#### XenSummit Asia

November 2-3, 2011 Seoul, Korea



## Prototype PV Drivers in SeaBIOS with upstream qemu

Daniel Castro Velasco
POSTECH

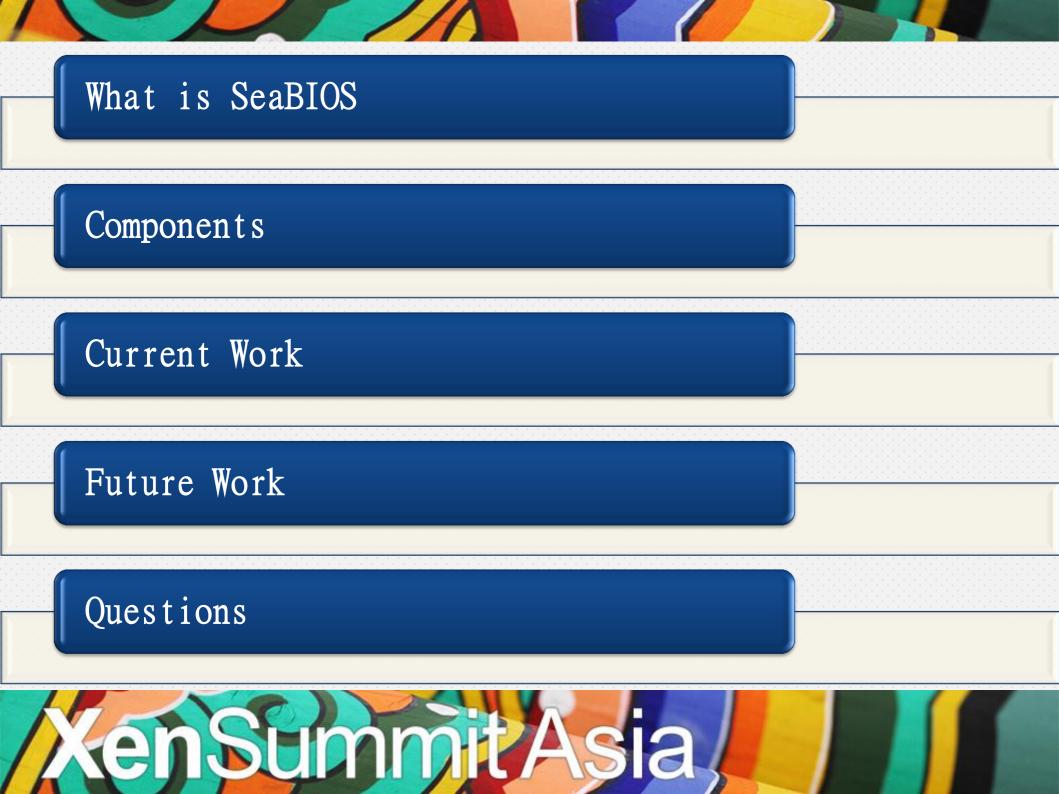
Sponsored by:











#### What is SeaBIOS

- SeaBIOS is an open-source legacy BIOS implementation which can be used in HVMLoader. It implements the standard BIOS calling interfaces that a typical x86 proprietary BIOS implements.
- HVMLoader is a binary blob deployed by the domain builder when a HVM machine is create.
- Currently SeaBIOS is the BIOS implementation used in Upstream Qemu.
- In this presentation we will use this terminology: term "backend" for the thing which implements the PV device, opposite, the PV driver "frontend". For emulated devices we use "device model" to refer to the thing which implements the device (upstream qemu).



#### Components

#### Seabios

- 32bit initialization code
- 16bit interrupt handling

#### Xen Headers

- Arch-x86/xen.h
- Version.h
- Xen.h
- Mvm\_op.h
- hvm/Params.h
- Event-channel.h

- Sced.h
- Xs\_wire.h
- Memory.h
- Blkif.h
- Grant\_table.h
- Ring.h



#### Components

#### Code

- Hypercall code
- Xenbus client (borrowed from hvmloader)
- Shared info page
- Grant table
- Rings for each block device
- SeaBIOS macros

#### SeaBIOS Help Macros

- Each low function modifies the segment pointers, if not specified SeaBIOS uses Code Segment pointer, if size of segment exceeds 256 kbytes, the segment grows.
  - Malloc\_low: allocate in CS Segment
- memalign\_high allocate in last segment, maybe above 1MB barrier.
- SET\_GLOBAL GET\_GLOBAL so global variables are usable from 16Bit
- SET/GET\_SEG GET\_PTRFLAT, to specify from which segment create the pointer, the later turns a pointer from segment:offset to full pointer —in 32bit
- Container\_of, return the container struct of drive\_s: i.e. xendrive\_s

Power on Selft Test (POST)

PCI\_DEVICE\_ID\_XEN\_BLK

Do\_once
Xenbus\_setup
Grant table
Shared info page

For each VBD
Struct xendrive\_s
drive\_s
blkinfo
Share VDB with Xenstore
via xen-wire
Once done register drive\_s
With add\_boot\_drive

```
Struct xendrive_s {
    struct drive_s drive;
    struct blkfront_info info:
Struct blkfront_info {
    int ring_ref;
    struct blkif_front_ring * private;
    struct blkif_sring * shared;
    void * buffer;
    int buf_gref;
Struct drive_s {
    u8 type;
    u64 sectors;
    u32 cntl_ids;
    ul6 blksize;
    ... others ...
```



#### On int13 (16bit Code)

```
(roughly) the following path:
romlayout.S:entry_13()
    disk.c:handle_13()
    disk.c:disk_13()
    disk.c:disk_1302()
        block.c:send_disk_op()
        block.c:__send_disk_op()
        block.c:process_op()
            Xen-blk-op.c:xen process op.
```

```
Struct disk_op_s {
    u64 lba; //sector
    void * buf_fl; //buffer
    struct drive_s *drive_g; //drive
    u16 count //sector count
    u8 command;
}
```



#### Xen\_process\_op

#### **READ**

Request Specific Sector (lba)
Build RING\_GET\_REQUEST -> PUSH\_REQUEST
RING\_GET\_RESPONSE -> OK
memcpy from private buffer to buf\_fl

#### WRITE:

memcpy from buf\_fl to private
Build RING\_GET\_REQUEST -> PUSH\_REQUEST
RING\_GET\_RESPONSE -> OK
clean private buffer.

Limitation: Per 8 sectors on a single write, read no more that buffer size.

Max Buffer size of 4096 bytes



#### Future Work

- Xen\_process\_op can avoid using the private buffer, but has to manage (many more) grefs in the shared buffer instead
- No shutdown sequence (free memory, channels, rings... etc.)
- Net PV Drivers Possibly PXE Support
- Benchmarks on performance

#### A patch example

These are used as part of the Xen hypercall interfaces.

```
Signed-off-by: Daniel Castro <evil.dani@gmail.com>
1 files changed, 58 insertions(+), 3 deletions(-)
diff --git a/src/xen.h b/src/xen.h
index 0ed1e9f..f65078a 100644
--- a/src/xen.h
+++ b/src/xen.h
@@ -17,6 +17,8 @@ static inline int usingXen(void) {
unsigned long xen_hypercall_page;
+typedef unsigned long xen_ulong_t;
+typedef unsigned long xen_pfn_t;
```



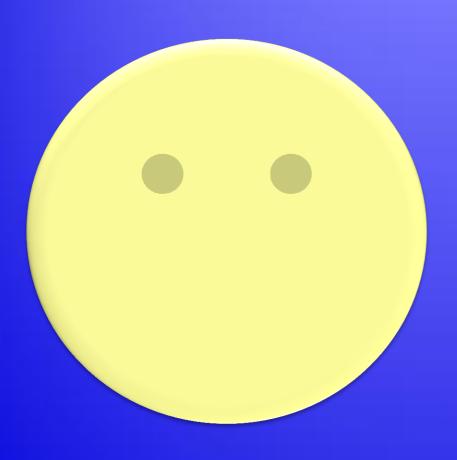
## Questions



### Special Thanks

- This work was sponsored by Google Summer of code
  - ◆Ian Campbell





# THANK YOU!