# XvMotion overview

# Migrate_ToStart

```
MigrateVigorStart
    Migrate_ToStart
        MigrateRPC_MigrateBegin
        MigratePlatformInitMigration
            vmxParams.hasDiskPreCopy = SVMotion_IsSVMotion()
            VMKernel_InitMigration(&tmpSpec, &vmxParams)
                MigrateTypeBridge_InitMigration
                    migrateTypeBridge[mi->type].fns._name(...);
// MIGRATE_TYPE_VMOTION for vmotion and TYPE_FSR for fsr
```

# VMotion_InitMigration

```
VMotion_InitMigration // called for vmotion, xvmotion
    vi = Migrate_Alloc
    VMotionInitStream
    MigrateNet_SetNetStackInstance
    VMotionCreateHelperWorlds
    VMotionStartHelperWorlds
    if (VMotion_IsSource(vi))
        VMotionQueue_Add(vi, VMotionSend_ResolveFeatures)
            ...
            VMotionQueue_Add(vi, VMotion_ReportUserRPCState)
                VMotion_PostVmxMsg(vi, MIGRATE_VMKMSG_ENABLE_USER_RPCS)
        VMotionQueue_Add(vi, VMotionSend_ResolveSwapType)
        VMotionQueue_Add(vi, VMotionSend_ResolveRtt)
```

# FSR_InitMigration

```
FSR_InitMigration // called for svmotion
        FSRAttachToRemoteFSR
                // Connects the two FSRInfos together.  Checks that the memory
                // sizes and swap types match.
                FSRPostVmxMsg(fsr, MIGRATE_VMKMSG_ENABLE_USER_RPCS)
                FSRPostVmxMsg(remoteFSR, MIGRATE_VMKMSG_ENABLE_USER_RPCS)
        // start swap copy
```

# Migrate_RPCsReady

really the start of everything

```
Migrate_RPCsReady // MIGRATE_EVENT_RPC_READY event
    MigrateInitiateCopy(&msgList) // no multiwriter
        MigratePlatformToStart
            // memory copy start for vmotion
            // setup storage streams for xvmotion
            // no-op for svmotion
        SVMotion_Start
            // copy for xvmotion and svmotion
            // no-op for vmotion
```

# MigratePlatformToStart

```
MigratePlatformToStart
    hasMemPreCopy = !(migrationState.spec.type == MIGRATE_TYPE_FSR);
    hasDiskPreCopy = SVMotion_IsSVMotion();
    VMKernel_MigratePreCopyStart(hasMemPreCopy, hasDiskPreCopy)
```

```
FSR_PreCopyStart
    FSRTransferDVSState
    FSRTransferDVSOOBRuntimeState
```

# MigratePlatformToStart

```
MigratePlatformToStart
    hasMemPreCopy = !(migrationState.spec.type == MIGRATE_TYPE_FSR);
    hasDiskPreCopy = SVMotion_IsSVMotion();
    VMKernel_MigratePreCopyStart(hasMemPreCopy, hasDiskPreCopy)
```

```
VMotion_PreCopyStart
    if (!hasDiskPreCopy)
        VMotionQueue_Add(vi, VMotionSend_PreCopyStart)
            VMotionSend_RemoteCall(vi, VMOTION_MSG_PRECOPY_START)
            VMotionStream_StartAsync(VMOTION_STREAM_PRECOPY)
            Action_Post(vmmLeader, MONACTION_MIG_PRECOPY_START); // monitor
    else
        XVMotion_SetupMigration
            xvm = (XVMotion *)VMotion_Alloc()
        VMotionQueue_Add(vi, XVMotion_DiskPreCopyStart)
            VMotionStream_StartAsync(vi, VMOTION_STREAM_STORAGE)
```

# Migrate_RPCsReady

really the start of everything

```
Migrate_RPCsReady // MIGRATE_EVENT_RPC_READY event
    MigrateInitiateCopy(&msgList) // no multiwriter
        MigratePlatformToStart
            // memory copy start for vmotion
            // setup storage streams for xvmotion
            // no-op for svmotion
        SVMotion_Start
            // copy for xvmotion and svmotion
            // no-op for vmotion
```

# SVMotion_Start

```
SVMotion_Start
        if SVMotion_IsSVMotion()
                SVMotion_IterateDisks(SVMotionDiskSetup)
                if (SVMotionIsXVMotion())
                migrationState.isXVMotion = TRUE;
                SVMotionQueueWorkerGroup
                        Worker_Queue(SVMotionMirroredModeThread,
                                        SVMotionMirroredModeThreadDone);
```

# SVMotionMirroredModeThread

```
SVMotionCreateDestDisks()
        SVMotionDiskCreate
                if (disk->isRemote)
                        SVMotionRemoteDiskCreate
                                SVMotionSyncRPC(MIGRPC_OP_SVMotionDiskCreate)
                                // on dest
                                SVMotion_DiskCreateRPC
                                        DiskLib_Create
                else
                        SVMotionLocalDiskCreate
                                DiskLib_Create
SVMotionLoadDestDisks())
SVMotionStun() // install mirrors
SVMotionThreadWaitForStun()
VmkEvent_RegisterHandler(VMKEVENT_GET_DISKCOPY_BITMAP,
                                SVMotion_SendDiskCopyBitmap)
SVMotionMirroredModeThreadDiskCopy
```

# SVMotionMirroredModeThreadDiskCopy

```
SVMotionMirroredModeThreadDiskCopy
        SVMotionInitDiskCopySchedulerQueue
            schedulerQueue[(*iter)++].disk = disk;
        SVMotionAsyncDiskCopySort
            schedulerQueue[i].weight to: 0=in-progress, -1=completed, n=to-go
        // 4 at a time, and if weight>=0
            SVMotionAsyncCopyInit
                SVMotionMirroredModeAsyncCopyIoctl
                    Mirror_SendAsyncCopyIoctl // SVM_ASYNC_DISK_COPY_START
```

# SVMAsyncDiskCopyStart

kernel land

```
SVMAsyncDiskCopyStart
    // starts a SVMAsyncDiskCopyThread
    SVMAsyncDiskCopyThread
        for every 64MB
            copyArgs->dm.fsdm.dmExtent[0].length = 64MB
            copyArgs->dm.fsdm.dmExtent[0].srcFileOffset = progress->currOffset;
            copyArgs->dm.fsdm.dmExtent[0].destFileOffset = progress->currOffset;
            SVMFDSIoctlMoveData // not really an Ioctl anymore
                SVMDMCopyDiskUsingBitmap // XXX leave for later
                    SVMMoveData
                SVMMoveData
                    if local and DM
                        FSS_IoctlByFH(IOCTLCMD_VMFS_VMKMOVEDATA) // DM
                    if xvmotion
                        SVMReadDataToXVMBuffer
```

# SVMReadDataToXVMBuffer

API for xvmotion transfer of 64MB

```
SVMReadDataToXVMBuffer
    SVMXVMProcessVMFSBlockData // vmfs optimizations
    for all dm->numExtents // usually 1
        SVMAsyncIORead // beware, xvmotion specific
            MigrateBridge_XVMAllocBlocks
            SVMAsyncIOIssueRead
                FSS_AsyncFileIO(comp=SVMAsyncIOReadDone)
                                // on completion
                                SVMAsyncIOReadDone
                                    MigrateBridge_XVMEnqueue
            SVMWaitForReadIOCompletion
```

# Summary of [s/x]vmotion disk copy

- vmx sends an start copy ioctl to the kernel
- kernel starts a thread that copies 64MB chunks at a time
- for each chunk, call the DM or the xvmotion DM

- mirroring? next

# Mirroring setup

```
init_module
        SVMRegister
                FDS_RegisterDriverWithAttributes(SVM_DRIVER_NAME, &svmOps, driverID)
                svmDriverID = driverID;
```

```
static FDS_DeviceOps svmOps = {
  SVM_AsyncIO,
  SVM_Ioctl,
  SVM_MakeDev,
  SVM_RemoveDev,
};
```

```
        SVMotionDiskCreateMirrorNode
                VMKernel_FDSMakeDev
                        SVM_MakeDev
                                DevLib_CreateDevice(devLibObj, svmDriverID)
```

# SVM_AsyncIO

```
SVM_AsyncIO(FDS_HandleID)

    if (ioFlags & FS_READ_OP)
        return FSS_AsyncFileIO(driverData->fhids[0]) // to the source only

    if (driverData->mode == SVMMIRROR_MODE_LOCAL)
        SVMAsyncIOLocalInt
    else
        SVMAsyncIORemoteInt
```

# SVMAsyncIOLocalInt

SVMAsyncIOLocalInt

```
        if ((endOffset <= driverData->dmCopyStartOffset)    // we already copied it

            Async_StartSplitIO // complete when everybody completes
            for (index = 0; index < ARRAYSIZE(driverData->fhids); index++) {
                FSS_AsyncFileIO(driverData->fhids[index])

        else // source only
            FSS_AsyncFileIO(driverData->fhids[0]) // 0 is source
```

# SVMAsyncIORemoteInt

SVMAsyncIORemoteInt

```
        if ((endOffset <= driverData->dmCopyStartOffset) // we already copied it

            MigrateBridge_XVMAllocBlocks
            MigrateBridge_XVMEnqueue // we already have the data to write
            FSS_AsyncFileIO(driverData->fhids[0]) // 0 is source

        else // source only
            FSS_AsyncFileIO(driverData->fhids[0]) // 0 is source
```

# XVMotion buffer

```
MigrateBridge_XVMAllocBlocks
        XVMotion_AllocBlocks
                XVMotionCheckForBlockMemLocked // wait
                XVMotionAllocSgaBlocksLocked // alloc


MigrateBridge_XVMEnqueue
        XVMotion_EnqueueBlocks
                for (i = 0; i < sga->length; i++)
                        XVMotionQueuePush(xvm->slice, dq, blk, offset, len)
```

# SVMotionMirroredModeThreadDiskCopy

```
SVMotionMirroredModeThreadDiskCopy
        // 4 at a time, and if weight>=0
            SVMotionAsyncCopyInit
                SVMotionMirroredModeAsyncCopyIoctl
                    Mirror_SendAsyncCopyIoctl // SVM_ASYNC_DISK_COPY_START
        // wait for all copies to finish

        SVMotionVMKernelPreCopyIterDone
            VMKernel_MigrateDiskPreCopyIterDone
                VMotion_DiskPreCopyIterDone
            VMotionQueue_Add(vi, VMotionSend_PreCopyStart)
                // start memory precopy
```