Linux-tiny
And Directions For Small Systems

Matt Mackall
mpm@digeo.com
July, 2004

The Problem: Kernel Bloat

The Problem: Kernel Bloat

- 1994:
  0.99 kernel
  16MHz 386SX
  4MB of RAM

The Problem: Kernel Bloat

- 1994:
  0.99 kernel
  16MHz 386SX
  4MB of RAM
- 2004:
  2.6 kernel
  1024 node IA64
  > 1TB of RAM

The Problem: Kernel Bloat

- 1994:
  0.99 kernel
  16MHz 386SX
  4MB of RAM
- 2004:
  2.6 kernel
  1024 node IA64
  > 1TB of RAM
- not happy with 4MB of RAM any more

What Happened?

What Happened?

- massive boom in personal computing and internet use

What Happened?

- massive boom in personal computing and internet use
- constant decrease of hardware cost

What Happened?

- massive boom in personal computing and internet use
- constant decrease of hardware cost
- Linux became a serious commercial endeavor

What Happened?

- massive boom in personal computing and internet use
- constant decrease of hardware cost
- Linux became a serious commercial endeavor
- kernel hackers got jobs (and big machines on their desks)

But there are still small machines!

But there are still small machines!

- embedded devices are getting increasingly sophisticated

But there are still small machines!

- embedded devices are getting increasingly sophisticated
- hand-held computers are far more powerful than the original Linux desktops

But there are still small machines!

- embedded devices are getting increasingly sophisticated
- hand-held computers are far more powerful than the original Linux desktops
- and people around the world still rely on trailing edge “legacy” machines

Where is the growth?

Where is the growth?

- accumulates change by change

Where is the growth?

- accumulates change by change
- features: wholly new code

Where is the growth?

- accumulates change by change
- features: wholly new code
- performance: trading size for speed

Where is the growth?

- accumulates change by change
- features: wholly new code
- performance: trading size for speed
- duplication: trading size for laziness

Where is the growth?

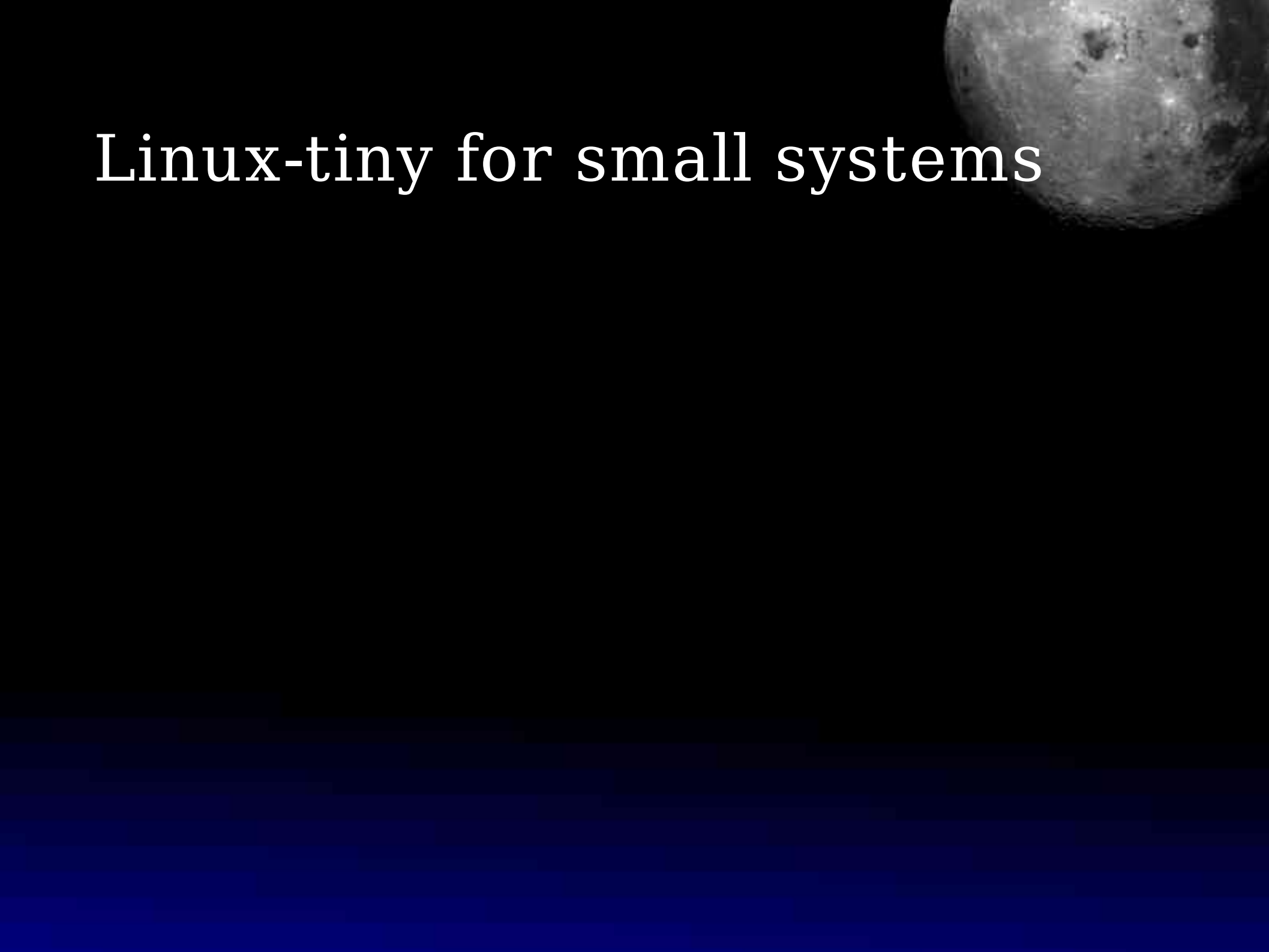
- accumulates change by change
- features: wholly new code
- performance: trading size for speed
- duplication: trading size for laziness
- compatibility: covering more bases

Where is the growth?

- accumulates change by change
- features: wholly new code
- performance: trading size for speed
- duplication: trading size for laziness
- compatibility: covering more bases
- correctness: fixing the corner cases

Where is the growth?

- accumulates change by change
- features: wholly new code
- performance: trading size for speed
- duplication: trading size for laziness
- compatibility: covering more bases
- correctness: fixing the corner cases
- migration: supporting ever more interfaces


Linux-tiny for small systems

Linux-tiny for small systems

- a collection of patches against 2.6

Linux-tiny for small systems

- a collection of patches against 2.6
- independently configurable

Linux-tiny for small systems

- a collection of patches against 2.6
- independently configurable
- minimally invasive and self-contained

Linux-tiny for small systems

- a collection of patches against 2.6
- independently configurable
- minimally invasive and self-contained
- mergeability is a priority

Linux-tiny for small systems

- a collection of patches against 2.6
- independently configurable
- minimally invasive and self-contained
- mergeability is a priority
- emphasis on debugging, auditing, and other support for embedded developers

Linux-tiny for small systems

- a collection of patches against 2.6
- independently configurable
- minimally invasive and self-contained
- mergeability is a priority
- emphasis on debugging, auditing, and other support for embedded developers
- initial target: minimal x86 web server

Finding bloat: using size(1) and nm (1)

```
2.6.5$ size vmlinux */built-in.o
   text    data     bss     dec     hex filename
3366220  673296  166824 4206340  402f04 vmlinux
1181276  250808   48000 1480084  169594 drivers/built-in.o
 735152   32593   30628  798373   c2ea5 fs/built-in.o
  18151    1120    1316   20587    506b init/built-in.o
  21841     172     204   22217    56c9 ipc/built-in.o
 159632   16115   42402  218149   35425 kernel/built-in.o
[...]
2.6.5$ nm --size -r vmlinux | head -20
00008000 b __log_buf
00007000 D irq_desc
00004e78 d pci_vendor_list
00004000 b bh_wait_queue_heads
00003c00 B ide_hwifs
0000213a T vt_ioctl
00002000 D init_thread_union
00001880 D contig_page_data
0000163b T journal_commit_transaction
00001500 b irq_2_pin
000012f5 T tcp_sendmsg
00001162 t n_tty_receive_buf
00001080 d per_cpu__tvec_bases
00001000 t translation_table
[...]
```

Finding bloat: measuring inlining

```
1560  get_current (1294 in *.c)
calls:
callers: <other>(336) capable(122) unlock_kernel(44)
lock_kernel(33) flush_tlb_page(11) flush_tlb_mm(10)
find_process_by_pid(6)
flush_tlb_range(4) current_is_kswapd(4) current_is_pdflush(3)
rwsem_down_failed_common(2) on_sig_stack(2) do_mmap2(2)
__exit_mm(2) walk_init_root(1) scm_check_creds(1)
save_i387_fsave(1)
sas_ss_flags(1) restore_i387_fsave(1) read_zero_pagealigned(1)
handle_group_stop(1) get_close_on_exec(1) fork_traceflag(1)
ext2_init_acl(1) exec_permission_lite(1) dup_mmap(1)
do_tty_write(1) de_thread(1) copy_signal(1) copy_sighand(1)
copy_fs(1) check_sticky(1) cap_set_all(1) cap_emulate_setxuid
(1) arch_get_unmapped_area(1)

546  current_thread_info (286 in *.c)
calls:
callers: <other>(207) copy_to_user(95) copy_from_user(86)
tcp_set_state(22) test_thread_flag(20) verify_area(13)
tcp_enter_memory_pressure(6) sock_orphan(3) icmp_xmit_lock(2)
csum_and_copy_to_user(2) tcp_v4_lookup(1) sock_graft(1)
set_thread_flag(1) neigh_update_hhs(1) ip_finish_output2(1)
gfp_any(1) fn_flush_list(1) do_getname(1) clear_thread_flag(1)
alloc_buf(1) activate_task(1)
```

Finding bloat: Vlasenko's inline hunter

```
 Size  Uses Wasted Name and definition
 ===== ==== ======
 ================================================
    56  461  16560 copy_from_user         include/asm/uaccess.h
   122  119  12036 skb_dequeue            include/linux/skbuff.h
   164   78  11088 skb_queue_purge        include/linux/skbuff.h
    97  141  10780 netif_wake_queue
include/linux/netdevice.h
    43  468  10741 copy_to_user           include/asm/uaccess.h
    43  461  10580 copy_from_user         include/asm/uaccess.h
   145   77   9500 put_page               include/linux/mm.h
    49  313   9048 skb_put                include/linux/skbuff.h
   109  101   8900 skb_queue_tail         include/linux/skbuff.h
   381   21   7220 sock_queue_rcv_skb     include/net/sock.h
    55  191   6650 init_MUTEX
include/asm/semaphore.h
    61  163   6642 unlock_kernel
include/linux/smp_lock.h
    59  165   6396 lock_kernel
include/linux/smp_lock.h
   127   59   6206 dev_kfree_skb_any
include/linux/netdevice.h
    41  289   6048 list_del               include/linux/list.h
    73   83   4346 dev_kfree_skb_irq
include/linux/netdevice.h
   131   39   4218 netif_device_attach
include/linux/netdevice.h
```

Finding bloat: tracking allocations

```
# cat /proc/kmalloc
total bytes allocated:   266848
slack bytes allocated:    37774
net bytes allocated:     145568
number of allocs:           732
number of frees:            282
number of callers:           71
lost callers:                 0
lost allocs:                  0
unknown frees:                0

   total    slack      net alloc/free  caller
     256      203      256     8/0     alloc_vfsmnt+0x73
    8192     3648     4096     2/1     atkbd_connect+0x1b
     192       48       64     3/2     seq_open+0x10
   12288        0     4096     3/2     seq_read+0x53
    8192        0        0     2/2     alloc_skb+0x3b
     960        0        0    10/10    load_elf_interp+0xa1
    1920      288        0    10/10    load_elf_binary+0x100
     320      130        0    10/10    load_elf_binary+0x1d8
     192       48       96     6/3     request_irq+0x22
    7200     1254     7200    75/0     proc_create+0x74
      64       43       64     2/0     proc_symlink+0x40
    4096      984        0     1/1     check_partition+0x1b
   69632        0    45056    17/6     dup_task_struct+0x38
```

Opportunities for trimming

Opportunities for trimming

- debugging interfaces: printk(), bug(), warn(), panic()

Opportunities for trimming

- debugging interfaces: printk(), bug (), warn(), panic()
- optional APIs: sysfs, ptrace, aio, posix-timers, uid16, vm86, ethtool, tcpdiag, igmp, rtnetlink, etc.

Opportunities for trimming

- debugging interfaces: printk(), bug (), warn(), panic()
- optional APIs: sysfs, ptrace, aio, posix-timers, uid16, vm86, ethtool, tcpdiag, igmp, rtnetlink, etc.
- memory structures: lookup tables, kernel stacks, etc.

Opportunities for trimming

- debugging interfaces: printk(), bug (), warn(), panic()
- optional APIs: sysfs, ptrace, aio, posix-timers, uid16, vm86, ethtool, tcpdiag, igmp, rtnetlink, etc.
- memory structures: lookup tables, kernel stacks, etc.
- replacing SLAB with SLOB

Opportunities for trimming

- debugging interfaces: printk(), bug (), warn(), panic()
- optional APIs: sysfs, ptrace, aio, posix-timers, uid16, vm86, ethtool, tcpdiag, igmp, rtnetlink, etc.
- memory structures: lookup tables, kernel stacks, etc.
- replacing SLAB with SLOB
- TinyVT: a minimal console

Results

Results

- historic 0.99pl15 kernel:
  Slackware 1.1.2, 1994 301K

Results

- historic 0.99pl15 kernel:
  Slackware 1.1.2, 1994 301K
- 2.6.5-tiny1 test config:
  IDE, ext2, TCP, NIC 363K
  boots with mem=2M (1664K w/ BIOS)

Results

- historic 0.99pl15 kernel:
  Slackware 1.1.2, 1994 301K
- 2.6.5-tiny1 test config:
  IDE, ext2, TCP, NIC 363K
  boots with mem=2M (1664K w/ BIOS)
- *Highly-modularized* vendor kernels:
  Fedora Core 2 1.2M
  SuSE 9.1 1.5M

Results

- historic 0.99pl15 kernel:
  Slackware 1.1.2, 1994 301K
- 2.6.5-tiny1 test config:
  IDE, ext2, TCP, NIC 363K
  boots with mem=2M (1664K w/ BIOS)
- *Highly-modularized* vendor kernels:
  Fedora Core 2 1.2M
  SuSE 9.1 1.5M
- Stock 2.6.5 default config 1.9M

A sample boot


A sample boot
Uncompressing Linux...

A sample boot

```
Uncompressing Linux... Ok, booting the kernel.
```

A sample boot

```
Uncompressing Linux... Ok, booting the kernel.
#
```

A sample boot

```
Uncompressing Linux... Ok, booting the kernel.
# mount /proc
```

A sample boot

```
Uncompressing Linux... Ok, booting the kernel.
# mount /proc
#
```

A sample boot

```
Uncompressing Linux... Ok, booting the kernel.
# mount /proc
# cat /proc/meminfo
```

A sample boot

```
Uncompressing Linux... Ok, booting the kernel.
# mount /proc
# cat /proc/meminfo
MemTotal:           980 kB
MemFree:            312 kB
Buffers:             32 kB
Cached:             296 kB
SwapCached:           0 kB
Active:             400 kB
Inactive:            48 kB
HighTotal:            0 kB
HighFree:             0 kB
LowTotal:           980 kB
LowFree:            312 kB
SwapTotal:            0 kB
SwapFree:             0 kB
Dirty:                0 kB
Writeback:            0 kB
Mapped:             380 kB
Slab:                 0 kB
Committed_AS:       132 kB
PageTables:          24 kB
VmallocTotal:  1032172 kB
VmallocUsed:          0 kB
VmallocChunk:  1032172 kB
#
```

Getting involved

- Much more to do
- Suggestions welcome!
- Project web page:
  http://selenic.com/tiny-about
- Mailing list:
  linux-tiny@selenic.com
  http://selenic.com/mailman/listinfo/linux-tiny