kboot
A Boot Loader Based on Kexec

http://kboot.sourceforge.net/

Werner Almesberger
werner@almesberger.net

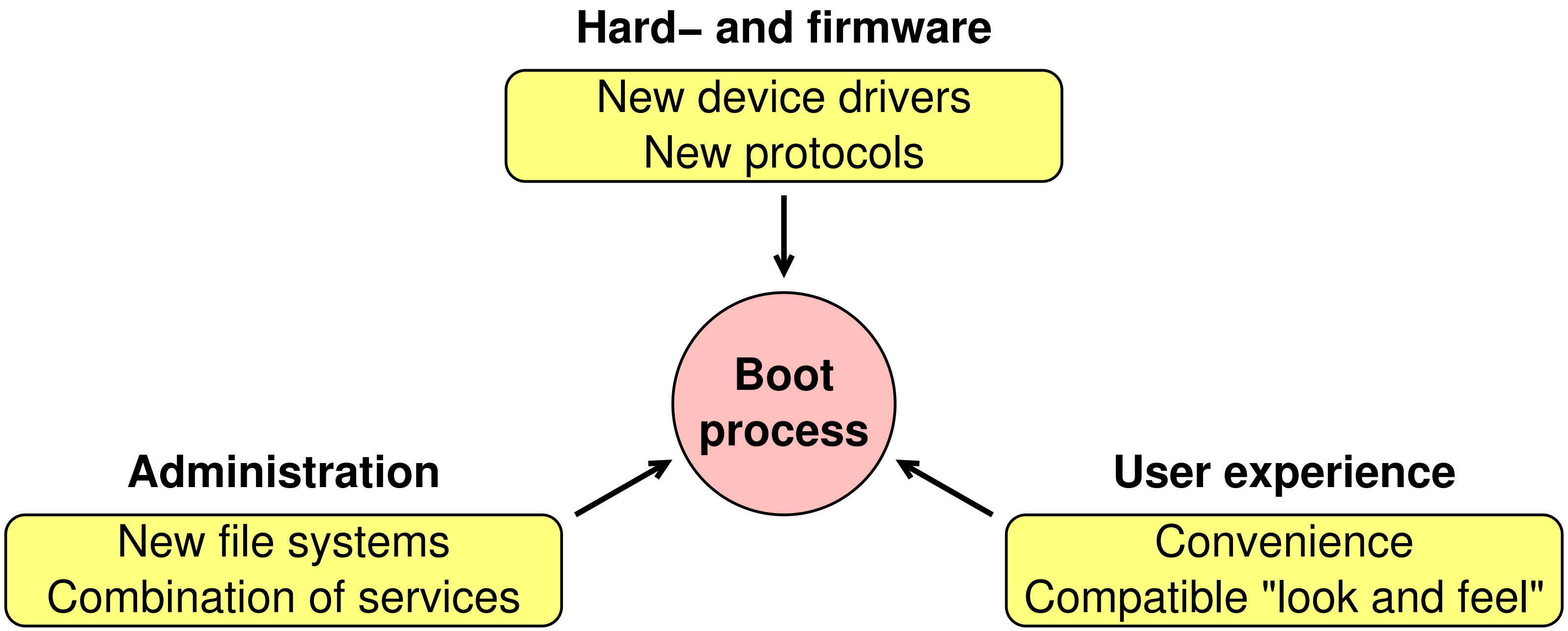
January 18, 2007

Why another boot loader ?

- Current boot loaders on Linux are feature-poor
  - Limited diagnostics and repair
  - Only support selected hardware
  - Cannot read all storage arrangements (file systems, RAID, etc.)
- Modern hardware tends to do more in software
  E.g., USB, Firewire, iSCSI
- Adding new features is hard
  - LILO: lots of assembler ...
  - GRUB: in C, but has its own world
- We can do better now

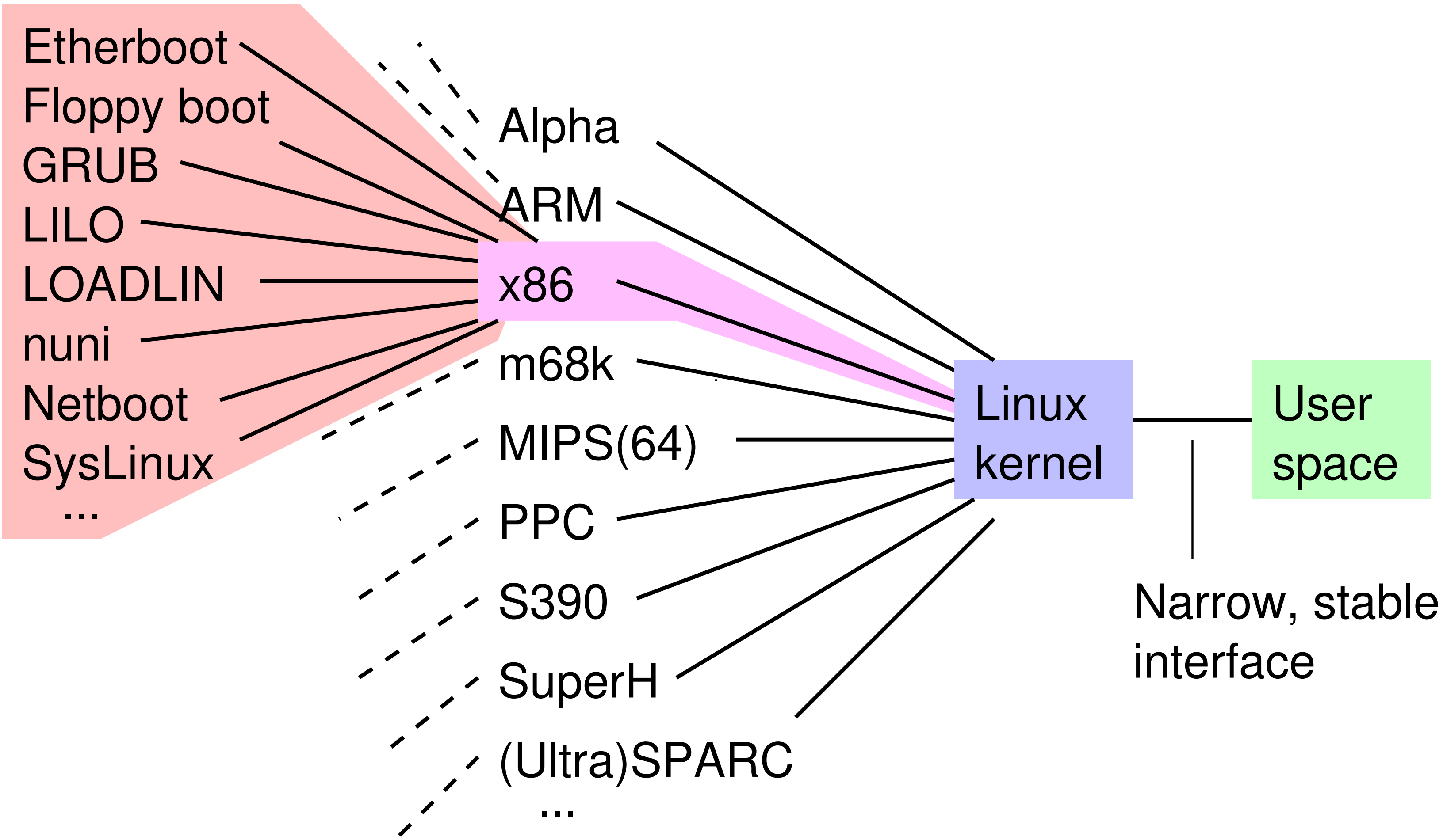
Requirements

How to handle all these requirements . . .



. . . without working up a sweat ?

Where to add features ?

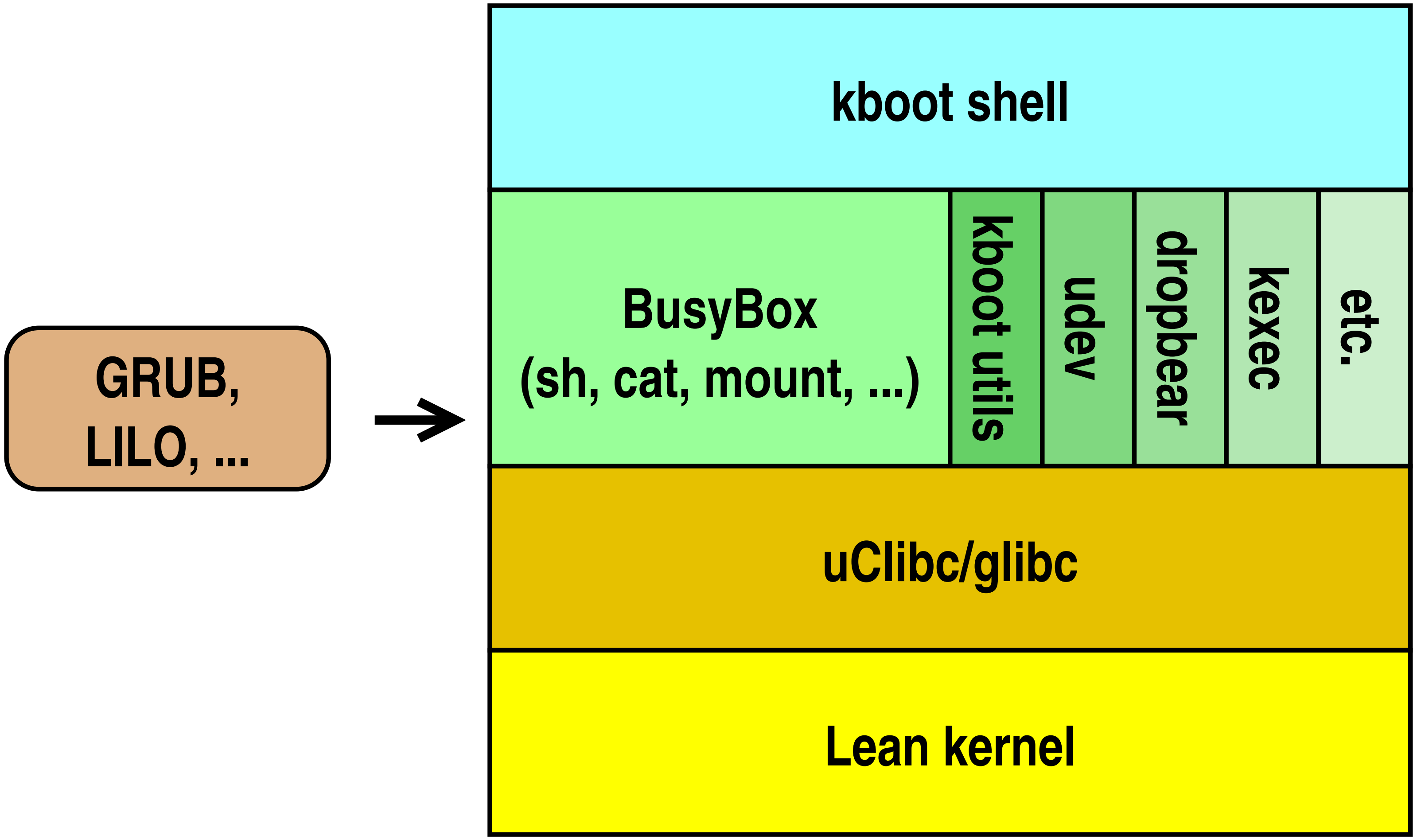


The Linux system

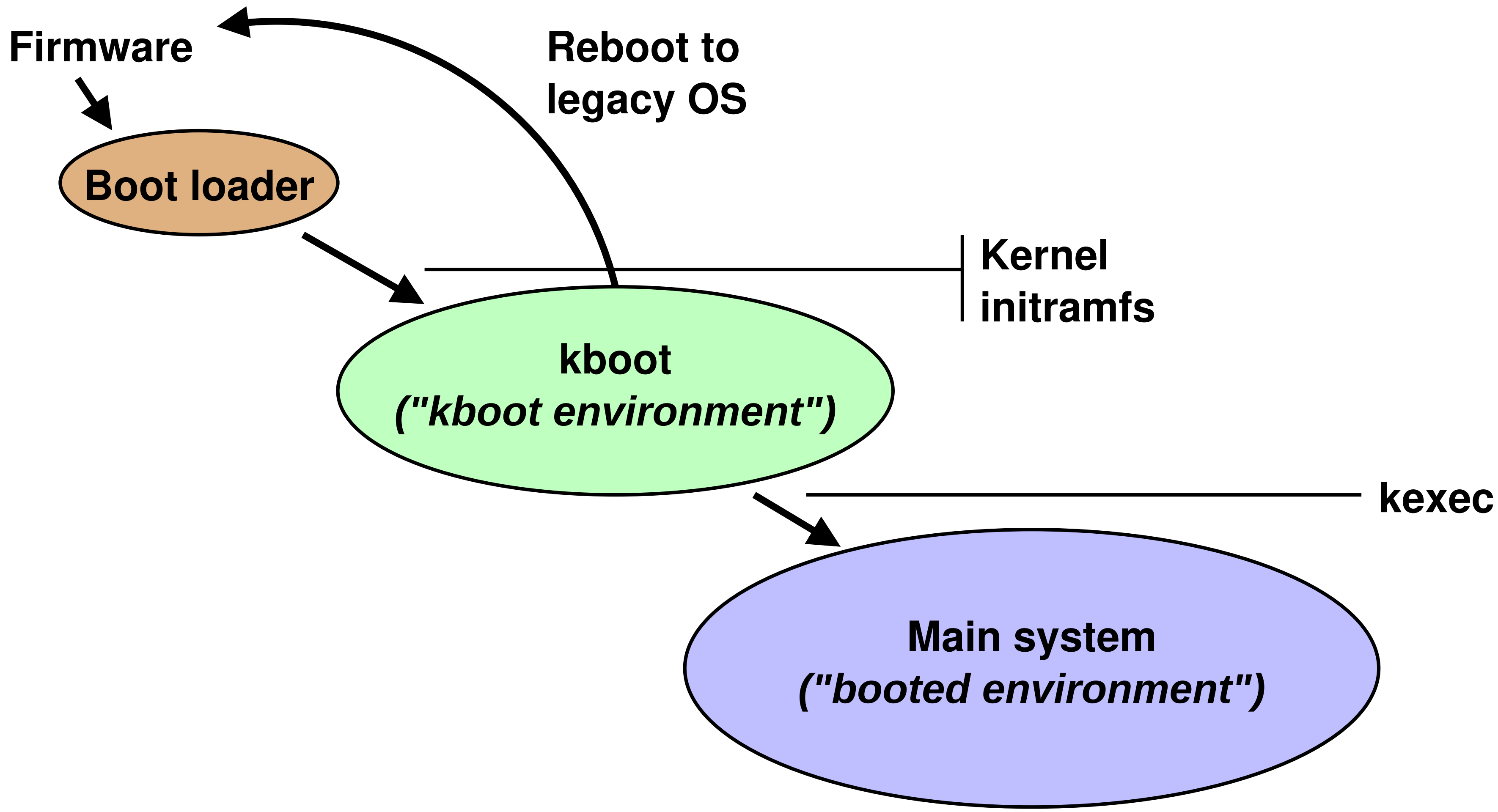
- Can reach (almost) all places where we want to put stuff (Kernel image, initramfs, etc.)
- Complete set of recovery tools
- Well-known user interface(s)
- Maintenance is somebody else's problem

⇒ It's the perfect boot loader

Heavy code reuse

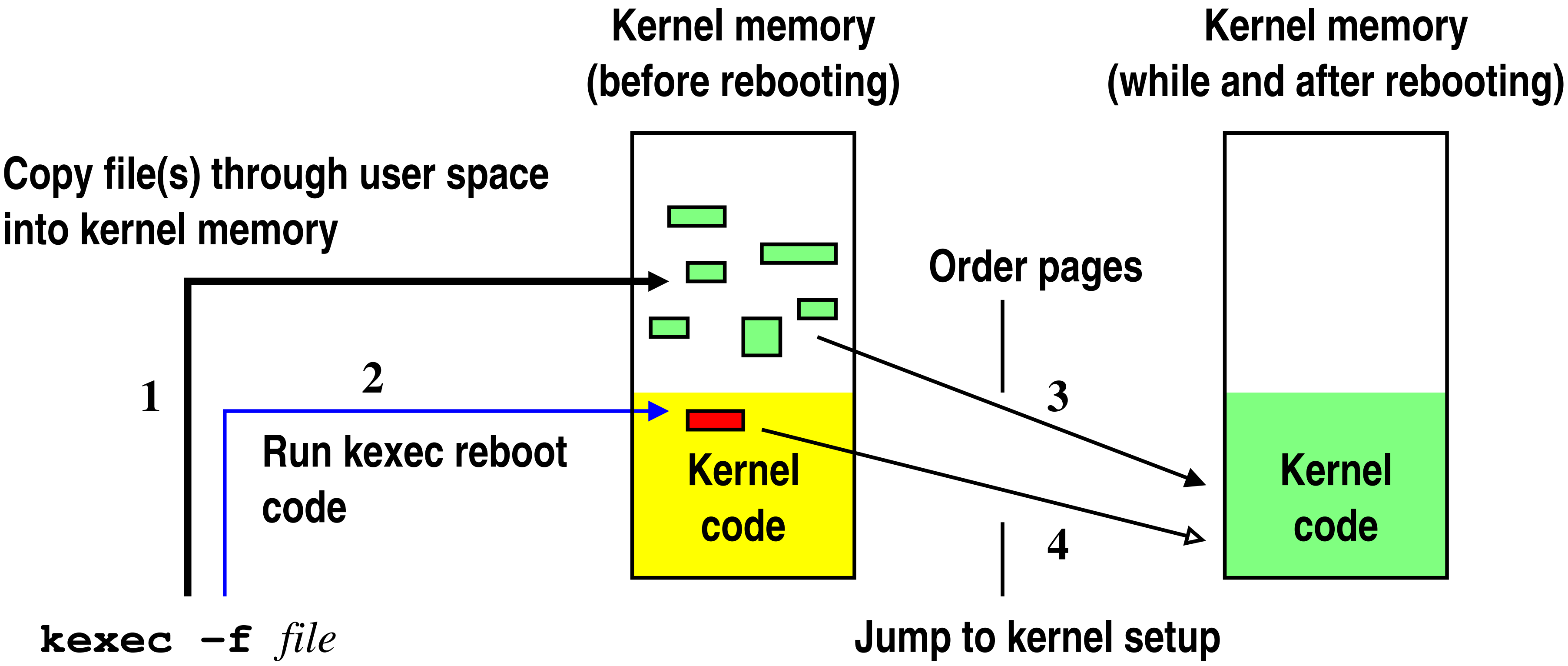


Boot sequence



kexec

By Eric Biederman.



kexec failures

Kexec fails if:

- Device drivers don't shut down properly
- Device drivers don't come back up properly
- Only the hardware/firmware can reset some devices

What this means for kboot deployment:

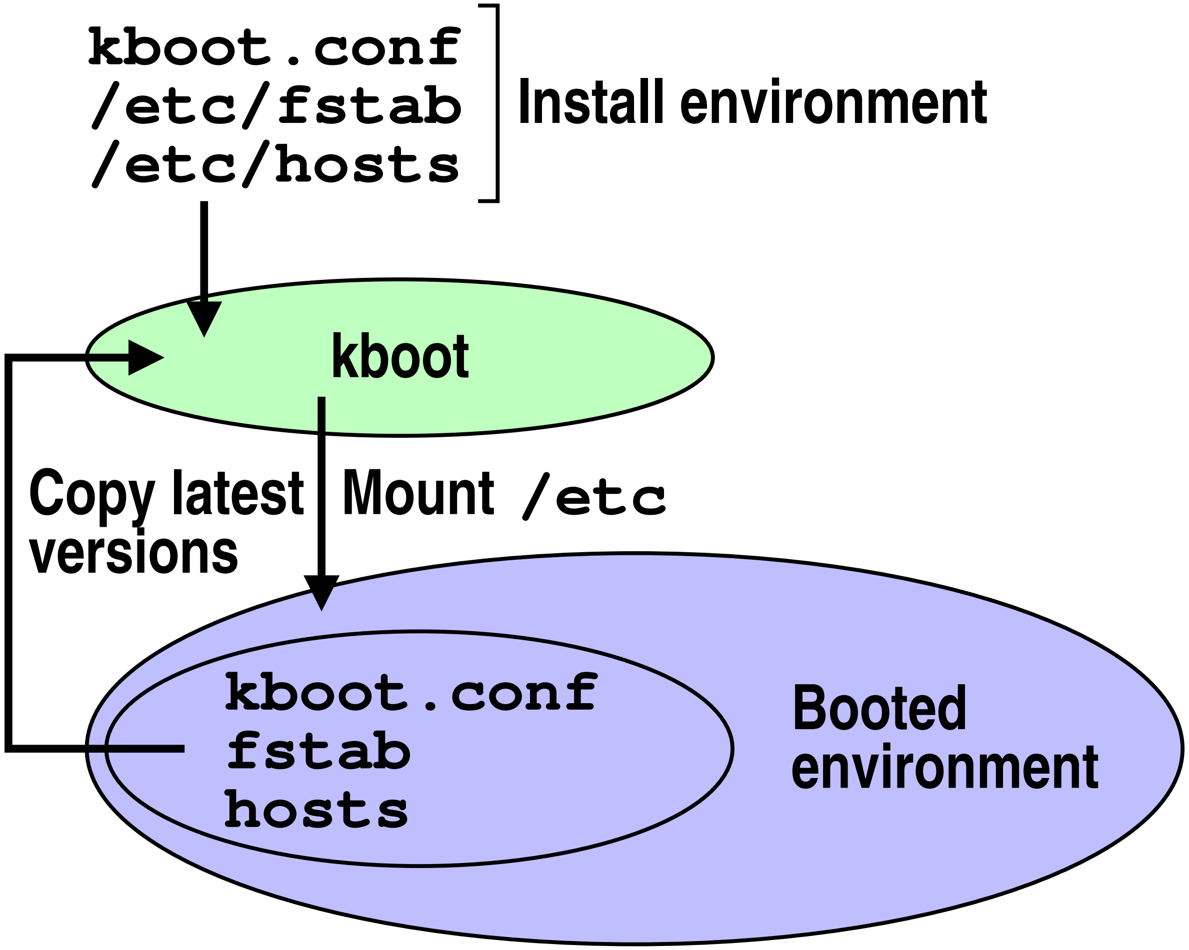
- There are <u>many</u> devices and drivers out there
- Only comparably few people have used kexec so far
- Kboot is (still) first "mass-market" application of kexec

$\Longrightarrow$ kboot will reveal driver bugs

kboot features

- Navigate with automounting
  - By path (enter booted environment), e.g., `/boot/bzImage-2.6.13.4`
  - By device, e.g., `/dev/hda2:/bzImage.26133`
  - NFS, e.g., `host.my.net:/stuff/kernel`
- Access (read) files on the network
  HTTP, FTP, TFTP (URL)
- Command execution
  - Local, mainly BusyBox
  - Chroot to booted environment
- Network configuration: DHCP, DNS
- Outbound and inbound SSH
- Timeouts, startup message, root device, initrd, keymap, ...

Configuration sources



Example session

Configure the network, with DHCP

Mount the root file system of the booted environment

Update `/etc/fstab`

Update `/etc/hosts`

`kboot: cd /boot`

`/etc/fstab`*:* `/boot` *is on* `/dev/hda1`

Mount `/dev/hda1` *on* `/mnt/root/boot`

`kboot /boot: ./bzImage initrd=http://www.mydoma.in/initrd.gz`

Download `http://www.mydoma.in/initrd.gz` *(wget)*

Look up `www.mydoma.in` *(*`/etc/hosts`*, DNS)*

Run `bzImage` *with kexec*

Future work (1)

- Configuration
  - Clean up organization
    E.g., build and installation should be separate
  - Use a configuration editor at build time, e.g., menuconfig
  - Persistent configuration changes at boot time
- Make extension easier
  - Allow "native" build (e.g., using glibc)
  - Simplify addition of user-provided packages
  - Converge with existing distribution
    E.g. OpenEmbedded or ROCK Linux
- Write regression tests

Future work (2)

- Support more environments
  - Other architectures than i386
    - “System Loader” (`kboot_sl`, by IBM) already supports S390
    - x86_64 sort of works
    - PlayStation 3 (PPC)
  - Include RAID, LVM, maybe SCP, SMB, ...
    (`kboot_sl` has EVMS)
  - Kernel modules
  - Direct booting of other operating systems
  - Multiple roots/file system hierarchies

Future work (3)

- Introduce device scan result prediction
  (Preliminary work by Dipankar Sarma)
- User interface
  - Be less chatty (kernel, daemons), more consistent diagnostics
  - Add a menu
  - Maybe add a splash screen (PS3 ?)
  - Rewrite in C (e.g., for better meta-character handling)
- Find a maintainer

Conclusion

- Proof of concept works
  - Not too big (can fit on a floppy)
  - A bit slow, but can be improved
- Extension is really easy
- First, stabilize and make prettier
- Then reap in the flexibility benefits

`http://kboot.sourceforge.net/`

Extra slides

Path names (1)

Paths anywhere in the file system hierarchy:

- Relative to the current directory
  `./bzImage`
- Relative to the kboot root
  `//tmp/foo`
- Relative to the root of the booted environment
  `/boot/bzImage-2.6.13.4`

Kboot consults `/etc/fstab` to automount directories.
E.g., `cd /home/user/kernel/v2.6` may first mount `/`, then `/home`.

Path names (2)

Paths on devices:

- With path to device file
  `/dev/hda2:/bzImage.26133`
- With name of device (under `/dev/`)
  `sda2:/subdir/file`
- Careful: `sda2` (and `/dev/sda2`)
  ... will (in the future) try to boot a legacy OS from `sda2`

Navigation with relative paths inside file systems, e.g.

```
kboot: cd sda2:
kboot /dev/sda2:/: cd subdir
kboot /dev/sda2:/subdir: _
```

Path names (3)

Paths and files in the network:

- NFS (full navigation)
  `host.my.net:/stuff/kernel`
- HTTP (download individual files)
  `http://host.faraw.ay/kernels/bzImage`
- FTP
  Like HTTP
- TFTP
  Like FTP and HTTP

Configuration files

Files read from `config/` at build time.

`kernel-config`	`.config` for the kernel
`fstab`	Data for automounting
`hosts`	Local hosts database
`passwd`	kboot user and password-based access
`authorized_keys`	Access without password
`ssh_host_{rsa,dsa}_key`, `dropbear_{rsa,dss}_host_key`	
	“Identity” of the kboot environment
`kboot.conf`	Configuration variables
`message`	Greeting message
language`.bkeymap`	Keyboard layout

Unified interactive configuration of system capabilities.

Configuration variables

Very similar to LILO:

`authorized_keys=`*path_to_file*	Access SSH without a password
`default=`*command*	Command for $\boxed{\text{Enter}}$ or timeout
`delay=`*seconds*	First input delay (short)
`initrd=`*path_to_file*	Initial RAM disk or RAM FS
`message=`*path_to_file*	Message file
`mount_rw=`*true_or_false*	Automount read-only or read-write
`restricted=`*true_or_false*	Only allow predefined commands
`root=`*path_to_file*	Root file system
`timeout=`*seconds*	General input delay (long)

All other variables are treated as labels/macros, e.g.,

```
my_kernel="/boot/bzImage-2.6.13.1 root=/dev/sda7"
```

Configuration sources (1)

- Basic capabilities set at build time
  E.g. whether to include DHCP support
- Configuration files on initramfs of kboot environment
  - Files from kboot's `config/` directory
    E.g. `kboot.conf`
  - Files from build host
    E.g. `/etc/fstab`
- Files copied from booted environment
  E.g. updates for the above files
- Manual settings and overrides
  E.g. `root=` or `initrd=` settings