Xen in Linux

Jeremy Fitzhardinge
jeremy.fitzhardinge@citrix.com
jeremy@goop.org

Overview

- Current state
- Upstreaming to Linux
- Development model
- How you can help

Current State

Default for xen-unstable

- Pvops dom0 is stable enough for general development use
- Now the default for xen-unstable
- No strong binding between xen and dom0 kernel
  - There are a few necessary tool-stack changes
  - May need to be Xen changes

Current pvops features

- Basic stuff all works:
  - Netback
  - Blkback
- Machine check
- Microcode update
- PCI passthrough
- Netchannel2
- ACPI processor control
- PV clock vsyscall (domU/0)

Missing features

- Blktap2 is the big one
  - Some preliminary work, but has gone stale
  - Question on how to handle PG_FOREIGN
- Other stuff:
  - Crashdump
  - PV drivers for HVM
  - Oprofile
  - pvscsi
  - Pvusb
  - ...?

TODO

- Performance still needs attention
  - Seems to be ~10-20% hit vs 2.6.18-xen
- Wider testing
  - Suspect save/restore needs some love
  - fbdev
- Balloon-up (work in progress, odd bug)
- Unify with Xen Client

TODO (2)

- DRM/graphics drivers
- Sort out Linux/Xen PAT incompatibility
- MTRR emulation?
- PV ticket spinlocks
  - First attempt at pvlocks not very efficient

Other Arches?

- IA64?
  - dom0?
- ARM?
  - Upstream at all?

Further Out

- More speculative ideas, which may or may not be good:
- Move all ACPI handling into Xen
- Dedicated “pciback domain”
  - So all devices look like pci passthrough, even for dom0
  - Avoids mixing models
  - Makes a stronger case for isolation, dom0 no longer part of TCB
- HVM dom0?

Upstreaming

A Quick History

- Xen project did initial Linux port
- Patches not viewed favorably by kernel community
  - VMWare also proposed VMI
- Breakthrough: paravirt-ops layer (pv-ops)
  - Xen
  - VMI
  - Lguest
- DomU support released in mainline early 2008

PV-ops today

- Has proved generally useful
- Refactored to allow multi-platform support
- Lots of useful hooks for research/experiments
- KVM also using pv-ops
- Vmware proposed removal of VMI support
  - Allows minor cleanups of VMI-only hooks

DomU is uncontroversial

- DomU support has been in mainline for a long time
- Nobody is advocating removing it
- PV-ops is here to stay
- New domU features and bugfixes readily accepted

Dom0 is controversial

- Historically the patches have been very intrusive
- The dom0/Xen interface does not make it easy
- Comparison with kvm's relatively clean kernel impact inevitable
- Success requires similar cleanliness in the kernel's Xen infrastructure

What happened last time?

- Attempted “big push” approach to upstreaming
- Works well if there are no significant objections
- There were significant objections
  - Well, there was a lot of noise
  - But the actual areas of objection were quite small

Why Bother Upstreaming?

- We want Xen to be as easy to use as possible
- Only possible with wide distro support
- Even Xen-agnostic (at best) distros will include Xen support if easy to do so
- That requires the substantial core to be in mainline
    - “low impact” driver-like patches easier to deal with out of tree

Why no patches lately? Given up?

- No
- But there has been a shift of emphasis:
  - The two “consumers” of xen.git have different goals and needs

Xen Developers and Users

- Want feature completeness
- Prefer to use 2.6.18-xen for xen features
  - Despite many other limitations
  - To an extent
- Answer: get xen.git into good shape for them
- Switch xen-unstable userbase
- Consolidate development effort behind pvops

Linux Upstream Maintainers

- Don't really care about Xen
  - Some noisy objections don't make a consensus
- But don't particularly object either
  - So long as it doesn't get in their way
- Primary interest is cleanliness of patches
- Are not swayed by Xen user stories to overlook ugly corners of dom0 patchset

So what do we conclude?

- Linux upstream hard to satisfy
- Doing so is time-consuming
- Doesn't directly help Xen users
- But necessary in the long term

What's Plan B?

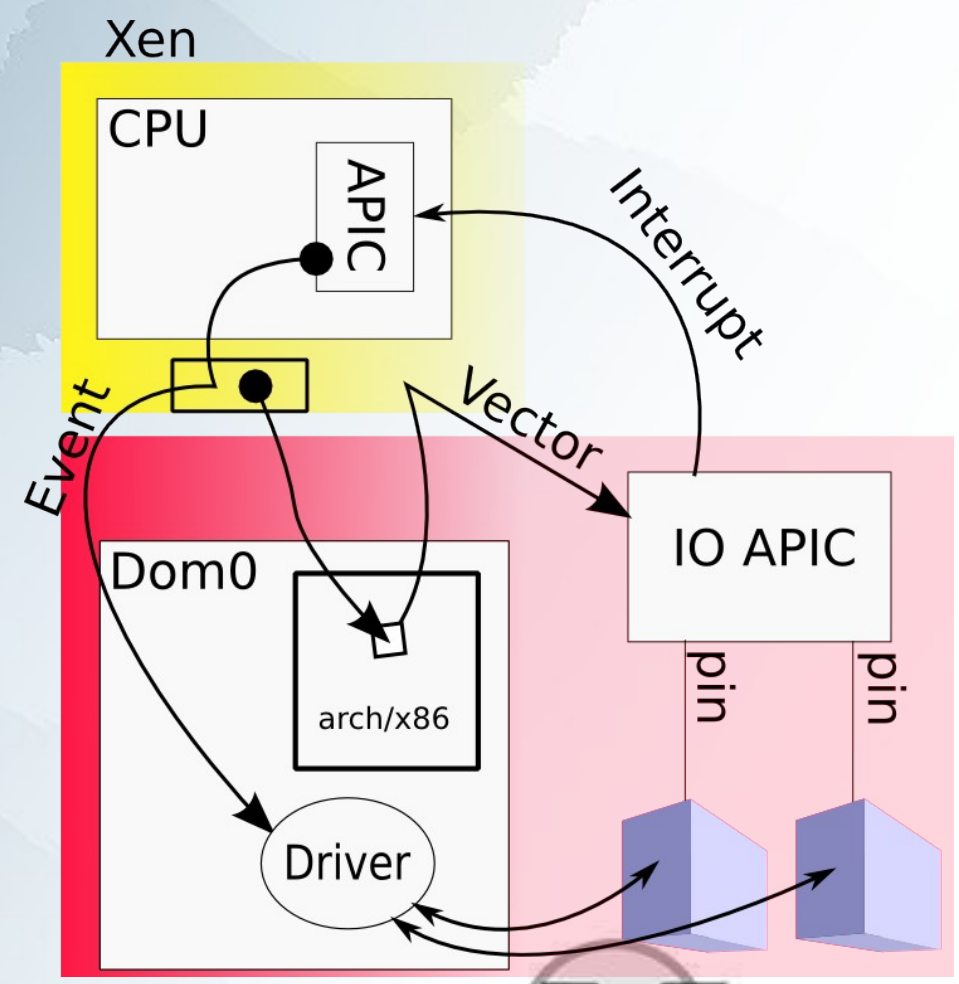
- So, focus on Xen users/developers first
  - Feature parity + more
  - Consolidate development
- While making sure that everything important is being tuned for upstream merging
- When there's a good set of dom0 functionality ready, upstream

Blockers

- Hardware interrupt routing/APIC programming
- SWIOTLB
- ACPI processor interface
- PG_foreign

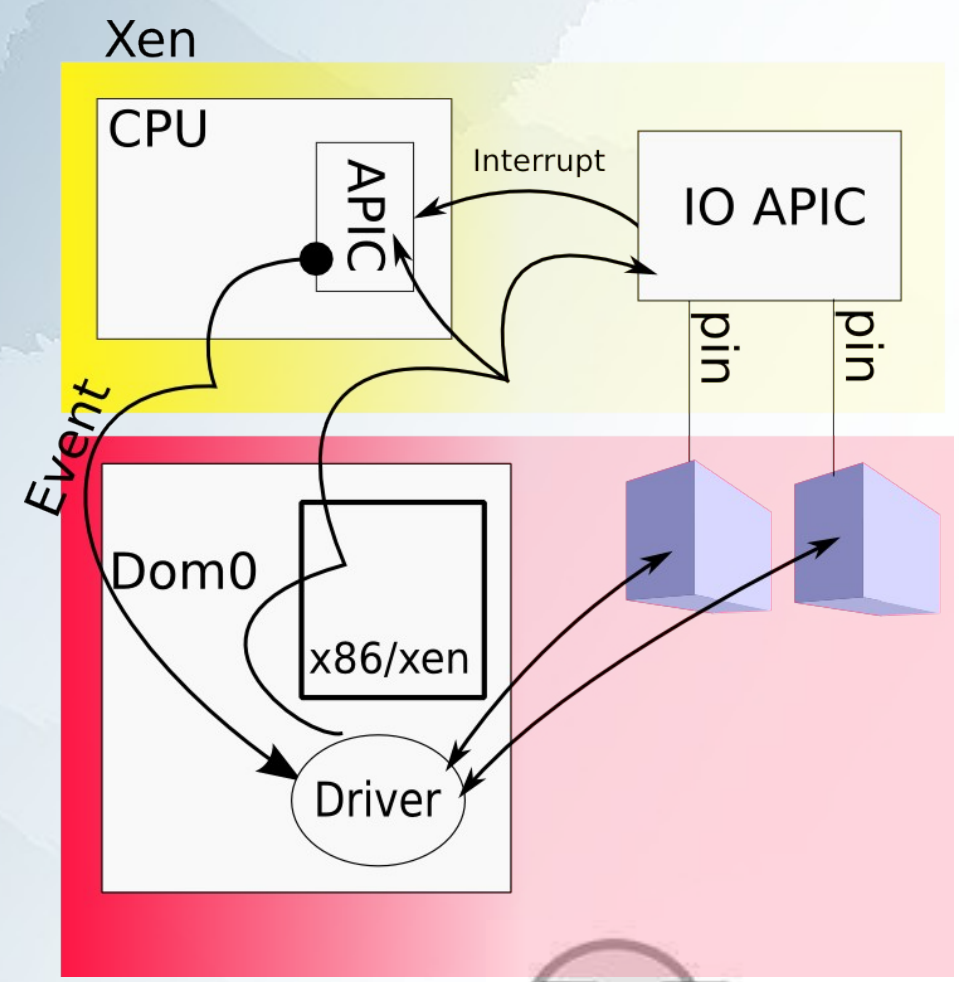
Current dom0 APIC model

- Complex
  - 4 xen/dom0 crossings
- Gets into Linux apic code
- Unecessary vector exposure
- Caused lots of lkml noise



Simplified Interrupt Model

- Only 2 xen/dom0 crossings
- Completely bypass x86 platform code
- No need for dom0 to program IOAPICs or know about vectors
- Works w/ MSI



SWIOTLB

- Not really any technical issue
  - Mostly a maintainer relations issue
- We need to:
  - Make sure swiotlb buffer is physically suitable
  - Do p2m/m2p translations on each page mapping
- Otherwise identical to existing swiotlb
- ...but maintainer doesn't want to allow hooks

SWIOTLB options

- Add hooks
  - Potential performance impact for non-Xen
- End up duplicating most swiotlb code
  - With some token, trivial sharing
- Duplicate the whole thing
- None really appealing
- Would also like to generalize to other IOMMUs
- Ideas?

ACPI processor interface

- Kernel ACPI code assumes kernel pCPU==ACPI CPU
- But vCPU != ACPI CPU
- Requires awkward hooks to reconcile
- Fundamentally, ACPI design hostile to Xen's
  - should ACPI all be in Xen?
- Hard to see how this can ever be merged as-is
  - But manageable as out-of-tree patch

PG_foreign

- Memory management of foreign (granted) pages is a general problem
- Affects netback, netchannel, blktap, etc
- PG_foreign page flag not acceptable for mainline
- No comprehensive alternative yet
- Things like skb destructors can solve it in an ad-hoc way
- I think the “splice” mechanism has potential

Timeline?
(I want it *now!*)

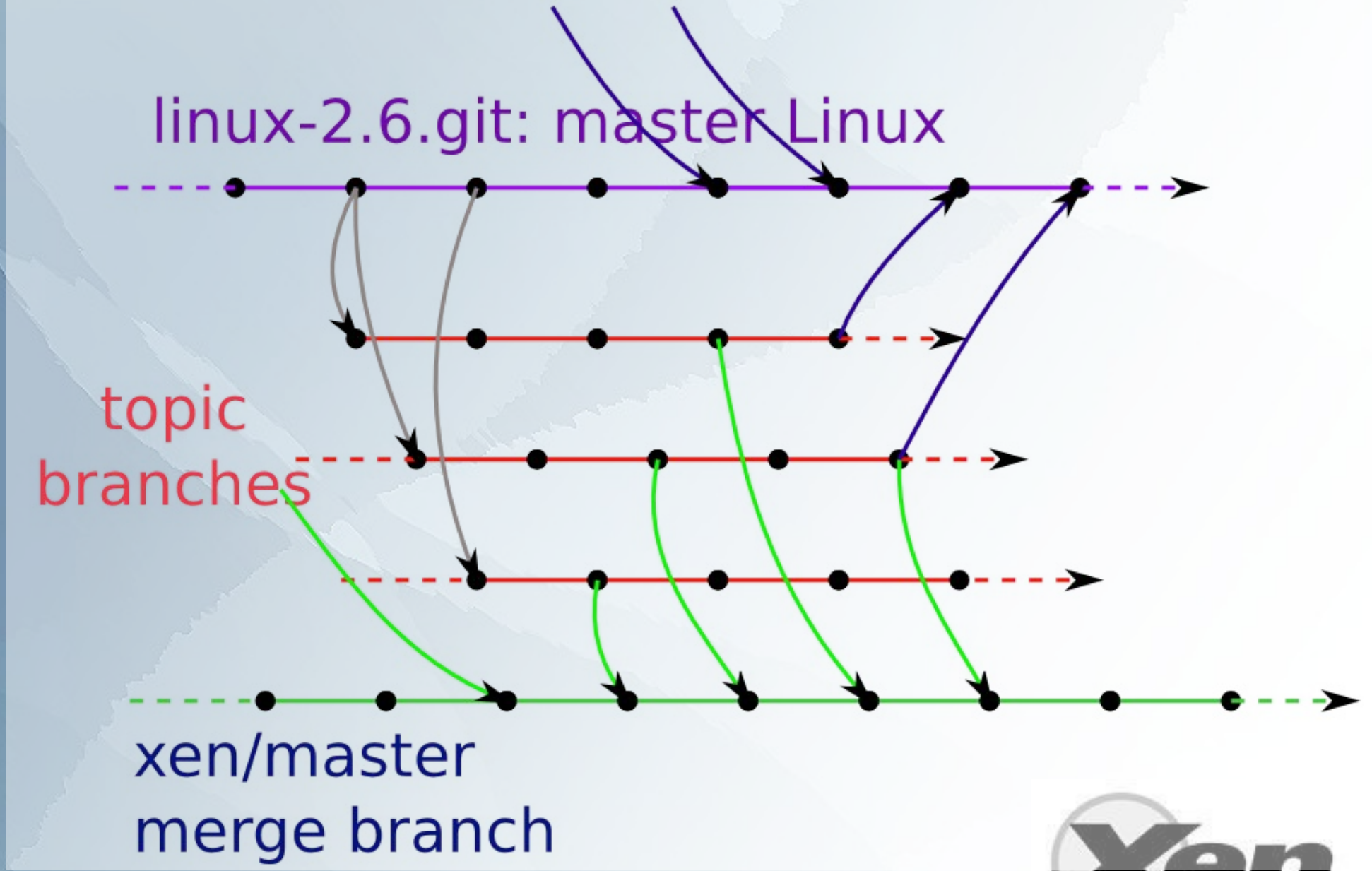
- No point pushing anything until at least basic dom0 boots and works
- We need to resolve at least APIC and SWIOTLB
- APIC is close to resolution
- SWIOTLB has more uncertainty
- At current progress, I would guess 2.6.33 merge window is a good target

Development Process

Xen.git

- Central repository for Xen/Linux development
- Acceptance policy: I'll take anything
  - Really, anything at all
  - But it may live off on a side branch until everyone is happy with it
- The goal is to make it as low-friction as possible to get useful work into the Xen community

xen.git Branch structure



linux-2.6.git

- Linus Torvald's master branch
  - “mainline”
  - Defines the current state of Linux
- Is the base of all subsequent changes
- Ultimate goal for a new changeset is to get merged here

Topic Branches

- Individual lines of development on their own branches
  - Typically one feature
- Minimal inter-dependencies
- Preferably based on mainline (linux-2.6)
- Merged together in xen/master
- Not necessarily individually bootable (but always compilable)

Merge Branches

- Topic branches are used by merging into a merge branch (xen/master)
- Merge branch should only contain merges
  - Occasionally have small fixup changes
- Merge branch should never be used as base
  - No topic branches rooted in merge
  - May be the base of another merge branch

xen.git topic branches

- xen/dom0/* - dom0 specific stuff
  - xen/dom0/core – essential to booting
  - xen/dom0/backend/* - backend drivers
  - .../apic* - interrupt stuff
  - .../pci – hardware driver support
- xen/* domU and dom0
  - xen/core
  - ...

Upstreaming

- Changes are always upstreamed from topic branch
- Ideally merge directly from topic into mainline
- But usually requires a fresh branch for cleanups
    - Inter-branch dependencies make this hard

Patch flow

- You write code
- You send it to me, xen-devel and any relevant maintainer
  - I will cc: missing maintainers if necessary
- If the patch looks basically sound and useful, I'll stick it into a topic branch
  - If it works (and breaks nothing else) I'll merge it
- When it makes sense, sent it upstream
  - Via appropriate maintainer

Know your maintainers

- The social aspects of kernel development are important
  - Big part of my job
- It's important to know who the maintainers are, what their views and opinions are
- Often worth changing technical content to route patches to favorable maintainers
- Ideal patch is purely Xen only, because then I can send it directly to Linus

Patch cleanup

- Patches/changes usually need cleaning
- Each change should do one logical thing
  - Never mix format cleanups with code changes
  - Make changes as small as possible, while being self consistent
  - Each change should compile incrementally
- No one change should touch Xen-specific and non-Xen code
  - In general, one subsystem at a time

Cleanup 2

- Always have complete, standalone, commit comment
  - Why is the change needed?
  - What does it do?
  - How does it do it?
- Passes scripts/checkpatch.pl
- diffstat -p1
- Always always have a Signed-off-by: line
  - I will never add this
  - Use the most official email address (company/employer/etc)

The Perfect Patch

- Is based on a linux-2.6 changeset
- Is already clean
- Only touches Xen-specific code
- I can pull it from directly from your git tree
- I can directly submit it to Linus for merging

From: Super Developer <super.developer@mycompany.com>
Subject: [PATCH] xen/fooble: undulate the fooble to avoid deadlock

This patch undulates the fooble to avoid the inherent deadlock.

This deadlock occurs in two curcumstances: once during the meta-undulation, but also when the gronkulack triggers the interrupt.

The obvious fix, smoothing the efflusion, doesn't work because it is also deadlock-prone.

The alternative is to undulate the fooble as this patch does, since it avoids the whole issue.

Signed-off-by: Super Developer <super.developer@mycompany.com>
Cc: Fooble Maintainer <dev@foobleco.com>

diff --git a/drivers/xen/fooble.c b/drivers/xen/fooble.c
index e2f1f15..b1c645d 100644
--- a/drivers/xen/fooble.c
+++ b/drivers/xen/fooble.c
@@ -1,4 +1,4 @@
...

How Can You Help?

Claim A Topic

- Find something that needs doing
- Do it
- Put your work in a git tree
- Send me pull requests
- But patches are OK too
  - But still, against a topic branch or mainline

Test Things

- Try things out
- Especially corner cases
- Root-cause the failures
- Supply fixes
  - Or at least tell the maintainer everything they need to fix it
- Test the fix!

Measure Things

- There's lots of things to be measured
- Use all the mechanisms available
- Some things may be unmeasurable
    - Add new instrumentation
- Work out how to make bad measurements better

Reporting Bugs

- Please, please, don't just send large raw log files
- *Always* say what you think has gone wrong
  - Even if you think it should be blindingly obvious
- (Same if you're showing that something has been fixed)

Also...

- When attaching logs, config files, patches etc:
  - Don't compress
  - Make sure mime type is text/plain
- Its always much easier if I can just read things in my mail viewer

Thanks!

- Konrad Wilk – swiotlb, pci passthrough
- Ian Campbell – lots of stuff
- Weidong Han – pci passthrough
- Xiantao Zhang – APIC redesign
- Yu Ke – ACPI processor interface
- Ke Liping, Yunhong Jiang – MCA, pcpu hotplug
- Steven Smith – netchannel2

Resources

- xen.git
  - Overview:
    - http://git.kernel.org/?p=linux/kernel/git/jeremy/xen.git;a=summary
  - Clone addr:
    - git://git.kernel.org/pub/scm/linux/kernel/git/jeremy/xen.git
    - http://git.kernel.org/smart/pub/scm/linux/kernel/git/jeremy/xen.git
- Wiki
  - http://wiki.xensource.com/xenwiki/XenParavirtOps