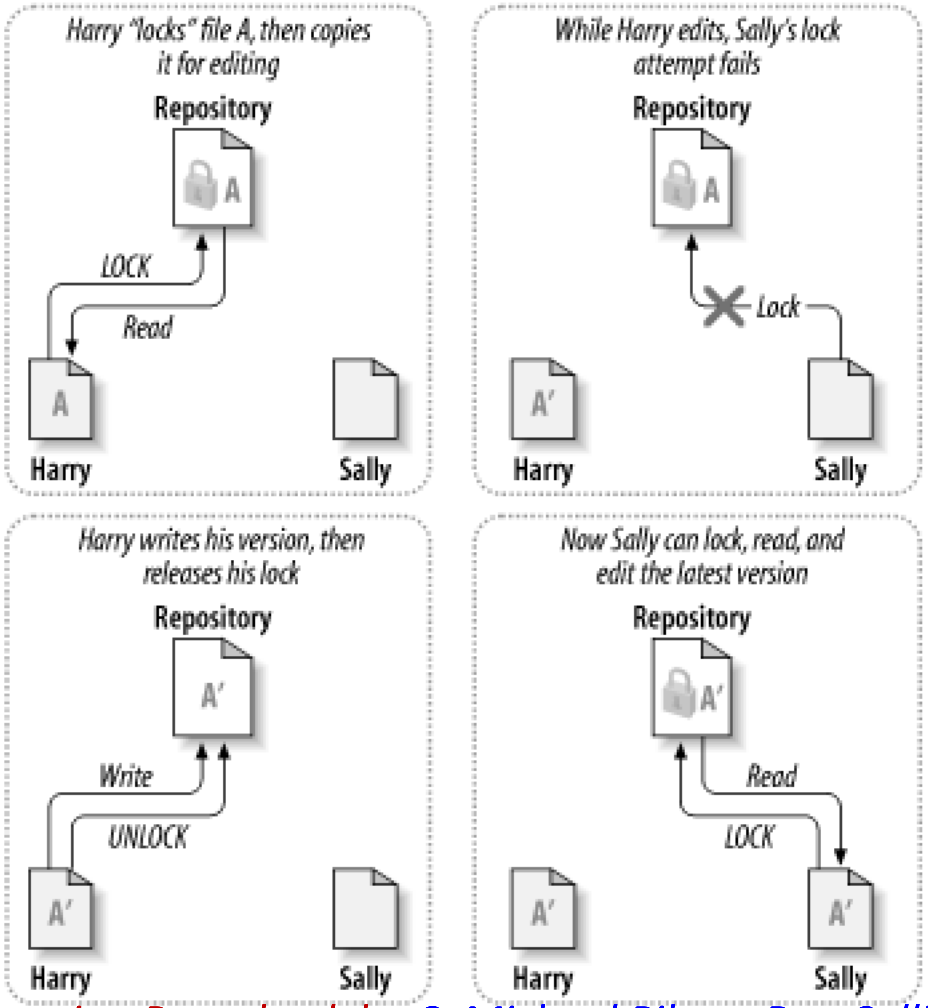
Version Control with Subversion Paperback by C. Michael Pilato, Ben Collins-Sussman , Brian W. Fitzpatrick

lock-modify-unlock problem


Two users read the same file
Repository
A
Read
Read
A
A
Harry
Sally
They both begin to edit their copies
Repository
A
A′
A″
Harry
Sally
Harry publishes his version first
Repository
A′
Write
A′
A″
Harry
Sally
Sally accidentally overwrites Harry's version
Repository
A″
Write
A′
A″
Harry
Sally

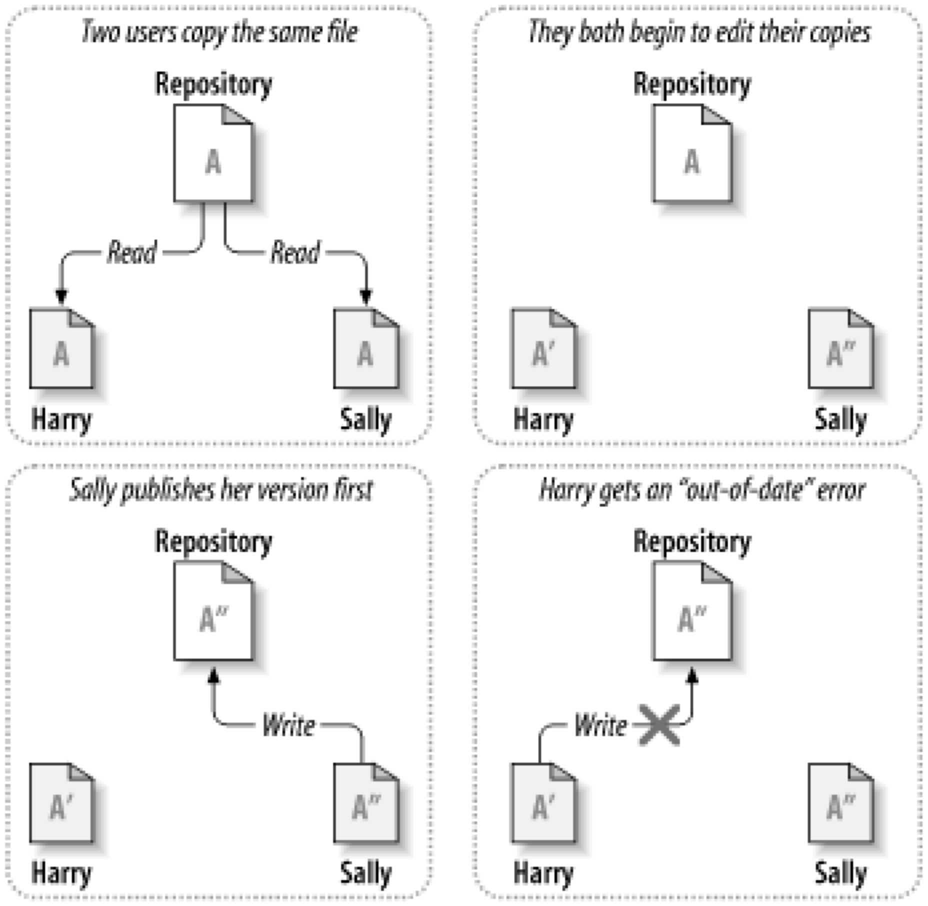
Version Control with Subversion Paperback by [C. Michael Pilato], [Ben Collins-Sussman], [Brian W. Fitzpatrick]

lock-modify-unlock solution


Harry "locks" file A, then copies it for editing
Repository
A
LOCK
Read
A
Harry
Sally
While Harry edits, Sally's lock attempt fails
Repository
A
Lock
A′
Harry
Sally
Harry writes his version, then releases his lock
Repository
A′
Write
UNLOCK
A′
Harry
Sally
Now Sally can lock, read, and edit the latest version
Repository
A′
Read
LOCK
A′
Harry
A′
Sally

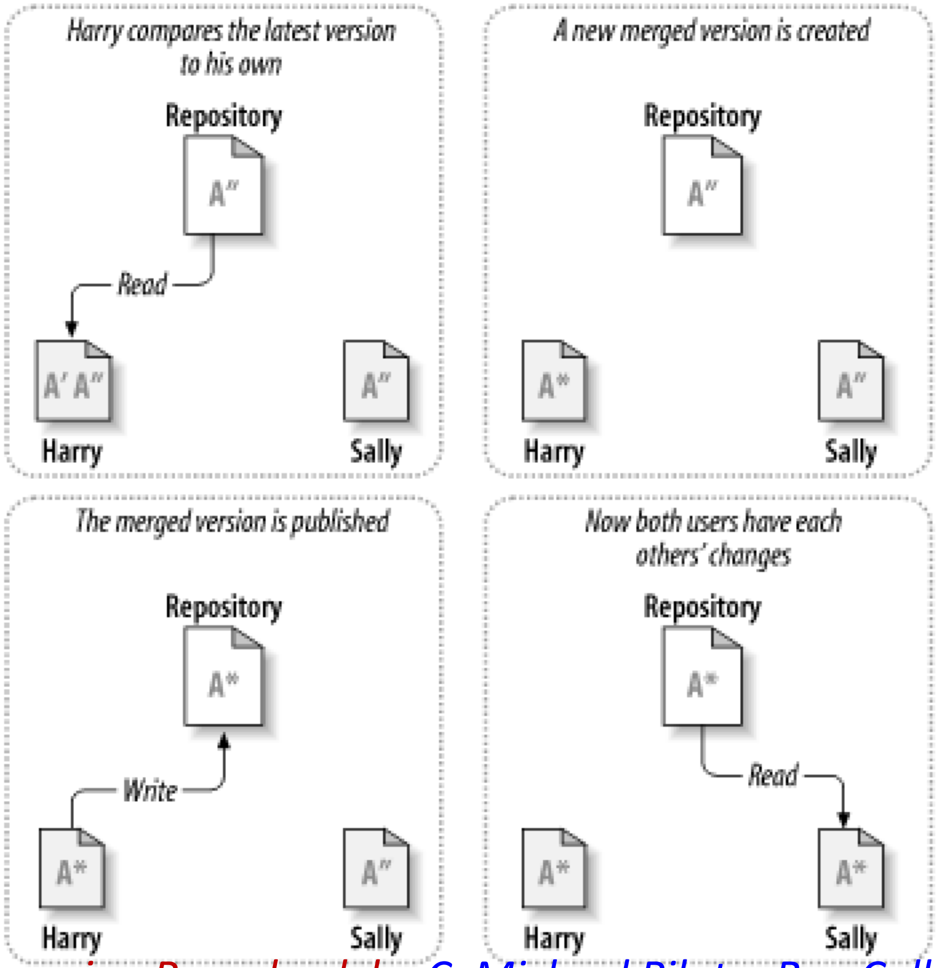
Version Control with Subversion Paperback by [C. Michael Pilato](), [Ben Collins-Sussman](), [Brian W. Fitzpatrick]()

copy-modify-merge problem


Two users copy the same file
Repository
A
Read
Read
A
Harry
A
Sally
They both begin to edit their copies
Repository
A
A′
Harry
A″
Sally
Sally publishes her version first
Repository
A″
Write
A′
Harry
A″
Sally
Harry gets an "out-of-date" error
Repository
A″
Write
A′
Harry
A″
Sally

Version Control with Subversion Paperback by C. Michael Pilato, Ben Collins-Sussman , Brian W. Fitzpatrick

copy-modify-merge solution



Harry compares the latest version to his own
Repository
A″
Read
A′ A″
Harry
A″
Sally
A new merged version is created
Repository
A″
A*
Harry
A″
Sally
The merged version is published
Repository
A*
Write
A*
Harry
A″
Sally
Now both users have each others' changes
Repository
A*
Read
A*
Harry
A*
Sally

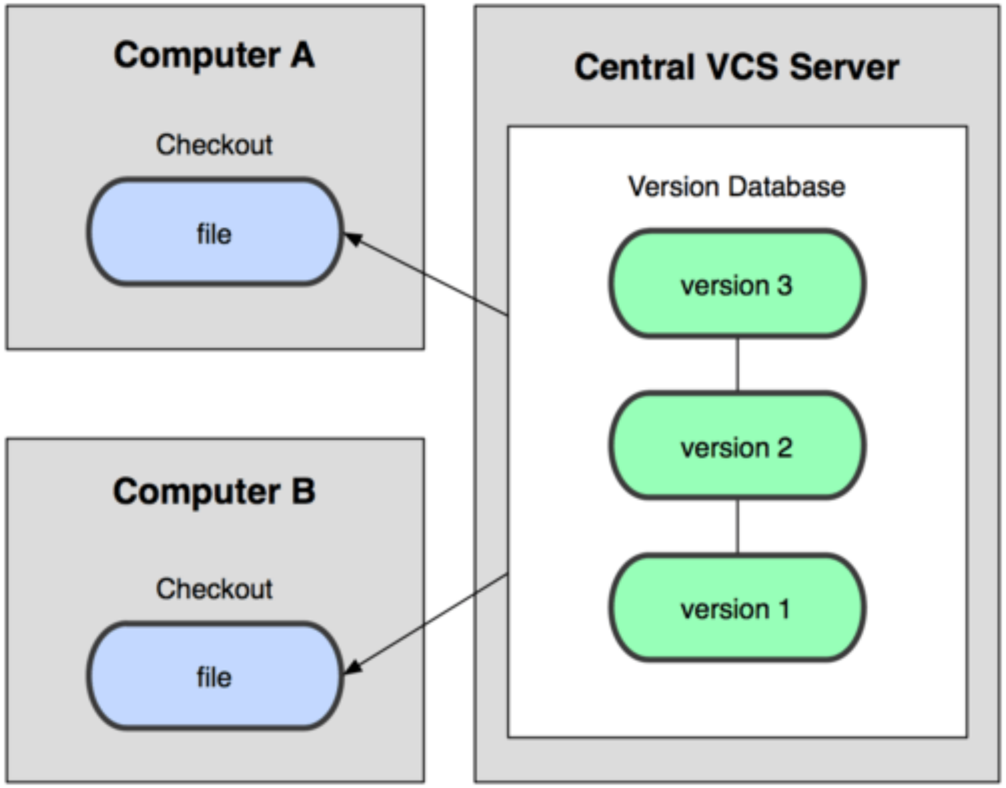
Version Control with Subversion Paperback by [C. Michael Pilato](), [Ben Collins-Sussman](), [Brian W. Fitzpatrick]()

Basic Work Cycle

- Update your working copy (check out your code)
- Make your changes (modify, add, remove, copy, move files/directories) on your working copy
- Review your changes you've made in your working copy
- Fix your mistakes (may start all over from unmodified state)
- Resolve any conflicts (merge others' changes)
- Publish (commit) your changes (lock and commit) Others can see your work, too!

Centralised Version Control

- One server holds the code base
- Clients access the server by check-in/ check-outs
- CVS, SVN

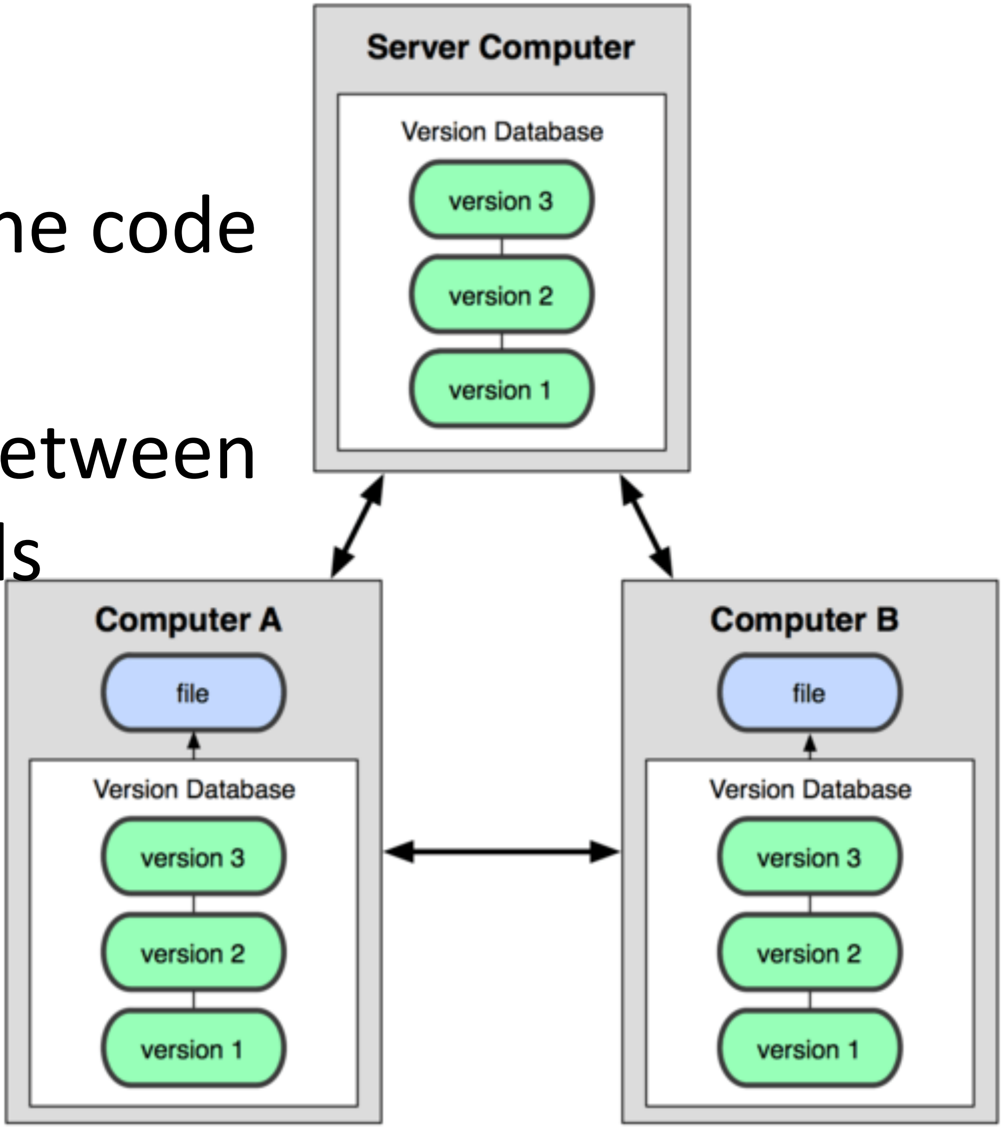


Pro Git, http://git-scm.com/book

What’s the problem with SVN/ CVS?

Distributed Version Control

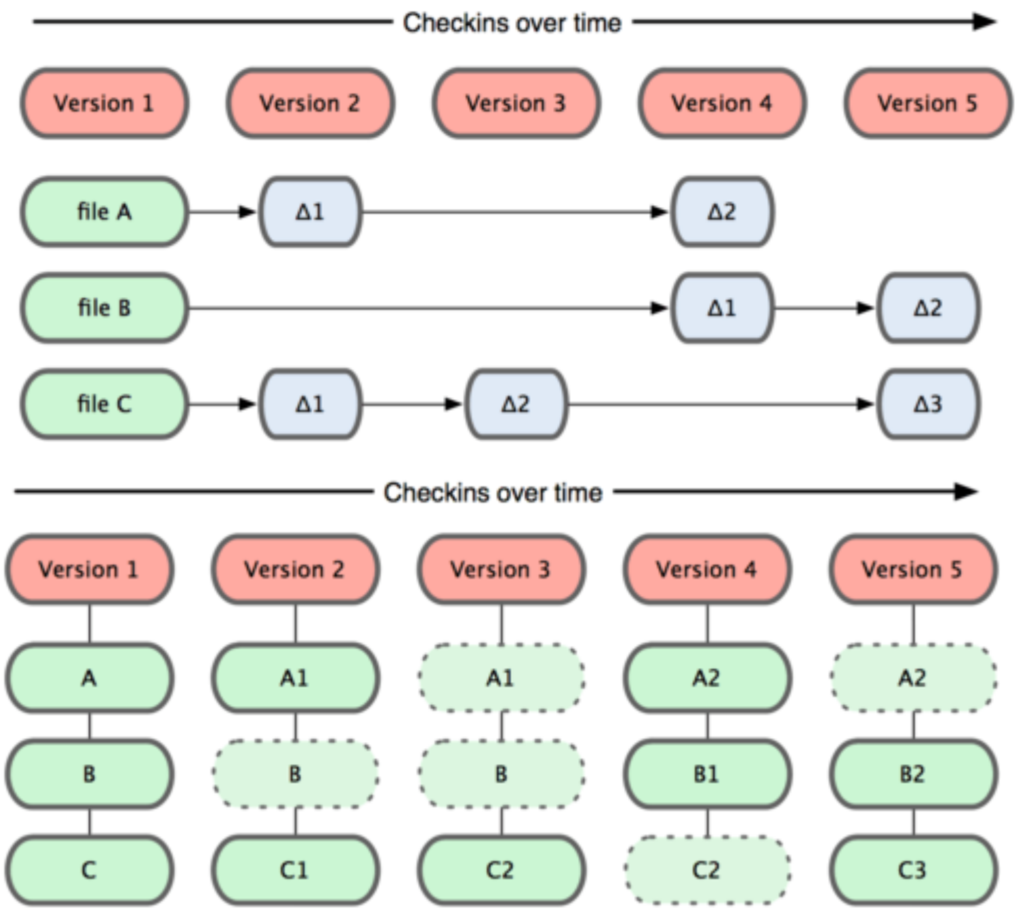
- Each client holds a complete copy of the code base
- Codes are shared between clients by push/pulls
- GIT



Pro Git, http://git-scm.com/book

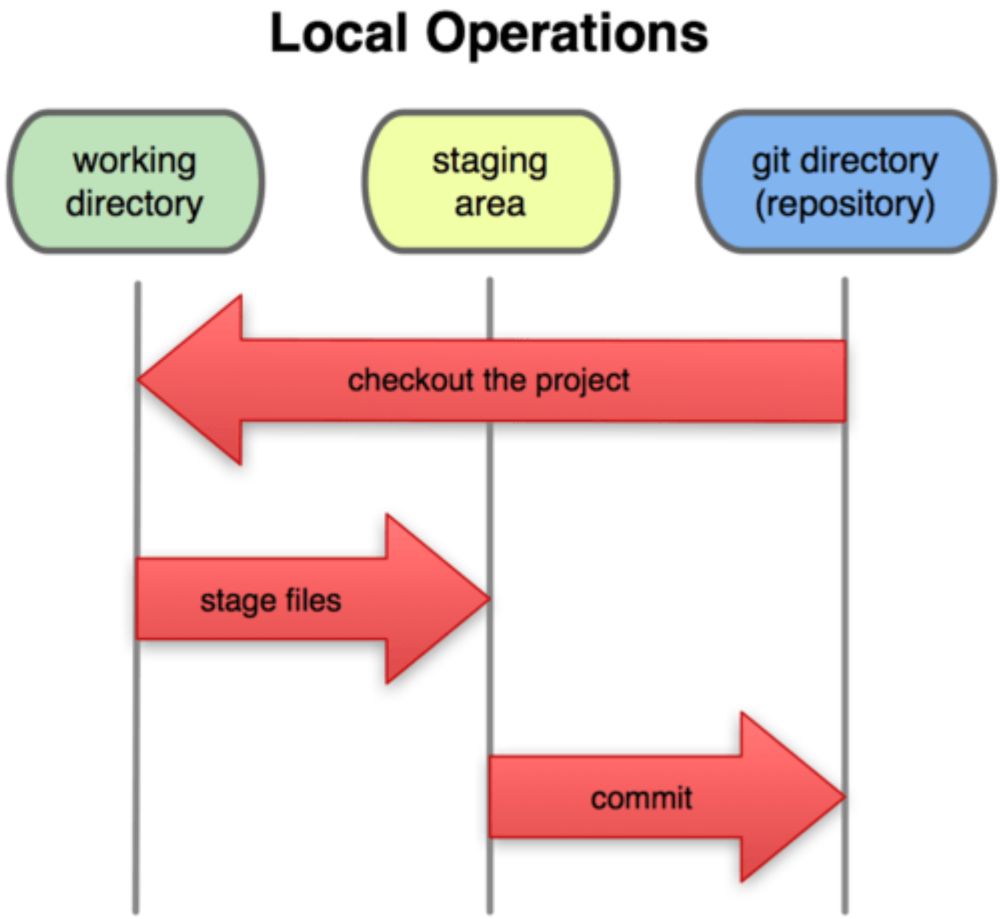
GIT

- Delta form vs. snapshot form



Pro Git, http://git-scm.com/book

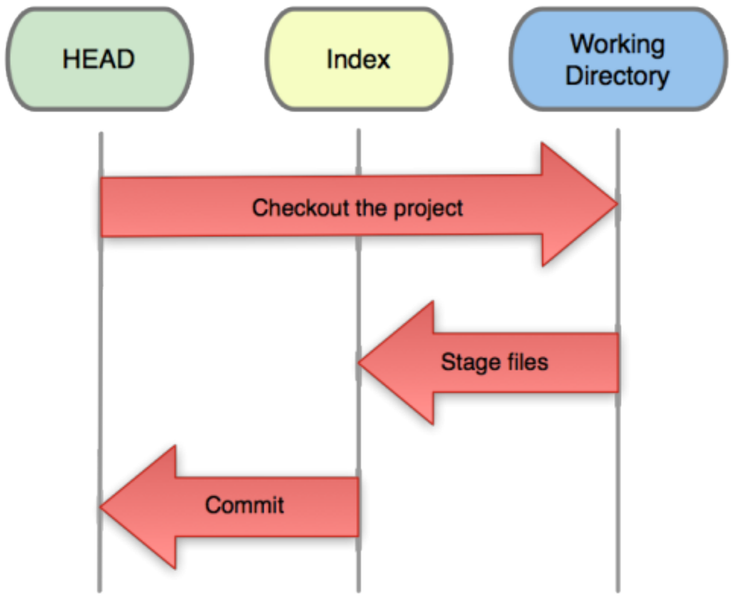
Local operations



Pro Git, http://git-scm.com/book

Basic GIT workflow

- Init a repo
- Edit files
- Stage the changes
- Review your changes
- Commit the changes

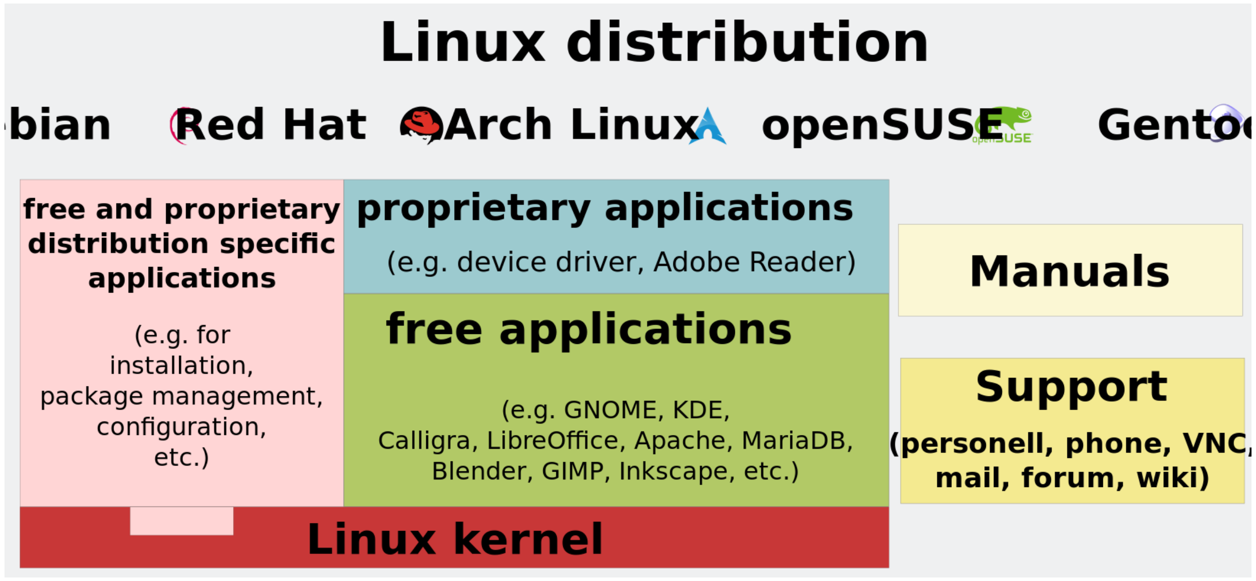


A case study of Linux

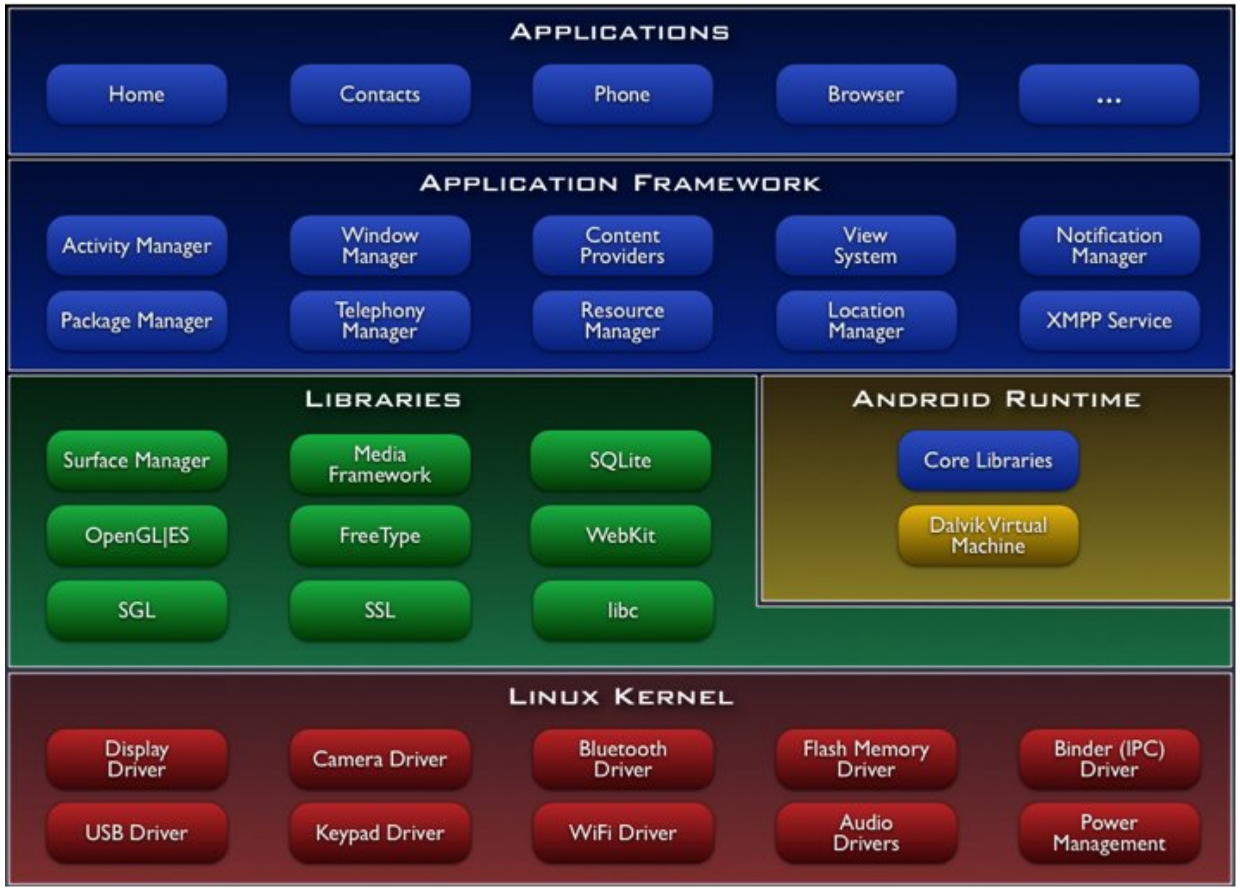
- Mainline
  - http://www.kernel.org/
- Other Linux Porting
  - ARM Linux (http://www.arm.linux.org.uk/)
- Linux versions (A.B.C)
  - A: kernel version
  - B: major revision (even-odd system version numbering system)
  - C: minor revision (every 2–3 months)

Linux Distribution

- Linux Distribution
  - Fedora Core
  - OpenSuSE
  - Debian
  - Ubuntu, Kubuntu
  - Gentoo
  - Slackware


Linux distribution
bian
Red Hat
Arch Linux
openSUSE
Gento
free and proprietary distribution specific applications
(e.g. for installation, package management, configuration, etc.)
proprietary applications
(e.g. device driver, Adobe Reader)
free applications
(e.g. GNOME, KDE, Calligra, LibreOffice, Apache, MariaDB, Blender, GIMP, Inkscape, etc.)
Manuals
Support
(personell, phone, VNC, mail, forum, wiki)
Linux kernel

Android


Applications
Home
Contacts
Phone
Browser
...
Application Framework
Activity Manager
Window Manager
Content Providers
View System
Notification Manager
Package Manager
Telephony Manager
Resource Manager
Location Manager
XMPP Service
Libraries
Surface Manager
Media Framework
SQLite
OpenGL|ES
FreeType
WebKit
SGL
SSL
libc
Android Runtime
Core Libraries
Dalvik Virtual Machine
Linux Kernel
Display Driver
Camera Driver
Bluetooth Driver
Flash Memory Driver
Binder (IPC) Driver
USB Driver
Keypad Driver
WiFi Driver
Audio Drivers
Power Management