# XvMotion Internals

*Arun*

*March 11, 2013*

**Confidential**

# Agenda

## XvMotion buffer management

- Migrate module heap overview
- Per Migration heap
- XvMotion buffer  - Bitmaps, bloomFilter, disk queues and block buffers

## XvMotion storage stream

- Storage stream generator
- Storage stream handler

## XvMotion IO

- Source  -  clone and mirror block allocation and read
- Destination - handle conflicting IO/Async IO write

# XvMotion Buffer Management

Overview migrate module heaps

# Migrate Module heap

- Dynamic heap
- Create at the module load time
- Small
- Mainly for meta data allocation
  - Per migration slice and XvMotion slice meta data
- Different from per migration default heap!

```
init_module
    Migrate_Init
        MigrateModuleHeapInit
            Heap_DetermineMaxSize (Migration Slices and XvMotion Slices)
            Heap_Create
```

# Per Migration Default Heap

- Static Heap
- Default heap is create at module load time
- If default heap is not available then heap is create during InitMigration time

```
init_module
    Migrate_Init
        PagePool_Create
            MigrateVASpaceInit
                VASpace_Reserve (12 * 768 MB)
            MigrateHeapCreateDefault
                MigrateHeapDetermineDefaultSize
                MigrateHeapCreate
```

# Per Migration Heap Creation

- If default heap is not available then create a heap

```
Migrate_InitMigration
    MigrateInfo_Alloc
        Migrate_HeapCreate
            MigrateVASpaceCreateHeap
                MigrateVASpaceReserveSlice
                    For numpages
                        Migrate_NiceAlloc
                            PagePool_Alloc
                        VASpace_Map
                    Heap_CreateStatic
```

# XvMotion Buffer Layout

# XvMotion buffer layout

```
/*
 * Each VASpace slice is laid out in the following structure:
 *
 * +----------------------------+ 0 + vpnoffset
 * |        Page Bitmap         | 1 to 2 Pages
 * +----------------------------+
 * |        Block Bitmap        | 4 to 16 Pages
 * +----------------------------+
 * |   Per-device Bloom Filter| 2 pages per device, 276 pages
 * +----------------------------+
 * |        Queue               | 128 to 512 Pages
 * +----------------------------+
 * |         Mapped             |
 * ...
 * |         Unmapped           |
 * +----------------------------+ XVM_MAX_BUFFER_SIZE * 512 + vpnoffset
```

# XvMotion Buffer

- Per Migration buffer size of 64 MB
- Max of 8 = 512 MB

- Page bit map
  - Tracks which pages have been allocated
  - 64 MB/4K = 16k bits = 1 page = 4KB = 16 kbits

- Block bitmaps
  - Tracks which blocks have been allocated
  - A block is 512 byte
  - 8 blocks per page
  - 64 MB = 16k pages * 8 = 128k blocks
  - 4 pages are required to track all blocks

# XvMotion Buffer

- Bloom filter bitmap
  - One bloom filter per disk
  - We support 4 simultaneous disk copy
  - 2 pages for bloom filter

- Disk queue
  - 4 disk queues based on 4 simultaneous disk copy
  - A queue element is 16 bytes.
  - To have a queue of 128 k blocks we need 512 pages

- Buffer overhead
  - 1 + 4 + 2 + 512 = 521 pages
  - 521 * 8 = 4168 (~4k/128k) blocks lost in overhead!

# XvMotion Buffer – Module init

- No heap management code
  - Neither static or nor dynamic heap
- We do our own buffer management
- Very efficient allocation!

```
init_module
    XVMotion_Module
           XVMotion_SetupVASpace
                  Migrate_ModuleAlloc (Slices)
                  VASpace_Reserve ( 8 * 64 MB)
                  Initialize the first slice i.e default
                   XvMotionBitmapInit (page, block and bloomfilter)
                   XvMotionQueueInit
```

# XvMotion Buffer – Migration Init

- Reserve slice at module init time
  - We only allocate bitmaps and queues
  - Block pages are NOT allocated!
  - No guarantee that a XvMotion can succeed in low memory conditions

```
VMotion_PreCopyStart
    VMotionPreCopyStart
        XVMotion_SetupMigration
            XVMotionAllocPool
                XVMotionAllocSliceBitmap
                    For (numPages = metadata of 521 pages)
                        Migrate_NiceAllocPage
                         VASpace_Map
```

# XVMotion Grow Buffer

- Block Pages
  - Even at Migration initialization time we don’t allocate block pages
  - Allocated on demand

```
MigrateBridge_XVMAllocBlocks
    XVMotion_AllocBlocks
        XVMotionCheckForBlockMemLocked
            XVMotionGrowBuffer
                for (slice->pageBitmap.len)
                    Migrate_NiceAllocPage
                        PagePool_Alloc
                    VASpace_Map
                    If (freeblocks > reqblocks)
                       break;
```

# Block Allocation

- Source – Mirror IO

  SVMAsyncIORemoteInt

  MigrateBridge_XVMAllocBlocks

  XVMotion_AllocBlocks

- Source - Clone - SVMAsyncIORead

  SVMAsyncIORead

  MigrateBridge_XVMAllocBlocks

  XVMotion_AllocBlocks

- Destination - Stream

  XVMotion_ReadPrepareBlocks

  XVMotionCheckForBlockMemLocked

# XvMotion Storage Stream

# Storage Stream

- Stream is divided into generator and handler
  - Generator – Sender side
  - Handler – Receiver side

- Sender Side
  - Generator - XVMotion_GetStorageBlocks
  - Payload write - XVMotion_WriteBlocks

- Receiver Side
  - Handler - XVMotion_HandleBlocks
  - Read prepare - XVMotion_ReadPrepareBlocks
  - Read complete - XVMotion_ReadCompleteBlocks

# Storage generator

- Generation - XVMotion_GetStorageBlocks
  - Prepare XVMotionBlockGroupData
    - Similar to VMotionPageGroupPublic
    - Same 128 block (vs Pages)
  - Round robin queue drain
  - Select the queue
  - Pick min blocks to transmit
    - MIN (readyForDrain, queuedForDraining)
  - Dequeue blocks from queue
  - Add address of the block to private data -> blockPtr
  - Maintain count of number of blocks

# Storage write

- Payload write - XVMotion_WriteBlocks
- Transfer memory contents from XvMotion buffer to stream buffer

  - For (publicData.numblocks)
    - Copy the block data to completion data
    - Note: Completion buffers are allocated at channel init time
    - So no stream allocation

# Handle a storage stream

- Handler  - XVMotion_HandleBlocks
- Prepare segments from public data
  - For (public data => numblocks)
    - Determine total blocks
    - Determine total length of blocks
  - Populate the segment with total len
  - Completion world will pick the segment and process

# Read storage stream

- Read prepare - XVMotion_ReadPrepareBlocks
  - Prepare a list of blocks to read from storage stream to XvMotion buffer
  - Get number of blocks from stream's public data num blocks.
  - Allocate num storage blocks from XvMotion buffer
    - XVMotionCheckForBlockMemLocked
      - XVMotionGrowBuffer (if required)
      - XVMotionAllocBlock

- Read complete - XVMotion_ReadCompleteBlocks
  - Read data from storage stream to XvMotion buffers prepared
  - For (pubGroup => numblocks)
    - Initialize a SGA
    - Populate SGA with pubGroup block data
    - XVMotionAddDiskIO

# XvMotion IO

# XvMotion IO

- Destination
- From storage stream add a new IO disk queue

  - XVMotionAddDiskIO
    - Allocate ioentry and make it point to SGA
    - XVMotionWaitForIOCount
      - dq->ioCount < XVM_MAX_DST_OIO_COUNT
    - XVMotionHandleConflictingIO
      - Check disk queue active IO
      - Is new IO to the same region as existing IO
      - If so enqueue and return
    - XVMotionAsyncIOWrite
      - Async_PushCallbackFrameSafe
      - FSS_AsyncFileIO

# Thank You