



# *Hacking Linux Kernel*

Amerigo Wang  
Associate Software Engineer

# Where is Linux kernel?

- Official website: <http://www.kernel.org>
- Bug report: <http://bugzilla.kernel.org>
- Mail archive: <http://lkml.org>
- Patch work (new): <http://patchwork.kernel.org/>
- Git trees: <http://git.kernel.org/>
- kernel.org has more than just the kernel.



# Finding Kernel tar balls

- The latest versions information is located at:  
[http://www.kernel.org/kdist/finger\\_banner](http://www.kernel.org/kdist/finger_banner)
- 2.6 stable kernels and patches are available at:  
<http://www.kernel.org/pub/linux/kernel/v2.6/>
- -rc kernels and patches for 2.6 are in:  
<http://www.kernel.org/pub/linux/kernel/v2.6/testing/>
- 2.4 stable kernels and patches (maintained by Willy Tarreau) available at: <http://www.kernel.org/pub/linux/kernel/v2.4/>
- -rt kernel patches are here:  
<http://www.kernel.org/pub/linux/kernel/projects/rt/>

# Compiling Linux Kernel

- `tar -xvjf linux-2.6.x.y.tar.bz2`
- `cd linux-2.6.x.y`
- `cp /boot/config-`uname -r` .config`
- `make oldconfig`
- `make && make modules`
- `make modules_install && make install`

# Tips

- `yes "" | make oldconfig`
- `make menuconfig/make xconfig/make defconfig`
- `zcat /proc/config.gz >.config`
- `mkinitfs` for Ubuntu
- `mkinitcpio` for Archlinux
- `make randconfig/make allyesconfig`
- `make tags/make cscope`



**Linux supports more different  
processors than any other  
operating system ever has.**





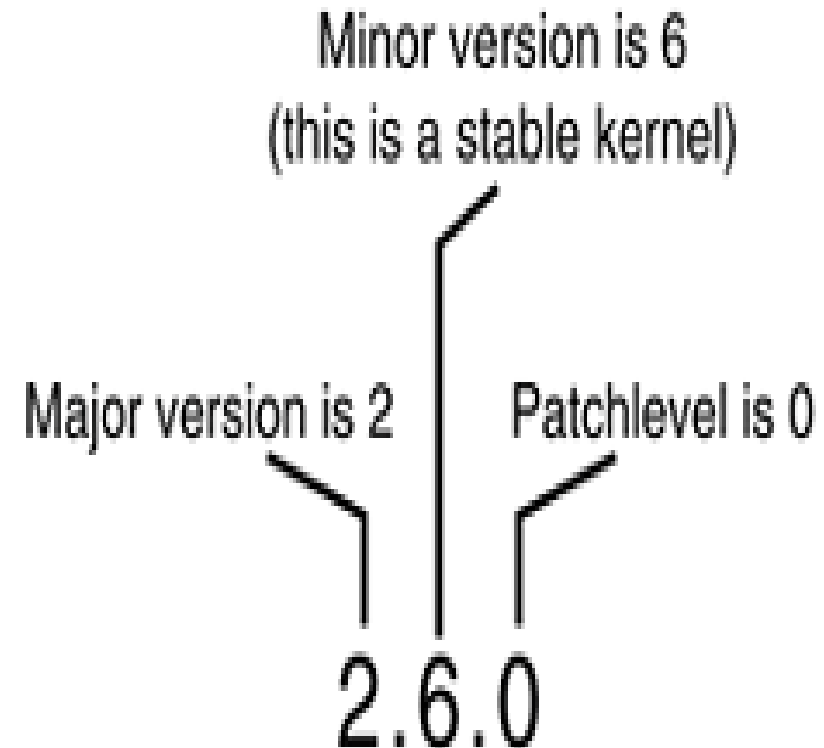
**Linux supports more devices,  
“out of the box”, than any other  
operating system ever has.**

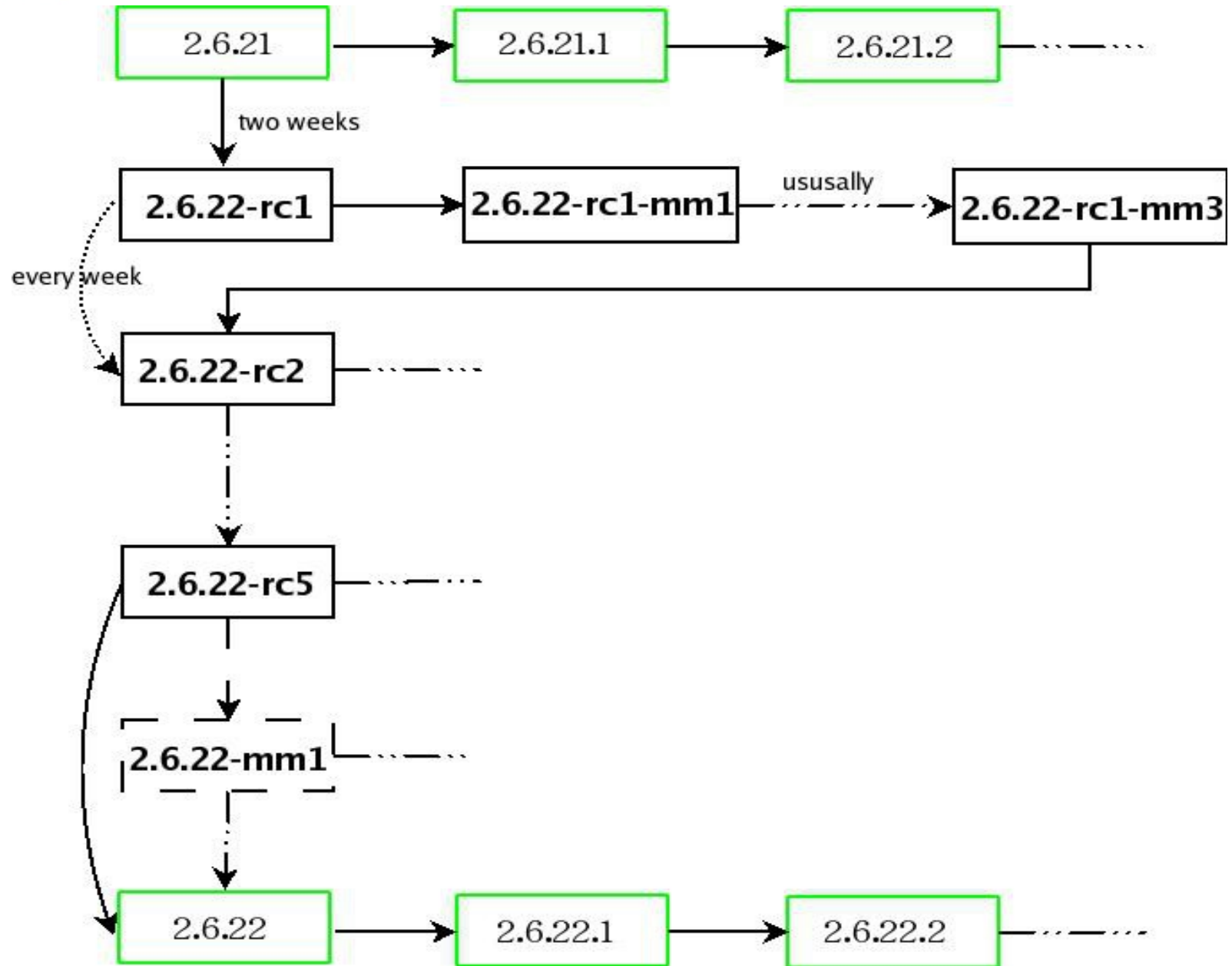


# *Linux Kernel Development*

# Kernel Versioning

- Up to version 2.5, the minor version number identified the type of kernel.
- This is not true since 2.6.
- 2.6.x(y) kernels are all stable release.
- 2.6.x kernels are the base stable releases released by Linus.
- 2.6.x.y kernels are -stable kernels released by stable team (Greg KH & Chris Wright), where 'y' denotes patch level.
- 2.6.x-rcX, 2.6.x-rcX-mmY, 2.6.x-mmX are development kernels.





## -mm tree and linux-next tree

- They are both staging trees, run on your own risk.
- You are welcome to test them.
- -mm is maintained by Andrew Morton, with quilt.
- -mmotm is a snapshot of Andrew's -mm patch queue, which is available at: <http://userweb.kernel.org/~akpm/mmotm/>
- linux-next is a snapshot of what the mainline is expected to look like after the next merge window closes, maintained by Stephen Rothwell, with git.
- <http://www.kernel.org/pub/linux/kernel/people/sfr/linux-next/>
- <http://linux.f-seidel.de/linux-next/pmwiki/>

# Kernel Development Model

- Changes are submitted as patches, with emails, to LKML.
- Patches are reviewed by other people and/or the maintainers.
- Patches are collected by relevant subsystem maintainers, or go to Andrew Morton by default.
- Subsystem maintainers submit their queued patches to Linus, Linus is the highest level maintainer.
- Anyone can submit patches to LKML. Trivial patches are also welcome.
- Stable releases are independent, [stable@kernel.org](mailto:stable@kernel.org)
- The kernel community is driven by credits and reputations.



Holy Linus Torvalds



Andrew Morton  
(-mm)



Greg KH  
(USB, drivers)



Ingo Molnar  
(x86, sched)



David Miller  
(network, sparc)



Alan Cox  
(TTY)



Rusty Russell  
(modules)



# Stable release

- Rule 1: It must be obviously correct and tested.
- Rule 2: It can not bigger than 100 lines, with context.
- Rule 3: It must fix only one problem that is critical.
- Rule 4: It can not contain any "trivial" fixes in it.
- Rule 5: It must be accepted by the relevant subsystem maintainer.
- When the -stable team decides for a review cycle, the patches will be sent to the review committee.
- The review committee has a short time in which to ACK or NACK the patch. The acked patches will be added to the latest -stable release.
- Depends on the major distributions, long-time supported stable releases are selected.

**“Open-source development  
violates almost all known  
management theories.”**

Dr. Marietta Baba,  
Dean of the College of Social Science,  
Michigan State University



# *Submitting patches*

# Patch

- A patch is a small text document containing a delta of changes between two different versions of a source tree.
- Patches are created with the 'diff' program (or git-diff, quilt etc.)
- ``diff -up`` or ``diff -Nrup`` is recommended.
- Demo
- Apply a patch with the ``patch`` program (or `git-applypatch`).
- Patches for the Linux kernel are generated relative to the parent directory holding the kernel source dir.
- Demo

# Patches for kernel releases

- Stable kernel patches apply to the base kernels.
- Base kernel release patches only apply to the previous base kernel version.
- Incremental patches upgrade from a specific release to the next release.
- The -rc patches are not incremental, they apply to a base 2.6.x kernel.
- The -mm kernels apply to either a base 2.6.x kernel or to a Linux -rc kernel.
- Example

# Submitting Your Patches

- Before you submit patches, please do read these files: Documentation/Changes, Documentation/CodingStyle, Documentation/SubmittingPatches, Documentation/SubmittingDrivers.
- “Patches are delivered via email only. Downloading them from internet servers is a pain.” -- Andrew Morton
- Check your email client. Read Documentation/email-clients.txt.
- Select the right destinations, Cc list. Always Cc the maintainers. Run scripts/get\_maintainer.pl.
- The major list: [linux-kernel@vger.kernel.org](mailto:linux-kernel@vger.kernel.org)
- Make your patch in a right format accepted by the community.

# Patch Format

- Subject format: [PATCH \$version \$n/\$total] \$subsystem: one-line summary
- Email body contents: description, your patch and your Signed-off-by.
- If your patch is too big, split it into pieces, send them separately.
- Always make your email text/plain.
- No MIME, no links, no compression and no attachments, if possible.
- Run scripts/checkpatch.pl to check your patch before sending.
- Example.

# Getting Your Patch Merged

- Be nice to every reviewers, especially the maintainers. :-)
- Keep on following your threads.
- Update your patch upon other people's suggestions, send the next version of it.
- If no one answers you, wait for some days and resend, with more relevant people Cc'ed.
- Making your patch description more detailed can also help.
- Be persistent, be patient, please.

# Tools

- Git
- Quilt
- mutt/alpine
- wiggle

*What else I can do*

# Testing Development Kernels

- Test the latest Linus tree, current -mm tree, everyday's linux-next tree.
- LTP is a good tool.
- Report bugs, regressions, or other problems to lkml.
- Or file a bug report in kernel bugzilla.
- Try your best to reproduce the bug.
- Bisect the git tree.
- If you can, try to fix it.

# Reviewing Patches

- Careful code reviews can prevent bugs.
- Review the patches that you are interested in.
- Test them by hands if possible.
- Put feedbacks to the author.
- Keep on moving with the author.

# Getting Started

- Trivial fixes or coding style fixes are a good start.
- [kernelnewbies.org](http://kernelnewbies.org)
- Fix compiler warnings, sparse warnings.
- Read the source code. :)
- Subscribe lkml.
- Alan Cox: “Ignore everyone who tells you kernel hacking is hard, special or different. It’s a large program, and bug fixing or driver tweaking can be a best starting point. It is however not magic, nor written in a secret language that only deep initiates with beards can read.”



# *References*

# References

- Greg KH, *Linux Kernel in a Nutshell*, O'Reilly, 2006.
- Greg KH, *Myths, Lies, and Truths about the Linux kernel*, [http://www.kroah.com/log/linux/ols\\_2006\\_keynote.html](http://www.kroah.com/log/linux/ols_2006_keynote.html)
- Documentation/development-process/\*

*Thanks mate! Happy Hacking!*

Questions?