Migration v2

Andrew Cooper

Citrix XenServer

17^{th} August 2015

Why Migration v2

- XenServer 6.2
  - 64bit Xen, 32bit Dom0
  - Inertia, More efficient to virtualise

Why Migration v2

- XenServer 6.2
  - 64bit Xen, 32bit Dom0
  - Inertia, More efficient to virtualise
- XenServer 6.5
  - 64bit Xen, 64bit Dom0
  - High MMIO regions above 2^{44} bits

Why Migration v2

- XenServer 6.2
  - 64bit Xen, 32bit Dom0
  - Inertia, More efficient to virtualise
- XenServer 6.5
  - 64bit Xen, 64bit Dom0
  - High MMIO regions above 2^{44} bits
- Rolling Pool Upgrade tests
  - Migrate VM from XS6.2 to XS6.5
  - Error on the receiving side:

Why Migration v2

- XenServer 6.2
  - 64bit Xen, 32bit Dom0
  - Inertia, More efficient to virtualise
- XenServer 6.5
  - 64bit Xen, 64bit Dom0
  - High MMIO regions above 2^{44} bits
- Rolling Pool Upgrade tests
  - Migrate VM from XS6.2 to XS6.5
  - Error on the receiving side:

```
xc: detail: xc_domain_restore: starting restore of new domid 1
xc: detail: xc_domain_restore: p2m_size = ffffffff00010000
xc: error: Couldn't allocate p2m_frame_list array: Internal error
```

Legacy Migration

```
int xc_domain_restore(xc_interface *xch,
...

if ( RDEXACT(io_fd, &dinfo->p2m_size, sizeof(unsigned long)) )
{
  PERROR("read: p2m_size");
  goto out;
}
DPRINTF("%s: p2m_size = %lx\n", __func__, dinfo->p2m_size);
```

Legacy Migration

- No format written down
  - Subsequently reverse engineered from existing code

Legacy Migration

- No format written down
  - Subsequently reverse engineered from existing code
- No header information at all
- Hard to extend
  - Written mostly as two monolithic functions
  - goto tangle
  - PV MSR support too complicated to implement

Legacy Migration

- No format written down
  - Subsequently reverse engineered from existing code
- No header information at all
- Hard to extend
  - Written mostly as two monolithic functions
  - `goto` tangle
  - PV MSR support too complicated to implement
- Asymmetry with Qemu handling
  - Save side's caller puts Qemu blob into the stream
  - Restore side pulls Qemu blob out and saves in magic path

Legacy Migration

- No format written down
  - Subsequently reverse engineered from existing code
- No header information at all
- Hard to extend
  - Written mostly as two monolithic functions
  - goto tangle
  - PV MSR support too complicated to implement
- Asymmetry with Qemu handling
  - Save side's caller puts Qemu blob into the stream
  - Restore side pulls Qemu blob out and saves in magic path
- Stream contents depends on compilation ABI
  - **Different between 32bit and 64bit**

VM Serialisation

- Information (Currently x86 specific)
  - Common: Page Data, TSC
    - HVM: Params, Context (Xen serialised state)
      - PV: Width, Levels, P2M, VCPU State, Shared Info

VM Serialisation

- Information (Currently x86 specific)
  - Common: Page Data, TSC
    - HVM: Params, Context (Xen serialised state)
      - PV: Width, Levels, P2M, VCPU State, Shared Info
- Suspend
  - Pause VM
  - Copy all memory

VM Serialisation

- Information (Currently x86 specific)
  - Common: Page Data, TSC
  - HVM: Params, Context (Xen serialised state)
  - PV: Width, Levels, P2M, VCPU State, Shared Info
- Suspend
  - Pause VM
  - Copy all memory
- Migrate
  - Enable logdirty
  - Copy all memory
  - — Query logdirty bitmap
  - — Copy dirty memory
  - — Loop
  - Pause VM
  - Copy remaining memory

Solution for XenServer

- Redesigned completely from scratch

Solution for XenServer

- Redesigned completely from scratch
- Specification written down
  - `docs/specs/libxc-migration-stream.pandoc`
  - Describes exact binary layout
  - Extensible

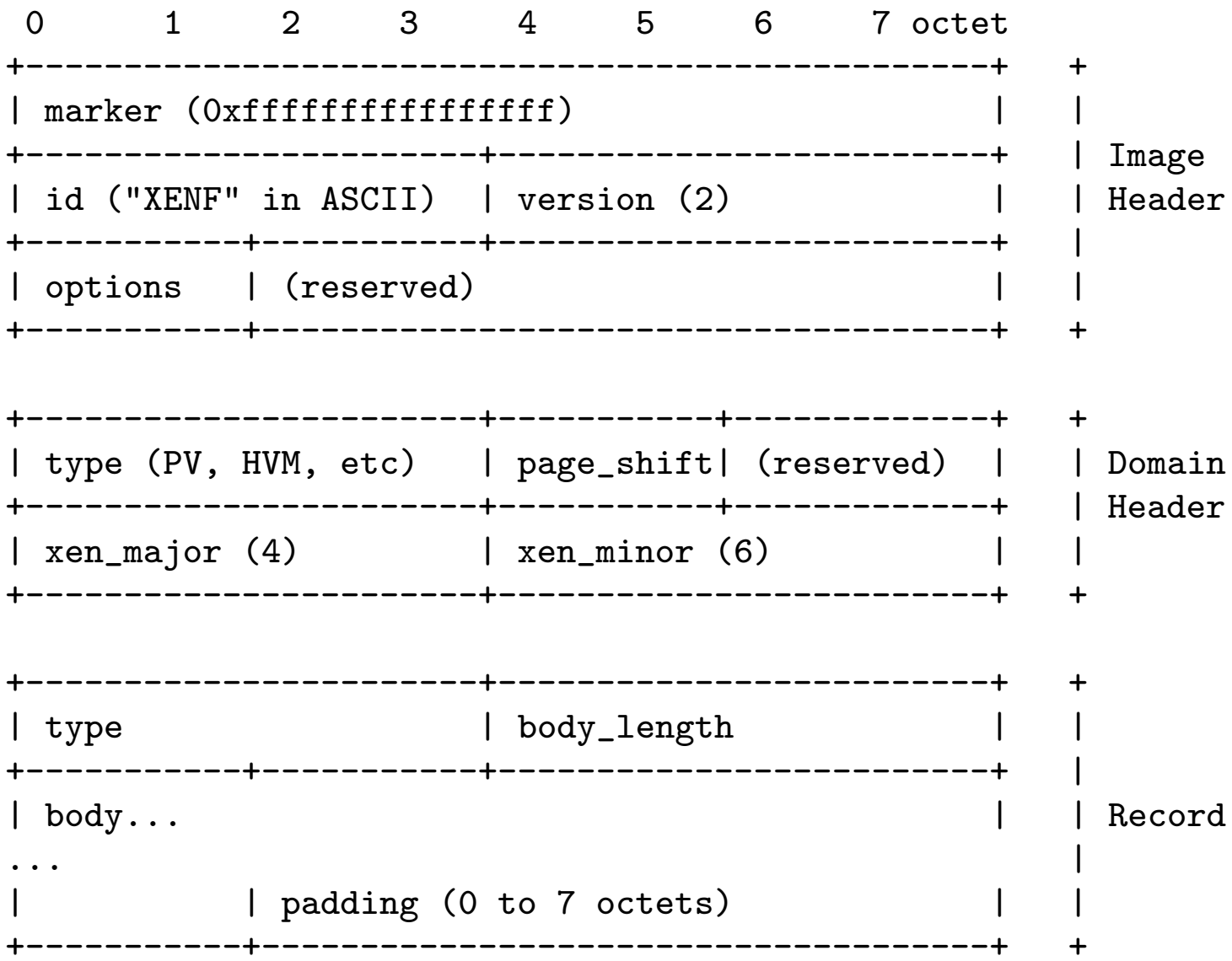
Solution for XenServer

- Redesigned completely from scratch
- Specification written down
  - `docs/specs/libxc-migration-stream.pandoc`
  - Describes exact binary layout
  - Extensible
- Reimplemented completely from scratch
  - Common save and restore algorithms
  - Per-guest-type hooks to implement

Solution for XenServer

- Redesigned completely from scratch
- Specification written down
  - `docs/specs/libxc-migration-stream.pandoc`
  - Describes exact binary layout
  - Extensible
- Reimplemented completely from scratch
  - Common save and restore algorithms
  - Per-guest-type hooks to implement
- Legacy conversion needed
  - `tools/python/scripts/convert-legacy-stream`
  - Reads in legacy stream
  - Writes out v2 stream

Stream format – libxc



Upstreaming

- Problems with libxl
  - No participation in stream
  - 'Toolstack Data' depends on compilation ABI

Upstreaming

- Problems with libxl
  - No participation in stream
  - 'Toolstack Data' depends on compilation ABI
- Design from scratch

Upstreaming

- Problems with libxl
  - No participation in stream
  - 'Toolstack Data' depends on compilation ABI
- Design from scratch
- Specification written down
  - `docs/specs/libxl-migration-stream.pandoc`
  - Describes exact binary layout
  - Extensible

Upstreaming

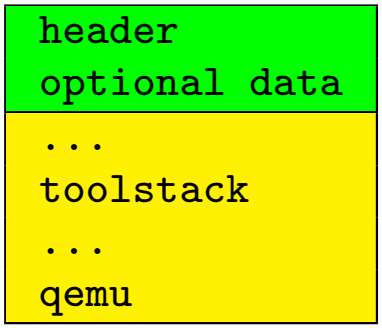
- Problems with libxl
  - No participation in stream
  - 'Toolstack Data' depends on compilation ABI
- Design from scratch
- Specification written down
  - `docs/specs/libxl-migration-stream.pandoc`
  - Describes exact binary layout
  - Extensible
- Write from scratch

Upstreaming

- Problems with libxl
  - No participation in stream
  - 'Toolstack Data' depends on compilation ABI
- Design from scratch
- Specification written down
  - `docs/specs/libxl-migration-stream.pandoc`
  - Describes exact binary layout
  - Extensible
- Write from scratch
- Compatibility script extended
  - Able to write libxl migration v2 streams
  - 'Qemu' and 'Toolstack data' layered appropriately

Framing

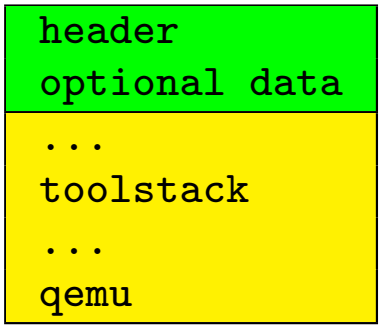
Legacy Migration



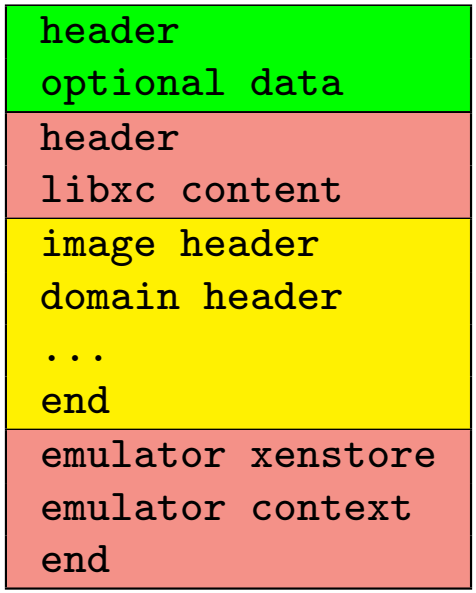
Key: xl libxl libxc

Framing

Legacy Migration



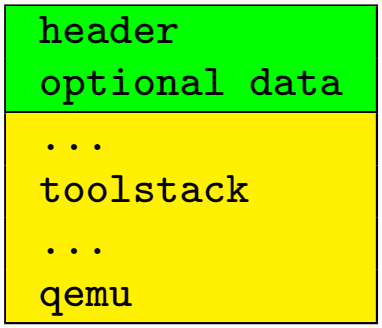
Migration v2



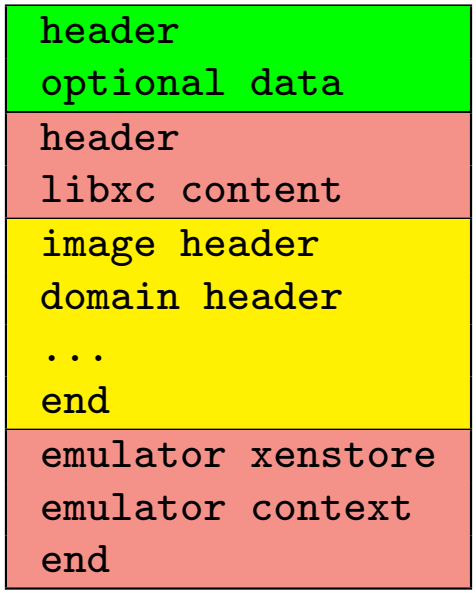
Key: xl libxl libxc

Framing

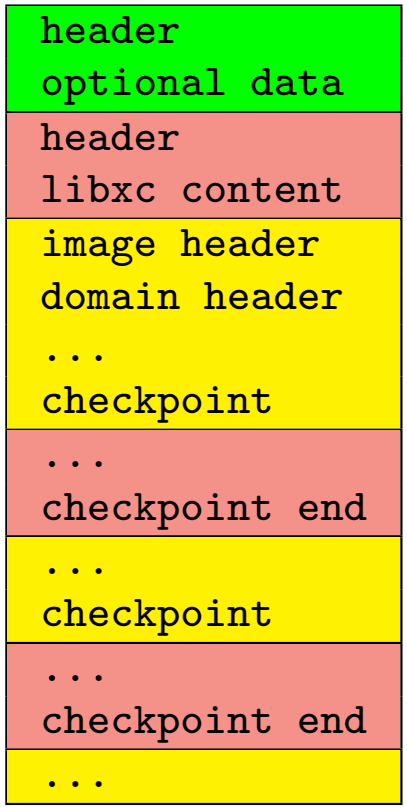
Legacy Migration



Migration v2



Remus Migration v2



Key: xl libxl libxc

General Notes

- Issues fixed
  - PV VCPU state corruption when racing with vcpu actions
  - PV guests with superpages abort on save, rather than failing to reconstruct pagetables on restore
  - More efficient handling of page data

General Notes

- Issues fixed
  - PV VCPU state corruption when racing with vcpu actions
  - PV guests with superpages abort on save, rather than failing to reconstruct pagetables on restore
  - More efficient handling of page data
- Issues still present
  - Guests which balloon
  - PV P2M structure changes
  - HVM guests with PoD pages

General Notes

- Issues fixed
  - PV VCPU state corruption when racing with vcpu actions
  - PV guests with superpages abort on save, rather than failing to reconstruct pagetables on restore
  - More efficient handling of page data
- Issues still present
  - Guests which balloon
  - PV P2M structure changes
  - HVM guests with PoD pages
- Areas for further work
  - Live migrate looping parameters
  - Linear P2M support

Status – Xen 4.6

- All committed
- Fully enabled (and tested)
- `xl save/restore/migrate/remus` function as before
- Legacy migration removed
- No noticeable difference to users

Migration v2

Any Questions?