Revision History

Version	Date	Author	Description
0.1	04/24/2012	Hill Zhao	Initial draft.

Acronym and Codename Table

1 Foreword

2 What Should Learn?

21 Know Products

- High level overview of products
- Workstation using menu, files location and what do they do(exec, driver, virtual machine logs and files), VM configuration, how to run debug mode

Reference:

[Start Hire]

https://wiki.eng.vmware.com/NewbieRoadmap

[BootCamp]

https://wiki.eng.vmware.com/EngBootCamp

22 Builds

- Setup ESX PC, PXE boot
- Install Workstation to be familiar with it
- Perforce, P4V
- Build private image

Reference:

[setup build of ESX]

https://wiki.eng.vmware.com/NewbieRoadmap#SetupBuildEnvironment

[store many vms]

https://wiki.eng.vmware.com/QAVMLibrary

[build tools]

https://wiki.eng.vmware.com/ToolsBuilding

23 Fix bugs

- Code base, source file, right developer

Reference:

[Useful paper and docs]

http://rd.eng.vmware.com/web/tech-talks/

2.4 White papers

- Virtual Machine Monitors: Current Technology and Future Trends
- VMware architecture
- Virtualizing IO

Reference:

[VMM: current and Future]

http://www.computer.org/portal/web/csdl/abs/html/mags/co/2005/05/r5039.htm

3 Build & Run

3.1 Perforce

3.2 Scons

- Like Make, scons build file is written by python
- Clean all the build
  - scons clobber
- get more debug info
  - scons LOGGING=aliases=info
  - scons LOGGING=aliases=info 2>&1 | tee rawtargets.txt
  - cat rawtargets.txt | awk ' /alias/ {print $5} ' | tr -d \' | sort | uniq > scons-targets.txt
- in Local.sc set "VERBOSE=1" to get more debug info
- If can not use "scons esx-all" then
  - dbc /build/apps/bin/scons esx-all

Reference:

https://wiki.eng.vmware.com/Scons

https://wiki.eng.vmware.com/SConsManual/User

3.3 Sandbox build

- gobuild

```
gobuild sandbox queue server --changeset=1854313 --branch=esx35ep3
gobuild sandbox queue server --changeset=1854315 --branch=esx40ep8
gobuild sandbox queue server --changeset=1854311 --branch=esx50ep4
gobuild sandbox queue server --changeset=1854107 --branch=esx41ep3
gobuild sandbox queue server --changeset 1854313 --branch esx35ep3 --buildtype release --accept-defaults
```

Reference:

https://wiki.eng.vmware.com/Beijing/CPDBJ/Members/Liyan/CIM

http://patchtool.eng.vmware.com/gss/hotpatch/index/ongoing/

3.4 ESXi

- PXE BOOT
  - Make sure your build host satisfies the requirements to perform a build.

- Create a Perforce client for source codes based on product, target, branch and changeset you'd like to build.
- Optionally, set up appropriate symlinks pointing to pre-synced source tree (under Linux only), if you'd like to avoiding syncing everything from Perforce.
- Optionally, under bora directory, set up Local.mk for targets built with make, or Local.sc for targets built with scons, for frequently used command line flags.
- Type "make", "scons", "maven", or whatever other command to build the desired targets.
- Find the deliverables you need.

- Go to https://buildweb.eng.vmware.com/dbc/ request one space, pek-dbc101.eng.vmware.com
- Go to http://p4user.eng.vmware.com/ to create your p4 user.
- Build ESXi code
  - ssh pek2-dbc101.eng.vmware.com
  - cp ~hillzhao/.bashrc ./ to prepare the workspace
  - p4 login, then p5 client to create one client or “p5 client esx50u1” to add more dir for this client
    - or p4 client need create .p4config to add “P4CLIENT=hillzhao-dbcbj-esx41u3” and “P4PORT=build-p4proxy.eng.vmware.com:1666”
  - p4 sync to checkout code
  - go to ./bora create Local.sc and add “ PRODUCT="esx" and “ BUILDTYPE="obj", if want to pxeboot need add ESX_PXE_PROVISION_DIR="/dbc/pek2-dbc101/hillzhao/pxe" and ESX_PXE_HTTP_ROOT=http://pek2-dbc101.eng.vmware.com/hillzhao/pxe
  - if want to less config, copy the esxconf.sc to bora
  - then go bora dir do “scons esx-all visor-pxe” to build code and pxe. For 50u2, need add “ESX_PXE_AUTOPARTITION="False"” in Local.sc .
  - Boot PXE, then get the MAC address. And enable intel-vt
  - Use the right mac addressThen ssh pxeuser@suite(ca$hc0w), then “/PXEconfig.pl -m D4:AE:52:64:10:82 -d dbc/pek2-dbc101 -p hillzhao/pxe -l China-Raycom” to set the pxe boot root dir.
  - Open vsphere vi client, and configure ESX, and storage, and nfs mount
  - If want to boot from official build, refer to PXEBOOT
- Nested boot ESX
  - Create vm with set the os type to "esxi5”, also set the vt-x support for vm setting.
  - Ssh esx (ssh root@10.117.7.240) to enable vhv for host echo 'vhv.allow = "TRUE"' >> /etc/vmware/config and allow nested VM run. (run in outer guest) echo 'vmx.allowNested = "TRUE"' >> /etc/vmware/config (esx41u3 vm’s /etc/vmware/config)
  - Then pxe boot prepare
    - hillzhao@pek2-dbc101:/dbc/pek2-dbc101/hillzhao$ export MAC="00:50:56:ac:55:11"
    - hillzhao@pek2-dbc101:/dbc/pek2-dbc101/hillzhao$ export PXE_LOCATIONS="China-Raycom"

        - hillzhao@pek2-dbc101:/dbc/pek2-dbc101/hillzhao$ ssh pxeuser@suite ./PXEconfig.pl -m $MAC -d dbc/pek2-dbc101 -p hillzhao/pxe -l $PXE_LOCATIONS
    - and enable restrict_backdoor add "monitor_control.restrict_backdoor = TRUE" to vmx config file of outer vmx. (/vmfs/volumes/ LocalStore/ esx41u3/esx41u3.vmx) (this need when the vm is created)
    - Then power on VM(create pek-exit15 nfs, memory, cpu, and then install ubuntu10.4)
    - Issue1: vm can not support >3g memory.
        - Rootcause: swap also need the same space as memory, and the disk space crated with only 16G and stat file is over 15G, then there is no space (should check /var/log/message. Or vmware.log)
        - Solution: create one new big datastore, and migrate it from original store.
    - Issue2: dhcp wrong address with 192.xxx not 10.xxx
        - Rootcause: other start dhcpd
        - Solution: replug the network line. (use esxcfg-vmknic -l, and dhclient-uw)
    - Issue3: apt-get update can not work
        - Rootcause: need set proxy in /etc/apt/apt.conf
    - Issue4: can not web browse
        - Rootcause: also need set proxy.
    - Issue5: ubunt10.04 can not reproduce ths issue.
        - solution: install 10.10
    - Issue6: after reboot can not see vm
        - Solution: vim-cmd solo/registervm or remove vm and add inventory in the vi client.
    - Issue7: can not pxe boot for not find the directory
        - Rootcuase: Suite set should not absolute path. should reset the mac and dir, and also make softlink to esx50 and esx40
    - Issue8: rm file can not p4 sync
        - Solution: p4 revert first, then sync.
    - Issue8: hot-add hard disk space to guest os,can not work
        - Solution: should add one new hard disk.
- You can access the support console by pressing Alt-F1. To come back to the Configuration screen, press Alt-F2

- ESX reboot
    - If esx5 is override by esx4.1 need build again.
    -
- Making VMFS and Core Partitions

- Create a core partition of size 115 MB using fdisk /dev/disks/vmhba1:0:0:0 (or whatever the proper device is for the disk you want to partition).
- Change the type of thie partition to fc, which is ESX dumps.
- Create another partition for a VMFS volume.
- Change the type of this partition to fb, which is VMFS.
- To create the VMFS volume, run vmkfstools -C vmfs3 /dev/disks/<diskdevice>.

- To burn a K/L install CD from an obj build (adjusting the device name as appropriate for your workstation) use:
- cd $BORA/build/scons/package/devel/linux32/obj/esx # when using an obj tree
- cdrecord dev=/dev/cdrw esx-DVD-*.iso
- ~ # esxcfg-dumppart -l
- ~ # cd /dev/disks
- /dev/disks # ls
  - vml.01000000003f393939393939393394c4420302052
  - vml.01000000003f393939393939393394c4420302052:1
  - vml.01000000003f393939393939393394c4420312052
  - vml.01000000003f393939393939393394c4420312052:1
  - vml.01000000003f393939393939393394c4420312052:2
- /dev/disks # vmkfstools -C vmfs3 -S storage2 vml.01000000003f393939393939393394c4420312052:2
  - Creating vmfs3 file system on "vml.01000000003f393939393939393394c4420312052:2" with blockSize 1048576 and volume label "storage2".
  - Successfully created new volume: 47be6b96-e048d70c-229a-0030485cd377
  - /dev/disks #

Reference:

https://wiki.eng.vmware.com/Build/HowToBuildProducts#VMware_ESX_Server_Version_5.x

http://en.wikipedia.org/wiki/Preboot_Execution_Environment

https://wiki.eng.vmware.com/PXEBoot [pxeboot]

https://wiki.eng.vmware.com/MoveToMain [how to build and boot pxe]

https://wiki.eng.vmware.com/ManualVmfsCoreSetup[Making VMFS and Core Partitions]

3.5 ViClient

- Create virtual machine

- Create data storage and nfs(pek-exit15, showmount –e pek-exit15)
- Edit vm and use iso file in nfs(pek-exit15)
- Power on with connected with ISO

3.6 CrossPort

- P5 crossport cs# esx50 esx41u3(get change num from the old bug)
- P4 opened can see the new change number
- P4 resolve
- Post-review (http://pa-dbc1007.eng.vmware.com/hfu/bin/t1-post-review.sh), then get review info at https://reviewboard.eng.vmware.com/r/331308/diff/#index_header
- If need change the diff only, use (http://pa-dbc1007.eng.vmware.com/hfu/bin/t1-post-review-diff-only.sh)
- Patchtool to create qa template.(http://patchtool.eng.vmware.com/qatemplate/index/index/detail?bug_id=867972&action=submit)
- P4 change cs# to fill the info for change.
- Change CheckinApproveRequest, and then wait to approve.
- P4 submit –c cs# (if failed for need more info, should p4 change cs# first)

3.7 Fix bug

- First all file ready only
- P4 edit to make write, and then p4 opened, can see them
- P4 change will get new change num
- Build them, do test and then do test-esx on host
- Then do post review, add test info on the webpage, and wait ship it
- Then do QE template, wait approve.
- P4 change cs# to fill the info for change.
- P4 submit –c change#

3.8 Test esx

- Mount nfs
  - esxcfg-nas -a -o pek2-dbc101.eng.vmware.com -s /dbc/pek2-dbc101/hillzhao/esx41u3 esx41u3
  - export VMTREE=/vmfs/volumes/ esx41u3/bora
  - export BLDDIR=/vmfs/volumes/ esx41u3/bora/build
  - export VMBLD=obj
- mount /build/toolchain
  - esxcfg-nas -a -o build-toolchain.eng.vmware.com -s /toolchain toolchain
  - mkdir /build
  - ln -s /vmfs/volumes/toolchain /build/toolchain
- set up perl symlinks
  - mkdir -p /usr/local/lib/perl5
  - ln -s /build/toolchain/lin32/perl-5.8.8/bin/perl /usr/bin/

- ln -s /build/toolchain/lin32/perl-5.8.8/lib /usr/local/lib/perl5/5.8.8

- setup uwpython symlinks(only for version 3.5 and lower)
  - ln -s $BORA_ROOT/uwpython-2.5/bin/python /usr/bin/
  - ln -s $BORA_ROOT/uwpython-2.5/lib/python2.5/ /lib/
- if you are using a non-login shell (e.g. ssh with a command) then source /etc/profile. This will make sure PYTHONHOME and PYTHONPATH have the libraries that we package and ship. /etc/profile is read automatically if you are using a login shell (e.g. ssh with no command)
- run $VMTREE/support/scripts/test-esx out of your tree.
- $VMTREE/support/scripts/test-esx -n 'cim/.*'

Reference:

https://wiki.eng.vmware.com/TestEsx

3.9 Build ESX3.5 (Make not Scons)

- View:

```
        //depot/bora/esx35/...          //haol-pa-lin-bld386-esx35/bora/...
        //depot/bora-floppy/esx35/...   //haol-pa-lin-bld386-esx35/bora-
floppy/...
        //depot/bora-root/esx35/...     //haol-pa-lin-bld386-esx35/bora-
root/...
        //depot/env64cc/esx35/...       //haol-pa-lin-bld386-esx35/env64cc/...
        //depot/bora-vmsoft/esx35/...   //haol-pa-lin-bld386-esx35/bora-
vmsoft/...
        //depot/console-os/esx35/...    //haol-pa-lin-bld386-esx35/console-
os/...
        //depot/vmkdrivers/esx35/...    //haol-pa-lin-bld386-
esx35/vmkdrivers/...
        //depot/esxrpms/main/...        //haol-pa-lin-bld386-esx35/esxrpms/...
        //depot/crosscompile/main/...   //haol-pa-lin-bld386-
esx35/crosscompile/...
```

- Symlinks

```
ln -s /build/trees/esx35/bora-root .
ln -s /build/trees/esx35/env64cc .
ln -s /build/trees/main/esxrpms .
ln -s /build/trees/main/crosscompile .
```

- Local.mk

  Now create a text file named Local.mk under directory bora to include all your build options. In the example below I've deliberately turned off all caches so I'll be building everything from source myself.

```
export PRODUCT=server
export OBJDIR=beta
export VERBOSE=3
export NUM_CPU=4
 MAKE += ESX_NO_COS_CACHE=1
```

- Build vmx
  make vmx
- Scons vmx

3.10 Build Tools

- Change local.mk
  - export VERBOSE=3
  - export NUM_CPU=4
  - export OBJDIR=beta
  - #export ARCH=x64
  - export ARCH=x86
  - export PRODUCT=tools-for-linux
  - export CROSSCOMPILE_TOP_DIR=/build/trees/main/crosscompile
    - or //depot/crosscompile/main/...
- Build tool kernel module
  - ~/bin/tl-make-cross.sh bld-2.6.32-279-amd64-RHEL6.3 > ./hill_bld-amd64-rhel63
  - This is actually make drivers for the rhel6.3 target
- Build tool for linux
  - ~/bin/tl-make-bora-vmsoft.sh > ./hill_build_tools_for_linux 2>&1

- Build by gobuild
  - gobuild sandbox queue tools --changeset=1915066 --branch=esx50u2

Reference:

https://wiki.eng.vmware.com/GoBuild/Components/Tools [build tools]

https://wiki.eng.vmware.com/CPD/Hosted/CPDToolsInstallerInfo [install tools]

3.11 Build Tools On Windows

- Net map \\build-toolchain.eng.vmware.com, apps and build
- Build env
  - rem build.env.bat
  - set TCROOT=T:
  - set BUILDAPPS=U:
  -
  - set PATH=%TCROOT%\win32\bin;%BUILDAPPS%\bin;%SystemRoot%\system32;%SystemRoot%;C:\Program Files (x86)\Vim\vim73\gvim.exe
  -
  - set P4CONFIG=p4config
  - set P4EDITOR=C:\Program Files (x86)\Vim\vim73\gvim.exe

- p4config
- p4 sync
- Local.mk
-
  - export VERBOSE=4
  - export NUM_CPU=24
  - #export MAKE_CROSS=0
  - export CROSSCOMPILE_TOP_DIR=/build/trees/crosscompile
  - #export PRODUCT=tools-for-freebsd
  - export PRODUCT=tools-for-windows
  - #export PRODUCT=tools-for-linux
  - #export PRODUCT=tools-for-solaris
  - export ARCH=x86
  - #export ARCH=x64
  - export OBJDIR=obj
- make tool-for-windows-iso
- **Computer science or relative major**
  - Hands-on Experience of Linux Kernel debugging
  - hands-on experience on Linux platform, system configuration, system admin
  - Familiar with Linux Kernel or other UNIX system kernel.
  - C programming, POSIX/UNIX systems programming
  - Script programming, Python preferred
- Good communication (Chinese and/or English).
- **Responsibilities**
  - Develop coredump summary of Linux. That automatically collects debug info for Linux core dump, and analyzes the crash core dump.
-

3.12 *Debug vmx hang (for linux)*

- "tar xzvf vmx-vm-support tarball"
- "./reconstruct.sh"
- Go to vmfs/vm_dir
- "tar xvzf *suspend.txt" (this the vmss tarball) to get the *.vmss
- goto bora, "make vmss2core", to get the "vmss2core" binary.
- Copy the *.vmss to dbc, and use vmss2core to generate the vmss.core0/1 "vmss2core –N6"
- strings vmss.core0 | grep vmlinuxz to get the linux kernel version
- Download the vmlinux.rpm from debuginfo.centos (kernel-debuginfo-2.6.18-8.el5.x86_64.rpm)

- "rpm2cpio kernel-debuginfo-2.6.18-8.el5.x86_64.rpm | cpio –div" to untar the rpm tarball
- "crash --machdep phys_base=0x200000 vmlinux vmss.core0" to get the vmcore debug info

Reference:

http://hfu-dell.eng.vmware.com/bugs/

https://wiki.eng.vmware.com/VmssToCore

http://debuginfo.centos.org/5/x86_64/

SLES:

- Get kernel-default-base-2.6.32.12-0.7.1.x86_64.rpm from SP1 repo

http://build-sles-smt.eng.vmware.com/repo/$RCE/SLES11-SP1-Pool/sle-11-x86_64/rpm/x86_64/

- Get debug info kernel-default-debuginfo-2.6.32.12-0.7.1.x86_64.rpm in sp1-debuginfo repo

http://build-sles-smt.eng.vmware.com/repo/$RCE/SLE11-SP1-Debuginfo-Pool/sle-11-x86_64/rpm/x86_64/

- Rpm2cpio, then gunzip vmlinux-2.6.32.12-0.7-default.gz
- Copy vmlinux-2.6.32.12-0.7-default and vmlinux-2.6.32.12-0.7-default.debug to the vmss.core directory, and run crash.

  ./lib/debug/boot/vmlinux-3.0.80-0.7-default.debug
- Get kernel-default-debugsource-3.0.80-0.7.1.x86_64.rpm for source file

  http://build-sles-smt.eng.vmware.com/repo/$RCE/SLE11-SP2-Debuginfo-Updates/sle-11-x86_64/rpm/x86_64/

3.13 Debug vmx coredump on Linux

- mount -t nfs bugs.eng.vmware.com:/bugs /bugs
- mount -t nfs build-storage60.eng.vmware.com:/storage60 /build/storage60
- mount -t nfs build-toolchain.eng.vmware.com:/toolchain /build/toolchain
- mount -t nfs build-toolchain.eng.vmware.com:/apps /build/apps
- export VMPROD=esx
- export VMBLD=release
- export VMTREE=/build/storage60/release/bora-110268/bora
- root@bo-virtual-machine:/bugs/files/0/0/4/4/7/9/4/1/sr1484570535/vm-support-iadadobdmi03p-2010-02-01--18.15.29702/vmfs/volumes/49f783e8-4f346819-b8da-002219c7e0b4/nO8fsaTTTXNU6iyA9epFNQ# /build/apps/bin/esx/vmkgdb64-7 $VMTREE/build/release/server/vmware-vmx vmware-vmx-core.000

3.14 Debug vmx hang (for vmx userworld)

- Vmware.log , first line get build num
- bld info xxxxx
- ~/bin/debug.env.sh 582267

```
hillzhao@pek2-dbc101:/dbc/pek2-dbc101/hillzhao/bugs/890012/vm-support-
PMURDEVMHP01-2012-06-12--04.32.12736$ ~/bin/debug.env.sh 582267
export VMPROD=esx
export VMBLD=release
export VMTREE=/build/storage26/release/bora-582267/bora


-----------------------------------------------------------------------
vmkdump_extract:
        $VMTREE/build/scons/package/devel/linux32/$VMBLD/
$VMPROD/apps/vmkdump_extract/vmkdump_extract vmware-*zdump.N
gdb from toolchain, e.g.
        /build/toolchain/lin32/gdb-6.8-1/bin/gdb


vmm core:
        $VMTREE/vmcore/support/debug/gdbWrapper.pl --core <vmware-coreN>


vmss2core:
        vm-support -x
        vm-support -Z <wid>
        vmss2core -W <some>.vmss > vmss.out
        ----
        $SRCROOT/apps/scripts/vm-support
        cd $SRCROOT; make vmss2core; $BUILDROOT/$VMBLD/support/debug/vmss2core/vmss2core


vmx core:
    $VMTREE/support/scripts/debug-uw vmx-zdump.000
    $VMTREE/support/scripts/debug-uw
$VMTREE/build/scons/package/devel/linux32/release/esx/vmware-vmx  vmware-vmx-zdump.000


vmkernel core:
        $VMTREE/support/scripts/debug-esx <vmkernel-core.N>
```

- export VMPROD=esx
- export VMBLD=release
- export VMTREE=/build/storage26/release/bora-582267/bora
- cp $VMTREE/build/scons/package/devel/linux32/release/esx/vmware-vmx ./
- ../../../esx41u3/bora/build/scons/package/devel/linux32/obj/esx/apps/vmkdump_extract/vmkdump_extract vmware-vmx-zdump.000
- ../../../esx41u3/bora/support/scripts/debug-uw vmware-vmx vmware-vmx-zdump.000 OR gdb vmware-vmx vmware-vmx-core.000
  - NOTE: if can not get symbol, try export VMKGDB=/build/apps/bin/esx/vmkgdb64-7

3.15 Debug vmm (monitor)

- export VMPROD=esx
- export VMBLD=beta
- export VMTREE=/dbc/pa-dbc1019/hillzhao/sandbox/esx50u2-1839049/bora
- tar xzvf vmmcores.gz

- $VMTREE/vmcore/support/debug/gdbWrapper.pl --core vmmcores
- Note: if nm failed
  - $VMTREE/vmcore/support/debug/gdbWrapper.pl --aslr-delta-script ./vmkernel-aslr-start-delta --core vmware64-core1

3.16 Generate vmm Elf binary

- Change code

bora/vmcore/support/debug/vmmvmkstacksyms.pl

```
    $linkerScript = $buildTree .
                    "/bora/vmcore/support/debug/modular-to-static-
linker.pl";
    GDBWVMM::SetLinkerScript(0, $linkerScript);
}
GDBWVMM::SetLinker($GDBWUTIL::binary, undef);
GDBWVMM::LinkStaticVMM(undef, $vmwareLog, $GDBWUTIL::binary);


print($GDBWVMM::staticVMMFile);
```

- Get the generate log

```
/vmfs/volumes/50df5849-72c0f35b-8bd3-000c29396c43/valgrind_test #
$VMTREE/vmcore/support/debug/vmmvmkstacksyms.pl -l vmware.log
   Linking monitor executable
EXEC: /build/toolchain/lin32/perl-5.10.0/bin/perl /vmfs/volumes/acde89a9-
67ac4879/bora/vmcore/support/debug/modular-to-static-linker.pl --linker
/vmfs/volumes/prod2013-stage-valgrind/bora/build/esx/obj/vmcore-
exported/obj/linker --log vmware.log --phase last --vmm
/tmp/RYqD5XXYmA/vmm64 --search /vmfs/volumes/prod2013-stage-
valgrind/bora/build/esx/obj/vmcore-exported/obj --find-binaries-script
/vmfs/volumes/prod2013-stage-valgrind/bora/support/scripts/find-binaries
```

- cp /build/storage60/release/bora-1157734/bora/vmcore/support/debug/vmmvmkstacksyms.pl ./
- cp /build/storage60/release/bora-1157734/bora/build/storage60/release/bora-1157734/bora/build/esx/release/vmcore-exported/release/linker ./
- $VMTREE/vmcore/support/debug/modular-to-static-linker.pl --linker $VMTREE/build/esx/release/vmcore-exported/release/linker --log vmware.log --phase last --vmm ./vmm64 --search $VMTREE/build/esx/release/vmcore-exported/release --find-binaries-script $VMTREE/support/scripts/find-binaries
- Generate objdump

```
hillzhao@pek2-dbc202:/dbc/pek2-dbc202/hillzhao/src/prod2013-stage-
valgrind/bora$ readelf -a vmm64 > valgrind_vmm64_readelf
```

3.17 Debug Linux Tools

- Copy unstrapped tools sharedlibrary to local PC
  - /build/storage60/release/bora-1065307/build/linux64/bora-vmsoft/build/release-x64/tools-for-linux/Linux/apps/vmtoolslib/libvmtools.so
  - /build/storage60/release/bora-1065307/build/linux64/bora-vmsoft/build/release-x64/tools-for-linux//Linux/services/vmtoolsd/libvmtoolsd.so

- Add symbols to gdb
- gdb -c core.2502 /usr/sbin/vmtoolsd
- (gdb) info sharedlibrary

```
0x00007fb4247512b0  0x00007fb4247b9ee8  Yes (*)     /usr/lib/vmware-
tools/lib/libvmtools.so/libvmtools.so
0x00007fb4245462b0  0x00007fb4245aeee8  Yes         /usr/lib/vmware-
tools/lib/libvmtoolsd.so/libvmtoolsd.so
(gdb) add-symbol-file ./libvmtools.so 0x00007fb4247512b0
add symbol table from file "./libvmtools.so" at
        .text_addr = 0x7fb4247512b0
(y or n) y
Reading symbols from /root/Downloads/libvmtools.so...done.
```

- (gdb) add-symbol-file ./libvmtoolsd.so 0x00007fb42452eb60

```
add symbol table from file "./libvmtoolsd.so" at
        .text_addr = 0x7fb42452eb60
(y or n) y
Reading symbols from /root/Downloads/libvmtoolsd.so...done.
```

- (gdb) bt

```
#0  0x00007fb4247b9d9b in Hostinfo_TouchXen () at /build/mts/release/bora-
1065307/bora/lib/misc/hostinfoHV.c:121
#1  0x00007fb4247b98ad in VmCheckSafe (checkFn=0x7fb4247b9d80
<Hostinfo_TouchXen>) at /build/mts/release/bora-1065307/bora-
vmsoft/lib/vmcheck/vmcheck.c:136
#2  0x00007fb4247b9980 in VmCheck_IsVirtualWorld () at
/build/mts/release/bora-1065307/bora-vmsoft/lib/vmcheck/vmcheck.c:243
#3  0x00007fb42452f19d in ToolsCoreRunCommand (option=0x0, value=0x0,
data=0x1, error=0x39b7232b60)
    at /build/mts/release/bora-1065307/bora-
vmsoft/services/vmtoolsd/cmdLine.c:5
```

3.18 Debug PSOD

- /var/core/ vmkernel-zdump.1
- Vmware.log , first line get build num
- bld info xxxxx
- ~/bin/debug.env.sh xxxxx
- debugzilla.py vmkernel-zdump.1 OR
- export VMPROD=esx
- export VMBLD=release
- export VMTREE=/build/storage26/release/bora-582267/bora
- vmkernel core: $VMTREE/support/scripts/debug-esx <vmkernel-core.N>
- vmkdump_extract_wrapper.py vmkernel-zdump.9 to get zdump log file

3.19 DVfilter config

- Dvfilter-generic
- Dvfilter vmknic bind address

- Dvfilter change *.vmx, this need change when poweroff, not power on.
- Dvfilter-config, vmxnet3 load with parameter, and others
- Host dvfilter 2222 open, must config this before guest os power on
- Guest os adapter connected on

  Note:

  If use external pnic as vswitch, there will vm hang and many packet dropping.

3.20 Swiscsi

- Add iscsi software adapter, Storage adapter, change to swiscsi
- Add the network ip in proiority, 196.10.1.1:3260,196.10.1.2:3260
- Add switch config:
  - esxcfg-vmknic -a -i 195.10.1.114 -n 255.255.255.0 -m 9000 NAS-10G
  - esxcfg-vmknic -a -i 196.10.1.25 -n 255.255.255.0 -m 9000 iSCSI-10G
- rescan the storage

321 Build vmware tools

- cd bora-vmsoft
- cp Local.mk
- make tools-for-linux

322 Kstats

To generate additional virtual machine statistics using the vmx-buildtype tool:

- Run this command and verify that the virtual machine is running on the host:

  vim-cmd vmsvc/getallvms
- Start the workload on the virtual machine from which the statistics must be gathered.
- Run this command:

  /bin/vmx-buildtype --vmname=<vm-displayname> --server localhost --buildType stats --ssr
- Run this command and verify that the virtual machine is now running with the stats VMX/VMM:

  head –n1 /vmfs/volumes/<pathtovm>/vmware.log

  The option should be STATS.
- After the test completes, run this command to stop the log collection:

  /bin/vmx-buildtype --vmname=<vm-displayname> --server localhost --buildType release --ssr
- Run this command and verify that the virtual machine is now running with the release VMX/VMM:

  head –n1 /vmfs/volumes/<pathtovm>/vmware.log

  The option should now be set to Release.

- Example:
  - /bin/vmx-buildtype --vmname=rhel5-dvfilter-app --server localhost --buildType stats --ssr
  - /bin/vmx-buildtype --vmname=rhel5-dvfilter-app --server localhost --buildType release --ssr
  - head –n1 vwmare.log
  - ../../sandbox/esx50u2-1839049/bora/support/scripts/kstats.prl -f 19 -l 21 vmware-stats.log > kstats.log

Reference:

https://wiki.eng.vmware.com/Performance/Automation/kstats

http://knova-prod-kcc-vip.vmware.com:8080/contactcenter/php/search.do?cmd=displayKC&docType=kc&externalId=1030549&sliceId=1&docTypeID=DT_KB_1_1&dialogID=129312708&stateId=1%200%20129286819

3.23 VMsample

- monitor_control.log_vmsample = TRUE
- Enable them at power-on time with "monitor_control.enable_vmsample = 1" in .vmx, or at run-time with `vmdumper -l`; `vmdumper <wid> samples_on`. They will appear in vmware.log for each VCPU.

3.24 VMMCallstack

- Goto ./stats directory
- $VMTREE/support/scripts/vmmCallstack.pl
- cd to the callStackProfile.[phasenum]
- ./viewCallstack --text > ./call_stack_tree
- ./viewCallstack --text --rootAt myfuncname
- Viewing VMKernel functions alongside monitor functions
  - monitor.nmistats = 1
  - sudo chmod a+rw /dev/vmkcall (This only needs to be run once after each boot-up)
  - vsish -e set /perf/vmkstats/command/start
  - vsish -e set /perf/vmkstats/command/stop
  -

Reference:

callstack

https://wiki.eng.vmware.com/CallstackProfiling

3.25 VMKStats

- Setting:
  - config default: Configure with the default event (unhalted_clock_cycles). If no other event is configured, vmkstats automatically configures the default event at startup.
    - vsish -e set /perf/vmkstats/command/config default

- config event: Use this command if you want to configure a different event type.
  - vsish -e set /perf/vmkstats/command/config eventname eventsel=value unitmask=value
- config remove: Remove any existing configuration and free up the reserved counter.
  - vsish -e set /perf/vmkstats/command/config remove
- period: Set period for the event. A sample will be taken after "period" number of events have elapsed..
  - vsish -e set /perf/vmkstats/command/period 5000
- XXX use 'periodmean=5000' on MN (?)
- userstack: Include profile for the given cartel. [Works on KLNext+ builds only]
  - vsish -e set /perf/vmkstats/command/userstack cartelID
- blockedtime: Include userworld blocked time if userworld profiling is enabled. [Works on KLNext+ builds only]
  - vsish -e set /perf/vmkstats/command/blockedtime 1
- start: Start profiling
  - vsish -e set /perf/vmkstats/command/start
- stop: Stop profiling but do not discard the collected samples.
  - vsish -e set /perf/vmkstats/command/stop
- reset: Throw away profiling data.
  - vsish -e set /perf/vmkstats/command/reset

-
  - esxcfg-nas -a -o pa-dbc1019.eng.vmware.com -s /dbc/pa-dbc1019/hillzhao/sandbox/esx50u2-1839049/ esx50u2
  - export VMTREE=/vmfs/volumes/esx50u2/bora
  - export BLDDIR=/vmfs/volumes/esx50u2/bora/build
  - export VMBLD=beta
  -
  - esxcfg-nas -a -o build-toolchain.eng.vmware.com -s /toolchain toolchain
  - mkdir -p /build
  - ln -s /vmfs/volumes/toolchain /build
  - mkdir -p /usr/local/lib/perl5
  - TOOLS=/build/toolchain/lin32
  - rm -f /usr/bin/perl
  - rm -f /usr/local/lib/perl5/5.8.8
  - ln -s $TOOLS/perl-5.8.8/bin/perl /usr/bin/
  - ln -s $TOOLS/perl-5.8.8/lib /usr/local/lib/perl5/5.8.8
  -
  - ln -s $TOOLS/python-2.5/bin/python /usr/bin/
  - ln -s $TOOLS/python-2.5/lib/python2.5/ /lib/
  -

- vsish -e set /perf/vmkstats/command/stop
- vsish -e set /perf/vmkstats/command/reset
- vsish -e set /config/Misc/intOpts/FindLongCLIs 10
- vsish -e set /perf/vmkstats/command/start
- vsish -e set /perf/vmkstats/command/stop
-
- $VMTREE/support/scripts/vmkcallstack.pl

- Result:

```
/vmfs/volumes/4ff50132-5711faea-0e15-d4ae5264110b/ubuntu12.04-32bit_1 #
$VMTREE/support/scripts/vmkcallstack.pl

VMTREE env is set. Will read the object files from
/vmfs/volumes/esx50u2/bora/build/scons/package/devel/linux32/beta/esx

Writing to new directory ./vmkstats-17Aug12-09:31:59

Pausing stats sampling while we gather data...

Executing
/vmfs/volumes/esx50u2/bora/build/scons/package/devel/linux32/beta/esx/apps/vmk
statsdumper/vmkstatsdumper -a -o ./vmkstats-17Aug12-09:31:59 -k
0x0000418000000000

Dumping status to ./vmkstats-17Aug12-09:31:59/status ...

Dumping images to ./vmkstats-17Aug12-09:31:59/images ...

Dumping samples to ./vmkstats-17Aug12-09:31:59/samples ...

Dumping callstacks to ./vmkstats-17Aug12-09:31:59/callStacks ...

Tags: k

Extracting symbol information from loaded modules...
  0x418000908000 0x4000 vmw_psp_rr
  0x418000c39000 0x4000 dvsdev
  0x418000c68000 0x15000 esxfw
  0x418000d42000 0x2f000 nfsclient
  0x418000c89000 0xb9000 vmfs3
  0x418000a2d000 0x4000 iscsi_linux
  0x418000b22000 0x11000 shaper
  0x418000db8000 0x64000 migrate
  0x418000d8e000 0x2a000 vmkstatelogger
  0x418000a0b000 0x2000 cnic_register
  0x418000d74000 0xb000 ipmi_msghandler
  0x41800092f000 0xd000 usb-storage
  0x418000a70000 0x42000 mptsas
  0x418000a31000 0x18000 bnx2i
  0x418000e1e000 0x5000 svmmirror
  0x418000b33000 0x1a000 cdp
  0x418000a02000 0x9000 netsched
  0x418000c7d000 0x9000 vmkapei
```

```
0x4180008c2000 0x5000 hid
0x418000ab2000 0x21000 lvmdriver
0x418000a21000 0xc000 cnic
0x418000d7f000 0xb000 ipmi_si_drv
0x418000000000 0x60b0a1 vmkernel
0x418000900000 0x2000 vmw_satp_local
0x418000a0d000 0x14000 bnx2
0x4180007e1000 0x4000 procfs
0x41800090f000 0x20000 libata
0x418000989000 0x30000 vmci
0x418000a49000 0x1e000 libfc
0x4180008cc000 0x34000 nmp
0x4180008c7000 0x5000 dm
0x418000b75000 0xc4000 tcpip3
0x418000891000 0x21000 usb
0x4180007eb000 0xa1000 vmklinux_9
0x418000903000 0x2000 vmw_psp_lib
0x418000b5f000 0x16000 fence_overlay
0x41800090c000 0x3000 vmw_psp_mru
0x418000d8a000 0x4000 ipmi_devintf
0x418000d71000 0x3000 dell
0x4180008b2000 0xa000 ehci-hcd
0x41800088d000 0x4000 random
0x418000e23000 0x31000 hbr_filter
0x418000c3d000 0x2b000 dvfilter
0x418000a6d000 0x3000 ata_piix
0x41800093c000 0xe000 vfat
0x4180009b9000 0x16000 iscsi_trans
0x418000c86000 0x3000 vmkibft
0x4180009cf000 0x33000 etherswitch
0x418000b4d000 0x12000 ipfix
0x4180007e5000 0x6000 vmkplexer
0x418000ad3000 0x36000 deltadisk
0x418000905000 0x3000 vmw_psp_fixed
0x41800094a000 0x37000 vprobe
0x418000902000 0x1000 vmw_satp_default_aa
0x418000e1c000 0x2000 cbt
0x41800088c000 0x1000 vmklinux_9_2_0_0
0x4180008bc000 0x6000 usb-uhci
0x418000b0b000 0x17000 heartbeat
```

```
  0x418000a67000 0x6000 libfcoe
  0x418000b09000 0x2000 multiextent
Results archived into directory ./vmkstats-17Aug12-09:31:59
Run ./viewCallstack from there to view results
/vmfs/volumes/4ff50132-5711faea-0e15-d4ae5264110b/ubuntu12.04-32bit_1 #
```

- Running .pl remotely
  - If for some reason you cannot run vmkcallstack.pl on ESX box (For example; if perl is not available), alternative method is to dump raw data on esx host and post process it on another linux machine.
    On Esx Host -
    mkdir vmkstatsDir
    vmkstatsdumper -d
    vmkstatsdumper -a -o <vmkstatsDir>

    On Remote Host -
    Copy over vmkstatsDir from Esx host to this host. If you have access to the corresponding build tree, then set the environment variable VMTREE and do the following:
    $VMTREE/support/scripts/vmkcallstack.pl --output=<vmkstatsDir> --visor --remote

    If you do not have access to the build tree, copy over the relevant object files to a directory (extract from .gz files) and do the following:
  - $VMTREE/support/scripts/vmkcallstack.pl --output=<vmkstatsDir> --visor --remote --imageLocations all:<objectDir>
    This will extract stats in <vmkstatsDir>
  - /build/storage60/release/bora-914609/bora/support/scripts/vmmCallstack.pl
    -
    - From sosreport get the uname-a
      - Linux nftlin15 2.6.32-279.el6.x86_64 #1 SMP Wed Jun 13 18:24:36 EDT 2012 x86_64 x86_64 x86_64 GNU/Linux
    - Download system.map from debuginfo.centos.org/6/x86_64/
    - /build/storage60/release/bora-914609/bora/support/scripts/vmmCallstack.pl --guestsym /dbc/pa-dbc1119/hillzhao/bugs/rpm/System.map-2.6.32-279.el6.x86_64
    - ./viewCallstackWithGuest --text
  - Cp vmkstats.tar.gz to ubuntu,
  - Cp vmcallstackview.jar to /support/tools/java/
  - ./viewcallstack

```
hillzhao@pa-dbc1019:/dbc/pa-dbc1019/hillzhao/bugs/own_esx_bug/vmkstats-17Aug12-10:11:59$ ./viewCallstack --text --printStats
[100.00] Callee Root [Cpu:All World:All Cartel:All Blocked:All] [0.00]
  -> [99.54] <lost samples> [99.54]
  -> [0.23] IDTEnter [0.23]
  |   -> [0.23] Int14_PF [0.00]
  |      -> [0.23] gate_entry [0.00]
```

```
|         -> [0.23] UserDoCopyOut [0.00]
|            -> [0.23] User_CopyOut [0.00]
|               -> [0.23] BC_ReadFHID [0.00]
|                  -> [0.23] UserFileReadv [0.00]
|                     -> [0.23] LinuxFileDesc_Read [0.00]
|                        -> [0.23] User_LinuxSyscallHandler [0.00]
|                           -> [0.23] gate_entry [0.00]
  -> [0.23] IDT_IntrHandler [0.23]
     -> [0.23] gate_entry [0.00]
        -> [0.23] Power_HaltPCPU [0.00]
           -> [0.23] CpuSchedIdleLoopInt [0.00]
              -> [0.23] CpuSched_IdleLoop [0.00]
                 -> [0.23] Init_SlaveIdle [0.00]
                    -> [0.23] SMPSlaveIdle [0.00]
Time and memory Stats for 1 callstacks
Timing Statistics (in ms):
FirstCS 135
Memory Statistics (in MB):
HeapMax 986 CurrHeap 743 CurrUsed 8 CurrFree 734
FirstCS 7
```

vmkstats

https://wiki.eng.vmware.com/VmkStats

https://wiki.eng.vmware.com/ESXProfiling

3.26 Performance Issue

- Get the network analysis result
  - s34.ads.admin.de.bbs-2012-07-13--06.31/commands$ cat vmware_-v.txt

    VMware ESXi 5.0.0 build-515841
  - vmsupport-net-analyzer.py 515841 vsi_traverse_-s.txt vsi_traverse_-s--l-0.txt net-dvs_-l.txt > ./hill_net_analyzer.txt
- in the vm-support dir of commands
  -
- Sched-stats
  - cp /build/storage25/release/bora-515841/bora/build/scons/package/devel/linux32/release/esx/vmvisor/sys/lib/libvmlibs.so ld_path/
  - export LD_LIBRARY_PATH=ld_path = (pwd)
  - /build/storage25/release/bora-515841/bora/build/esx/release/apps/sched-stats/sched-stats -c commands/vsi_traverse_-s--l-0.txt > hill_sched_stats

3.27 WinDBG

- Installing windb by \\hfu-dell\share\iso\m\Setup\WinSDKDebuggingTools_amd64\
- ~/bin/vmss2core -Wsv100341-hang.vmss to generate memory.dmp
- Windbg netmapping pa-dbc1019
- Crash dump add the the memory.dmp
- Copy SOS.dll(in 64bit) C:\Windows\Microsoft.NET\Framework64\v2.0.50727, vmwdbgx64.dll
- SOS
  - !threadpool
  - http://blogs.msdn.com/b/johan/archive/2007/11/13/getting-started-with-windbg-part-i.aspx
  - http://msdn.microsoft.com/en-us/library/bb190764.aspx
- Vmwdbgx64:
  - deadlock
  - https://wiki.eng.vmware.com/CPD/Platform/WindowsDebugging

3.28 OSP Install

- Installing the OSPs on most modern Linux systems (SLE 10+, RHEL 5+, Ubuntu 8.04+) can be done in the following steps:
  - Configure your package manager to point to the VMware Tools OSPs repository
  - Update your package manager's package cache (Ubuntu only)
  - Install the appropriate vmware-tools-esx-kmod for your running kernel.
  - Install vmware-tools-esx
- Installing osp tools for sles11sp2
  - /etc/zypp/repos.d/esx-50u2.repo
    - [esx-50u2]
    - name=esx-50u2
    - enabled=1
    - autorefresh=1
    - baseurl=http://build-squid.eng.vmware.com/build/mts/release/sb-1174683/publish/tools/esx/5.0u1/sles11.2/x86_64/
    - path=/
    - type=rpm-md
    - keeppackages=0
  - linux-ad08:~ # zypper clean
  - linux-ad08:~ # zypper refresh
  - linux-ad08:~ # zypper install vmware-tools-esx
  - linux-ad08:~ # zypper install vmware-tools-plugins-dndcp

3.29 Frobos

- export VMPROD=esx

- export VMBLD=obj
- export VMTREE=/dbc/pek2-dbc101/hillzhao/express_patch_sb/vsphere51u2/bora
- hillzhao@pek2-dbc101:/dbc/pek2-dbc101/hillzhao/esx50u2/bora/vmcore/frobos/runtime/scripts$ ./frobos-run -c
- esxcfg-nas -a -o build-toolchain.eng.vmware.com -s /toolchain toolchain
  - mkdir -p /build
  - ln -s /vmfs/volumes/toolchain /build
  - mkdir -p /usr/local/lib/perl5
  - TOOLS=/build/toolchain/lin32
  - rm -f /usr/bin/perl
  - rm -f /usr/local/lib/perl5/5.8.8
  - ln -s $TOOLS/perl-5.8.8/bin/perl /usr/bin/
  - ln -s $TOOLS/perl-5.8.8/lib /usr/local/lib/perl5/5.8.8
- esxcfg-nas -a -o pek2-dbc101.eng.vmware.com -s /dbc/pek2-dbc101/hillzhao/esx50u2/ esx50u2
  - export VMTREE=/vmfs/volumes/esx50u2/bora
  - export BLDDIR=/vmfs/volumes/esx50u2/bora/build
  - export VMBLD=obj
- ./frobos-run

```
#!/bin/sh


branch="esx50u2"

workingDir="/vmfs/volumes/4ff50132-5711faea-0e15-d4ae5264110b/hillzhao"

testname="818659-LOWEST_PRI"

loglevel=1


echo "Run frobos $branch"

./frobos-run --nobuild --nop4 --normOnPass --workingDir $workingDir
--resultDir $workingDir all:818659 -mm BT --grubargs "/loglevel $loglevel"


echo "copy the logs"

cp $workingDir/frobos-runlog  /vmfs/volumes/4ff50132-5711faea-0e15-
d4ae5264110b/hillzhao/$testname-$branch-frobos-log

cp $workingDir/runtime/$testname.0/vmware.log /vmfs/volumes/4ff50132-
5711faea-0e15-d4ae5264110b/hillzhao/$testname-$branch-vmware-log


echo "removing..."

rm -rf $workingDir/runtime/*


echo "finished"
```

3.30 Kdump

- system-config-kdump
- /boot/grub/grub.conf
  - crashkernel=128M@16M/etc/kdump.conf
- chkconfig kdump on
- service kdump start
- service kdump status
- echo 1 > /proc/sys/kernel/sysrq
- echo c > /proc/sysrq-trigger
-

Reference:

https://access.redhat.com/knowledge/docs/en-US/Red_Hat_Enterprise_Linux/6/html/Deployment_Guide/s2-kdump-configuration-gui.html

3.31 On line gdb vmx esx50

Debug userworld on ESXi 5.0 or higher from off-host

Say that we are debugging a daemon or other long running program which you wish to attach to, on ESXi 5.0, from remote Linux box.

1. set VMTREE/VMBLD

2. cp public keys of you Linux box to ESXi, so that ESXi does not require you to enter passwd when debugging.

- On linux box: ssh-keygen -t rsa (generate public/private keys in ~/.ssh/)
- scp ~/.ssh/id_rsa.pub root@<ESXi host>:/tmp/
- On ESXi, cat /tmp/id_rsa.pub >> /etc/ssh/keys-root/authorized_keys

  Note that from ESXi 5.0, the location of authorized_keys is: /etc/ssh/keys-<username>/authorized_keys
- Make sure that PermitRootLogin is set to 'yes' and PasswordAuthentication are set to 'no' in the /etc/ssh/sshd_config file
- /etc/init.d/SSH restart

3. $VMTREE/support/scripts/debug-uw -u root -r <ESXi host> /path/to/binary cartel-id

- the remote-host is required, it will be connected to via ssh
- /path/to/binary is the path as if you were on the ESXi host. The binary will be found from the mirror image (on linux box) created by debug-uw.

For example, a tmp directory is created on my linux box:

```
-bash-4.1$ ls /tmp/debug-uw.BHu9nNRA/

bin dev etc gdb.cmd lib lib64 opt proc productLocker sbin tmp usr var vmfs
vmimages vmupgrade
```

- cartel-id is the running process (as per ps)
- user is optional, defaults to root

3.3.2 On line gdb vmx on esx41u3

- Build vmtree to access source code, symbol (before this make toolchain mounted)
  - esxcfg-nas -a -o pek2-dbc101.eng.vmware.com -s /dbc/pek2-dbc101/hillzhao/esx41u3/ esx41u3
  - export VMTREE=/vmfs/volumes/esx41u3/bora
  - export BLDDIR=/vmfs/volumes/esx41u3/bora/build
  - export VMBLD=obj
  - mkdir -p /dbc
  - mkdir -p /dbc/pek2-dbc101
  - mkdir -p /dbc/pek2-dbc101/hillzhao
  - ln -s /vmfs/volumes/esx41u3 /dbc/pek2-dbc101/hillzhao/esx41u3
  - set solib-absolute-prefix /vmfs/volumes/esx41u3/bora/build/esx/obj/debugInfo/usr/lib/debug/
  - handle SIGPIPE nostop noprint pass
- ps | grep vmx
  - find the least num of pid
- gdb /bin/vmx
  - attach pid

1. On esx host, start gdbserver
~#vmkgdbd 5010
Listening on port 5010

/.ssh #ps | grep vmx
139224 139224 vmx /bin/vmx
131043 139224 mks:win-2k3-32sp2 /bin/vmx
131044 139224 vcpu-0:win-2k3-32sp2 /bin/vmx

2. copy unstrapped vmx to esxi
scp ./vmvisor/sys-unstripped/bin/vmx root@10.117.5.146:/vmfs/volumes/datastore1\\(1\)/vmx-un
ln –s /vmfs/…/vmx-un /bin/vmx

3. on dbc, gdb connect gdbserver
a)
$ ./bora/support/scripts/debug-uw -r 10.117.5.146 /dbc/pek2-dbc101/hillzhao/esx41u3/bora/build/esx/obj/vmware-vmx 139224
(gdb) bt
#0 0x0000000054977092 in _start () from /lib64/ld-linux-x86-64.so.2
#1 0x000003ffcb999390 in ?? ()
#2 0x000003ffcbd56210 in ?? ()
#3 0x0000000000018680 in ?? ()
#4 0x000000000000000c in ?? ()
#5 0x000003ffcb99c500 in ?? ()
#6 0x000000001f478fc1 in ?? ()
#7 0x0000000000000000 in ?? ()
(gdb) info threads
  3 Thread 3 (#131044 vcpu-0:win-2k3-32sp2) 0x000000001f47cbfa in ?? ()
  2 Thread 2 (131043 mks:win-2k3-32sp2) 0x0000000054977092 in _start () from /lib64/ld-linux-x86-64.so.2
* 1 Thread 1 (139224 vmx) 0x0000000054977092 in _start () from /lib64/ld-linux-x86-64.so.2
(gdb) set solib-absolute-prefix /dbc/pek2-dbc101/hillzhao/esx41u3/bora/build/scons/package/devel/linux32/obj/esx/uwlibs/
(gdb) shared
(gdb) info sharedlibrary
No shared libraries loaded at this time.
(gdb) bt
#0 0x0000000054977092 in ?? ()
#1 0x000003ffcb999390 in ?? ()
#2 0x000003ffcbd56210 in ?? ()
#3 0x0000000000018680 in ?? ()
#4 0x000000000000000c in ?? ()
#5 0x000003ffcb99c500 in ?? ()
#6 0x000000001f478fc1 in ?? ()
#7 0x0000000000000000 in ?? ()
(gdb) info threads
  3 Thread 3 (#131044 vcpu-0:win-2k3-32sp2) 0x000000001f47cbfa in ?? () <<<<NO symbol
  2 Thread 2 (131043 mks:win-2k3-32sp2) 0x0000000054977092 in ?? ()
* 1 Thread 1 (139224 vmx) 0x0000000054977092 in ?? ()
warning: Couldn't restore frame in current thread, at frame 0
0x0000000054977092 in ?? ()
(gdb)

Or
$ vmkgdb64 (or gdb)
(gdb) target remote 10.117.5.146:5010
Remote debugging using 10.117.5.146:5010
0x08065b7d in ?? ()
(gdb) monitor !ps | grep vmx
135736 135736 vmx   /bin/vmx

```
135740 135736 mks:win-2k3-32sp2   /bin/vmx
135741 135736 vcpu-0:win-2k3-32sp2 /bin/vmx
(gdb) attach 135736
A program is being debugged already.  Kill it? (y or n) n
Not killed.
(gdb) monitor attach 135736
(gdb) bt
#0  0x08065b7d in ?? ()
(gdb) info threads
  3 Thread 3 (#135741 vcpu-0:win-2k3-32sp2)  0x00000001 in ?? ()
  2 Thread 2 (135740 mks:win-2k3-32sp2)  0x00000004 in ?? ()
* 1 Thread 1 (135736 vmx)  0x0000000c in ?? ()
warning: Couldn't restore frame in current thread, at frame 0
0x0000000c in ?? ()
(gdb) set solib-absolute-prefix /dbc/pek2-
dbc101/hillzhao/esx41u3/bora/build/scons/package/devel/linux32/obj/esx/uwlibs/
(gdb) shared
(gdb) info sharedlibrary
No shared libraries loaded at this time.
(gdb) bt
#0  0x0000000c in ?? ()
(gdb) info threads
  3 Thread 3 (#135741 vcpu-0:win-2k3-32sp2)  0x00000001 in ?? () <<<<no symbol>>>>
  2 Thread 2 (135740 mks:win-2k3-32sp2)  0x00000004 in ?? ()
* 1 Thread 1 (135736 vmx)  0x0000000c in ?? ()
warning: Couldn't restore frame in current thread, at frame 0
0x0000000c in ?? ()
(gdb)
```

3.3.3 *Debug userworld on ESXi 5.0 or higher from off-host*

Say that we are debugging a daemon or other long running program which you wish to attach to, on ESXi 5.0, from remote Linux box.

1. set VMTREE/VMBLD
2. cp public keys of you Linux box to ESXi, so that ESXi does not require you to enter passwd when debugging.
   - On linux box: `ssh-keygen -t rsa` (generate public/private keys in ~/.ssh/)
   - `scp ~/.ssh/id_rsa.pub root@<ESXi host>:/tmp/`
   - On ESXi, `cat /tmp/id_rsa.pub >> /etc/ssh/keys-root/authorized_keys`

     Note that from ESXi 5.0, the location of authorized_keys is: /etc/ssh/keys-<username>/authorized_keys
   - Make sure that PermitRootLogin is set to 'yes' and PasswordAuthentication are set to 'no' in the /etc/ssh/sshd_config file
   - /etc/init.d/SSH restart
3. $VMTREE/support/scripts/debug-uw -u root -r <ESXi host> /path/to/binary cartel-id

- the remote-host is required, it will be connected to via ssh
- /path/to/binary is the path as if you were on the ESXi host. The binary will be found from the mirror image (on linux box) created by debug-uw.

For example, a tmp directory is created on my linux box:

```
-bash-4.1$ ls /tmp/debug-uw.BHu9nNRA/

bin dev etc gdb.cmd lib lib64 opt proc productLocker sbin tmp usr var vmfs vmimages vmupgrade
```

- cartel-id is the running process (as per ps)
- user is optional, defaults to root

3.34 On-host debugging(higher than esxi5.0)

This wiki shows how to debug a simple userlevel application(let's say, busybox).

1. Make sure you are debugging on a 'obj' host.

2. Prepare your libraries:

- In most cases, you would like to have **debuginfo** in your shared-libraries, so you need to put them on the host, and set solib-path when debugging.

You can find debug-version libs in **bora/build/scons/package/devel/linux32/obj/esx/uwlibs/usr/lib/debug/**

- The binary will not recognize lib*.so.debug when loading, we need rename them(on dbc-box):

```
find bora/build/scons/package/devel/linux32/obj/esx/uwlibs/usr/lib/debug/ -type l -name '*.debug' | xargs rename .debug ""
```

- Scp the above directory to your host, or simply mount the VMTREE to host

3. Start gdb

- set solib-absolute-prefix ${path_to_the_debug_libs}
- use 'dir' to include source code files if neccessary

3.35 Debug coredump for esx35

- copy all from esx35 /usr/lib/vmware/lib/*.so to linux
- mkdir /bugs
- mkdir /build
- mkdir /build/storage60
- mkdir /build/toolchain
- mkdir /build/apps
- mkdir /usr/lib/vmware
- mkdir /exit14
- mkdir /exit14/home
- mkdir /bldmnt
- mount -t nfs bugs.eng.vmware.com:/bugs /bugs
- mount -t nfs build-storage60.eng.vmware.com:/storage60 /build/storage60

- mount -t nfs build-toolchain.eng.vmware.com:/toolchain /build/toolchain
- mount -t nfs build-toolchain.eng.vmware.com:/apps /build/apps
- export VMPROD=esx
- export VMBLD=release
- export VMTREE=/build/storage60/release/bora-110268/bora
- root@bo-virtual-machine:/bugs/files/0/0/4/4/7/9/4/1/sr1484570535/vm-support-iadadobdmi03p-2010-02-01--18.15.29702/vmfs/volumes/49f783e8-4f346819-b8da-002219c7e0b4/nO8fsaTTTXNU6iyA9epFNQ# /build/apps/bin/esx/vmkgdb64-7 $VMTREE/build/release/server/vmware-vmx vmware-vmx-core.000

3.36 Kgdb linux kernel

- make 编
  make modules_install 安
  make install 安
- Target linux kernel: add this “kgdboc=ttyS0,115200 kgdbwait” on grub.conf
- Debug linux: gdb vmlinux, then (gdb) set remotebaud 115200
- (gdb) target remote /dev/ttyS0
- echo g > /proc/sysrq-trigger
- add-symbol-file /root/scull/scull.ko 0xd099a000

Reference:

http://blog.csdn.net/jie12310/article/details/4564853

3.37 Debug Bios

- debugStub.listen.guest32 = "TRUE"
- debugStub.hideBreakpoints= "TRUE"
- monitor.debugOnStartGuest32 = "TRUE"
- gdb
  - target remote localhost:8832
  - file BIOS.440.DBG
  - file vmlinux
  - b *0x7c00
  - b handle_IRQ_event

Reference:

http://wiki.osdev.org/VMware

3.38 linux kernel redirect to console (ttyS0)

- console=ttyS0,115200n8 console=tty0, earlyprintk=ttyS0,115200n8 loglevel=7
- Find the kernel line (grub config file) which corresponds to your currently running kernel. Add the following at the end of that line - console=tty0 console=ttyS0,9600n8:

```
title Red Hat Enterprise Linux ES (2.6.9-42.0.10.ELsmp)
       root (hd0,0)
```

```
        kernel /vmlinuz-2.6.9-42.0.10.ELsmp ro root=LABEL=/ console=tty0
console=ttyS1,19200n8
        initrd /initrd-2.6.9-42.0.10.ELsmp.img
```

- Create the serial port ttyS0 with redirect to one file. (/vmfs/volumes/50df5849-72c0f35b-8bd3-000c29396c43/centos-5.5-32-sunk/serialport.log)

3.39 install MAC OVF

- deploy ovf
- If failed, use the vmdk directly bug create new virtual machine.
-

3.40 Dubug workstation vmware-vmx

- make vmcore-exports
- netmap the dbc directory to windows
- local.mk
  - export PRODUCT=ws
  - export OBJDIR=obj
  - export VERBOSE=5
  - export ARCH=x64
  - export NUM_CPUS=24
  - export WIN32_LINUX_BUILDROOT=Y:/hillzhao/express_patch_sb/hosted11-pd-rel/bora/build
  - export I_AM_SLOPPY=1
- make vmx
- make ws-all | tee ws-all.log
- cp vmware-vmx-debug to workstation directory
- cp vc dependence dll
  - cp Microsoft.VC80.DebugCRT
- windbg attach vmware-vmx-debug.exe process
  - .reload /f vmware-vmx-debug.exe
- vmx.buildType = "debug"
- vmx.noUIBuildNumberCheck = "TRUE"

3.41 generate vmss from vmx-suspend.txt

- reconstruction.sh
  - cat `ls vmfs/volumes/8c739d9a-3299ab0b/V-W2K8R2-WEB-CHT-64-100\ \(d788d9a3-fbcc-4f38-a25f-465a8b4f3dfc\)/debug-hung-vm_vmfsvolumes8c739d9a-3 299ab0bV-W2K8R2-WEB-CHT-64-100\(d788d9a3-fbcc-4f38-a25f-465a8b4f3dfc\)V-W2K8R2-WEB-CHT-64-100\(d788d9a3-fbcc-4f38-a25f-465a8b4f3dfc\)vmx-suspend.txt. FRAG-* | sed 's/\(.*\)\(FRAG-\)\(.*\)/\3\t\1\2\3/' | sort -n | cut -f 2` > vmfs/volumes/8c739d9a-3299ab0b/V-W2K8R2-WEB-CHT-64-100\ \(d788d9a3-fbcc-4f 38-a25f-465a8b4f3dfc\)/debug-hung-vm_vmfsvolumes8c739d9a-3299ab0bV-W2K8R2-WEB-CHT-64-100\(d788d9a3-fbcc-

4f38-a25f-465a8b4f3dfc\)V-W2K8R2-WEB-CHT-64-1 00\(d788d9a3-fbcc-4f38-a25f-465a8b4f3dfc\)vmx-suspend.txt

- tar xzvf vmfs/volumes/8c739d9a-3299ab0b/V-W2K8R2-WEB-CHT-64-100d788d9a3-fbcc-4f38-a25f-465a8b4f3dfc/debug-hung-vm_vmfsvolumes8c739d9a-3299ab0bV-W2K8R2-WEB-CHT-64-100\(d788d9a3-fbcc-4f38-a25f-465a8b4f3dfc\)V-W2K8R2-WEB-CHT-64-100\(d788d9a3-fbcc-4f38-a25f-465a8b4f3dfc\)vmx-suspend.txt
- ~/bin/vmss2core -W V-W2K8R2-WEB-CHT-64-100\ \(d788d9a3-fbcc-4f38-a25f-465a8b4f3dfc\)-6e460272.vmss

3.42 Build ESXi and VMX in Local Linux

- adduser hillzhao
- mkdir -p /dbc/pek2-dbc101/hillzhao/
- p4 sync prod2013-stage code
-
- mount -t nfs exit14.eng.vmware.com:/vol/vol0/home /exit14/home
- Actually do following mount...
- mount -t nfs build-toolchain.eng.vmware.com:/toolchain /build/toolchain
- mkdir /build/apps
- mount -t nfs build-toolchain.eng.vmware.com:/apps /build/apps
-
- /exit14/home/mts/build_mounts/build_mounts.pl
- export PATH=$PATH:/build/toolchain/lin32/binutils-2.20.1/bin/:/build/apps/bin/
-
- mount -t nfs build-toolchain.eng.vmware.com:/mts /build/mts
- scons vmx

3.43 Esxtop Replay

- cat commands/vmware_-v.txt
- debug.env.sh build_xxx_no
- $ export LD_LIBRARY_PATH=$VMTREE/../build/linux64/bora/build/esx/$VMBLD/vmvisor/sys/usr/lib:$VMTREE/../build/linux64/bora/build/esx/$VMBLD/vmvisor/sys/lib
- $VMTREE/../build/linux64/bora/build/esx/release/vmvisor/sys/lib/ld-linux.so.2 $VMTREE/build/scons/package/devel/linux32/$VMBLD/esx/apps/esxtop/esxtop -R .

If you want to gdb debug esxtop, we can use

- gdb /build/storage60/release/bora-1065491/bora/../build/linux64/bora/build/esx/release/vmvisor/sys/lib/ld-linux.so.2
- (gdb) run ~/bin/esxtop -R .

Or on host

- esxcfg-nas -a -o build-storage60.eng.vmware.com -s /storage60 storage60
- esxcfg-nas -a -o pek2-dbc101.eng.vmware.com -s /dbc/pek2-dbc101/hillzhao/esx51u1/ esx51u1
- gdb /vmfs/volumes/08f171bd-dbcf5569/bora/build/esx/obj/apps/esxtop/esxtop

- b main
- r -R esx-top/esx-alvhcamp71.wdr.de-2013-06-03--13.59/
- break bora/lib/vmkctl/advstats/vscsiStatsImpl.cpp:131 (only after run esxtop, the shared library symbol can be loaded)
- break bora/lib/vmkctl/advstats/vscsiStatsImpl.cpp:460 if i == 530

3.44 Git.eng.vmware.com

- git clone ssh://git@git.eng.vmware.com/private/hillzhao/tools
- git add -A #All those new files we've added
- git commit -am "First commit"
- git push origin master

Reference:

https://wiki.eng.vmware.com/Git

3.45 SVS

- p4 login -s
- p4 login -a
- # or for ALL perforce servers (mind the underscore in p4_login)
- p4_login -A –a
- svs submit --upload-p4ticket
- dbc -t svs submit -c 2310029 --testsuite=svs-esx-suite --buildtarget=esxall:release
- OR svs precheckin -c 2310029 --testsuite=svs-esx-suite --buildtarget=esxall:release
  - If exec svs failed use not “dbc –t”
  - If file need resolve, edit –c /reopen (p4 edit -c 2310029 public/vm_basic_math.h)

Reference:

https://wiki.eng.vmware.com/SVS

3.46 VMDBSH

- Shell script run in dbc

```
BRANCH_ROOT=/dbc/pek2-dbc101/hillzhao/esx51u1

VMDBSH=$BRANCH_ROOT/bora/build/build/vmdbsh/obj/uw32/vmdbsh
ESX_SERVER_HOST=10.117.7.240
ESX_SERVER_USERNAME=root
ESX_SERVER_PASSWD=
VMX_PATH=/vmfs/volumes/4ff50132-5711faea-0e15-
d4ae5264110b/ubuntu1204_64/ubuntu1204_64.vmx

#/db/connection/#1/
#[/]$ mount /vm
#/vm/#_VMX/guest/guestInfo/config/nicInfo/
#bget xdr

#echo "/vm/#_VMX/vmx/mstat/#vm/#heartbeat/value"
```

```
#$VMDBSH -h
while true; do

printf "%-15s%-15s%-15s\n" "heartbeat" "uptime" "runningStatus"

$VMDBSH -e "connect -t auth -H $ESX_SERVER_HOST -O 902 -U
$ESX_SERVER_USERNAME -P '$ESX_SERVER_PASSWD' -v $VMX_PATH" \
    -e "mount /vm"
```

- Run in host
  - vmdbsh -e "connect -v /vmfs/volumes/4ff50132-5711faea-0e15-d4ae5264110b/ubuntu1204_64/ubuntu1204_64.vmx"
  - mount vm

3.47 Windbg Commands

.hh command	get help on a particular command (open up in a new window)
dd 0x12345678	dump memory at address 0x12345678, showing 1 dword
dd 0xFFFFF80001234567 L20	dump memory at address 0xFFFFF80001234567, showing 0x20 dwords
~1	change to CPU 1
kv	show current stack, verbose
kd 20	dump raw stack in hex form, 20 words, possibly resolving some symbol names
!analyze -v	show verbose bugcheck analysis (note that VMs suspended and converted to dumps will have a non-fatal bugcheck)
lmv	list modules loaded (verbose, includes file dates, checksums, versions, etc)
lm	list modules loaded (not verbose)
r	dump registers (general-purpose, segment, iopl, eflags, current instruction)
!pte fffffadf`e2d0c000	show (guest perspective) pagewalk information for address fffffadf`e2d0c000
!process 0	list processes
!process 0 0xf	list processes with lots of verbosity (0xf is the OR of four flags bits all set)
!thread fffffadfe5d39bf0	list thread at fffffadfe5d39bf0 (includes various time/process stats, stack, ...)
dt _THREAD 0xfffff800abcdefab	dump "_THREAD" structure at address 0xfffff800abcdefab
!pcr	dump processor control register for current processor
!locks	dump locks, those threads holding/waiting on them (both shared and exclusive)
!dpcs 1	The !dpcs extension displays the deferred procedure call (DPC) queues for a specified processor.
!irpfind	dump outstanding I/O request packets (including possibly some which have been retired) in the non-paged pool
!handle	Show the handles, contains process handle
.formats	Show different formats of number
uf nt!KiSaveProcessorControlState	Print functions assamble code

!running !running -it	All processor runing threads
.trap b839290c	The .trap command displays the important registers for the specified trap frame. also instructs the kernel debugger to use the specified context record as the register context
.cxr	The .cxr command displays the context record saved at the specified address. It also sets the register context
!idt	The !idt extension displays the interrupt service routines (ISRs) for a specified interrupt dispatch table (IDT).
!irql	The !irql extension displays the interrupt request level (IRQL) of a processor on the target computer before the debugger break.
!drvobj \Driver\pvscsi	Dump the driver object info
dt storport!_RAID_UNIT_EXTENSION fffffa8006f91b10 PendingQueue.	Show the struct storport
!stacks [Detail] !stacks 2 ndis!	displays information about the kernel stacks. Only show ndis related stacks
!ready	displays summary information about each thread in the system in a READY state.
!for_each_process ".process /p /r @#Process;!ntsdexts.locks"	List each process locks
logopen c:\stacks.txt; ! for_each_process "!process @#Process; ~*kv1000"; .logclose	List each process stack
vertarget	Information about CPU, system
!for_each_process ".process /p /r @#Process;!ntsdexts.locks"	Check each threads locks
.load wow64exts !for_each_thread "!thread @#Thread 1f;.thread /w @#Thread; .reload; kb 256; .effmach AMD64"	Check 32bit thread in 64 core dump
!pcitree	
!ndiskd.minidriver fffffa8007aeabb0 –handlers !ndiskd.miniport fffffa8007c051a0	List ndis extension displays a miniport block.
!verifier	The **!verifier** extension displays the status of Driver Verifier and its actions.
!runaway	snapping a user mode dump of the process
!exqueue	The !exqueue extension displays a list of items currently queued in the ExWorkerQueue work queues.

3.48 Valgrind

- Compile
  - git clone ssh://git@git.eng.vmware.com/cayman_esx_valgrind.git
  - cd cayman_esx_valgrind
  - git checkout vmkernel-main
    - git checkout remotes/origin/vmkernel-main-uw-1(for check in)
  - git submodule update --init --recursive
  - cd esx_valgrind
  - make all
- Run Valgrind
  - esxcfg-nas -a -o pek2-dbc202.eng.vmware.com -s /dbc/pek2-dbc202/hillzhao/src/cayman_esx_valgrind/build/obj/esx64/valgrind/install/opt/valgrind/ valgrind\
  - export VALGRIND_ESXI=1
  - export VALGRIND_BIN=/vmfs/volumes/valgrind/bin
  - export VALGRIND_LIB=/vmfs/volumes/valgrind/lib/valgrind
  - vsish -e set /config/User/intOpts/UserProcEnable 1
  - scp build/build/vmx/obj/uw64/vmware-vmx root@10.117.8.208:/vmfs/volumes/chwang-db51-1/centos-5.5-32-sunk/vmware-vmx
  - ln –s /vmfs/volumes/chwang-db51-1/centos-5.5-32-sunk/vmware-vmx /bin/vmx-debug
  - /vmfs/volumes/valgrind/bin/valgrind --trace-syscalls=yes -d -d -d --leak-check=full /bin/vmx-debug ++swap=false -qx /vmfs/volumes/50df5849-72c0f35b-8bd3-000c29396c43/centos-5.5-32-sunk/centos-5.5-32-sunk.vmx >valgrind_debug.log 2>&1
  - /vmfs/volumes/valgrind/bin/valgrind --leak-check=full /bin/vmx-debug -qx /vmfs/volumes/50df5849-72c0f35b-8bd3-000c29396c43/valgrind_test/valgrind_test.vmx >valgrind_debug.log 2>&1
- Debug coredump
  - gdb ./obj/esx64/valgrind/build/memcheck/memcheck-amd64-linux memcheck-amd64-linux-core.000

Reference:

https://wiki.eng.vmware.com/CPD/ESX/Valgrind

3.49 Codeviz

```
root@tim-virtual-machine:/home/hillzhao/vmcore-main/bora# history | grep apt-get
  500  apt-get install zlib1g-dev
  537  apt-get install ibpopt-dev
  538  apt-get install libpopt-dev
  539  apt-get install binutils-dev
```

```
  589  apt-get install g++
  592  apt-get install binutils-dev
 1183  apt-get install htop
 1191  apt-get install sysstat
 1194  apt-get install linux-tools-common
 1196  apt-get install linux-base
 1198  apt-get install linux-tools-3.2.0-23
 1212  apt-get install grapviz
 1213  apt-get install graphviz
 1250  apt-get install nautilus-open-terminal
 1251  apt-get install binutils
 1252  apt-get install tree
 1253  apt-get install bison
 1308  history | grep apt-get

 ===build codeviz
mkdir codeviz
cd codeviz/
gcc -v
apt-get install graphviz
tar xzvf codeviz-1.0.11.tar.gz
cd codeviz-1.0.11/
ls compilers/
cp ../gcc-3.4.6.tar.gz compilers/
cd compilers/
./install_gcc-3.4.6.sh
apt-get install nautilus-open-terminal
apt-get install binutils
apt-get install tree
apt-get install bison
export C_INCLUDE_PATH=/usr/include/x86_64-linux-gnu && export
CPLUS_INCLUDE_PATH=$C_INCLUDE_PATH
export OBJC_INCLUDE_PATH=$C_INCLUDE_PATH
export LIBRARY_PATH=/usr/lib/x86_64-linux-gnu
add --disable-multilib;
vi codeviz/codeviz-1.0.11/compilers/install_gcc-3.4.6.sh
../gcc-3.4.6/configure --prefix=$INSTALL_PATH --enable-shared --enable-
languages=c,c++ --disable-multilib || exit
PLATFORM=x86_64-unknown-linux-gnu
export LANG=en_US
./configure
```

```
make
make install

/usr/local/bin/gcc -o test test.c
genfull
gengraph --output-type gif -f main

=====build codeviz for 4.6.3 gcc====
http://gmplib.org/,http://www.mpfr.org/,http://www.multiprecision.org/
M4
1)  tar zxvf m4-1.4.1.tar.gz
2)  cd m4-1.4.1
3)  ./configure
4)  make
5)  make check
6)  make install
GMP
1)tar jxvf gmp-4.3.2.tar.bz2
2)cd gmp-4.3.2
3)./configure
4)make
5)make check
6)  make instal
MPFR
1)  tar jxvf mpfr-2.4.2.tar.bz2
2)  cd mpfr-2.4.2
3)  ./configure --with-gmp-include=/usr/local/include --with-gmp-
lib=/usr/local/lib
4)  make
5)  make check
6)  make install
MPC
1)  tar zxvf mpc-0.8.1.tar.gz
2)  cd mpc-0.8.1
3)  ./configure --with-gmp-include=/usr/local/include --with-gmp-
lib=/usr/local/lib
4)  make
6) make check
7)  make install
```

```
export C_INCLUDE_PATH=/usr/include/x86_64-linux-gnu && export
CPLUS_INCLUDE_PATH=$C_INCLUDE_PATH
export OBJC_INCLUDE_PATH=$C_INCLUDE_PATH
export LIBRARY_PATH=/usr/lib/x86_64-linux-gnu
add --disable-multilib;
vi codeviz/codeviz-1.0.11/compilers/install_gcc-4.6.3.sh
../gcc-4.6.3/configure --prefix=$INSTALL_PATH --enable-shared --enable-
languages=c,c++ --disable-multilib || exit
export LANG=en_US
./configure
make
make install


===build prod2013 code from local Linux machine====
  adduser hillzhao
 mkdir -p /dbc/pek2-dbc101/hillzhao/
 p4 sync prod2013-stage code


mount -t nfs exit14.eng.vmware.com:/vol/vol0/home /exit14/home
Actually do following mount...
  mount -t nfs build-toolchain.eng.vmware.com:/toolchain /build/toolchain
  mkdir /build/apps
  mount -t nfs build-toolchain.eng.vmware.com:/apps /build/apps


/exit14/home/mts/build_mounts/build_mounts.pl
  export PATH=$PATH:/build/toolchain/lin32/binutils-
2.20.1/bin/:/build/apps/bin/


mount -t nfs build-toolchain.eng.vmware.com:/mts /build/mts
  scons vmx


===codeviz call graph====
===build gcc==
 hillzhao@pek2-dbc202:/dbc/pek2-
dbc202/hillzhao/src/cayman_esx_toolchain/cayman_esx_toolchain$ rm
../build/obj/stamps/gcc-stamp
  make all
=== make gcc tarball===
cd /dbc/pek2-dbc202/hillzhao/src/cayman_esx_toolchain/build/publish
rm usr.tar.gz
tar czvf usr.tar.gz usr
scp usr.tar.gz hillzhao@pek2-dbc101:/dbc/pek2-dbc101/hillzhao/log/codeviz-
usr.tar.gz
```

```
====build vmx==
hillzhao@pek2-dbc101:/dbc/pek2-dbc101/hillzhao/prod-2013/prod2013-
stage/bora/build/package/COMPONENTS/cayman_esx_toolchain/ob-939563/linux64$ rm
-rf usr
cp /dbc/pek2-dbc101/hillzhao//log/codeviz-usr.tar.gz ./
tar xzvf codeviz-usr.tar.gz


find ./ -iname "*.c.cdepn" -print
rm -rf build/build/vmx-nonvmcore/
find ./ -iname "*.c.cdepn" -print | xargs rm
scons vmx
vi ./vmx/main/vmx.c.cdepn
==== remove no use path===
F {VMX_BoostVCPUThread} {bora/vmx/main/vmx.c:2715}
C {VMX_BoostVCPUThread} {bora/vmx/main/vmx.c:2931}
{VThread_AdjustThreadPriority}
C {VMX_BoostVCPUThread} {bora/vmx/main/vmx.c:2931} {NumVCPUs}
C {VMX_BoostVCPUThread} {bora/vmx/main/vmx.c:2931}
{VThread_AdjustThreadPriority}
C {VMX_BoostVCPUThread} {bora/vmx/main/vmx.c:2931} {VThread_VCPUIDToThreadID}
C {VMX_BoostVCPUThread} {bora/vmx/main/vmx.c:2931}
{VThread_AdjustThreadPriority}
C {VMX_BoostVCPUThread} {bora/vmx/main/vmx.c:2931} {MonitorLoop_RunningVCPUs}
C {VMX_BoostVCPUThread} {bora/vmx/main/vmx.c:2931}
{VThread_AdjustThreadPriority}
C {VMX_BoostVCPUThread} {bora/vmx/main/vmx.c:2931} {Panic}
C {VMX_BoostVCPUThread} {bora/vmx/main/vmx.c:2931} {__builtin_expect}


to >>>


F {VMX_BoostVCPUThread} {vmx.c:2715}
C {VMX_BoostVCPUThread} {vmx.c:2931} {VThread_AdjustThreadPriority}
C {VMX_BoostVCPUThread} {vmx.c:2931} {NumVCPUs}
C {VMX_BoostVCPUThread} {vmx.c:2931} {VThread_AdjustThreadPriority}
C {VMX_BoostVCPUThread} {vmx.c:2931} {VThread_VCPUIDToThreadID}
C {VMX_BoostVCPUThread} {vmx.c:2931} {VThread_AdjustThreadPriority}
C {VMX_BoostVCPUThread} {vmx.c:2931} {MonitorLoop_RunningVCPUs}
C {VMX_BoostVCPUThread} {vmx.c:2931} {VThread_AdjustThreadPriority}
C {VMX_BoostVCPUThread} {vmx.c:2931} {Panic}
C {VMX_BoostVCPUThread} {vmx.c:2931} {__builtin_expect}
```

```
genfull
gengraph --output-type gif -d 6 -f VMX_BoostVCPUThread
```

3.50 Debug windows Tools crash dump

- Map \\build-storage60.eng.vmware.com to local device X
- In buildweb.eng.vmware.com get the ESXi version 1065491's tools version 1065307
- Add vmtools directory to symbol path in windbg
  - X:\release\bora-1065307\build\windows-2008\bora-vmsoft\build\release-x64\tools-for-windows\Win32\services\vmtoolsd;X:\release\bora-1065307\build\windows-2008\bora-vmsoft\build\release-x64\tools-for-windows\win32\apps\vmtoolslib;X:\release\bora-1065307\build\windows-2008\bora-vmsoft\build\release-x64\tools-for-windows\win32\services\plugins\hgfsServer
- .reload
- 0:000> .ecxr

```
rax=0000000000000001 rbx=00000000001ff0f0 rcx=0000000000000000
rdx=0000000001ece5a0 rsi=0000000000000000 rdi=0000000001fbf040
rip=000007fef22af312 rsp=00000000001fefc0 rbp=000000000000002b
 r8=000000000000002b  r9=00000000001ff0f0 r10=0000000001ec2be0
r11=0000000001ece5a0 r12=0000000001ece5a0 r13=0000000000000000
r14=000000000000001e r15=0000000000000000
iopl=0         nv up ei pl nz na po nc
cs=0033  ss=002b  ds=002b  es=002b  fs=0053  gs=002b
efl=00010206
vmtools!Message_Send+0x32:
000007fe`f22af312 0fb701          movzx   eax,word ptr [rcx]
ds:00000000`00000000=????
```

- 0:000> kb

```
  *** Stack trace for last set context - .thread/.cxr resets it
RetAddr           : Args to Child
: Call Site
000007fe`f22afc88 : 00000000`001ff0f0 00000000`0000002b 00000000`00000000
00000000`004dd0bb : vmtools!Message_Send+0x32 [d:\build\ob\bora-
1065307\bora-vmsoft\lib\message\message.c @ 179]
000007fe`f2263e02 : 00000000`01ebb330 00000000`01ebc5f0 00000000`01ebc5f0
00000000`004dd101 : vmtools!RpcOut_send+0x18 [d:\build\ob\bora-1065307\bora-
vmsoft\lib\rpcout\rpcout.c @ 160]
000007fe`f2061183 : 000007fe`f20622c8 00000000`001ff9b0 00000000`001ff2d8
00000000`00000000 : vmtools!RpcInSend+0x72 [d:\build\ob\bora-1065307\bora-
vmsoft\lib\rpcchannel\bdoorchannel.c @ 184]
00000001`3f305bce : 00000000`03e0e2a0 00000000`01ea90c0 00000000`03d64a28
00000000`001ff240 : hgfsServer!HgfsServerCapReg+0xa3 [d:\build\ob\bora-
1065307\bora-vmsoft\services\plugins\hgfsserver\hgfsplugin.c @ 122]
00000000`00345b90 : 00000000`00000000 00000000`001ff950 00000000`001ff950
00000000`00523b24 : vmtoolsd!
g_cclosure_user_marshal_POINTER__POINTER_BOOLEAN+0x6e [d:\build\ob\bora-
1065307\bora-vmsoft\build\release-x64\tools-for-
windows\win32\services\vmtoolsd\subdirs\services\vmtoolsd\svcsignals-gm.c @
88]
00000000`003603e2 : 00000000`03e0e2a0 00000000`001ff2d8 00000000`00000003
00000000`01eab5a0 : gobject_2_0!g_closure_invoke+0x240
[c:\toolchain\src\glib-2.22.4-1\glib-2.22.4\gobject\gclosure.c @ 772]
00000000`0035f55c : 00000000`01ebc710 00000000`00000000 00000000`01ea90c0
00000000`001ff550 : gobject_2_0!signal_emit_unlocked_R+0x8e2
[c:\toolchain\src\glib-2.22.4-1\glib-2.22.4\gobject\gsignal.c @ 3253]
00000000`0035faab : 00000000`01ea90c0 00000000`00000002 00000000`00000000
```

```
00000000`001ff6f0 : gobject_2_0!g_signal_emit_valist+0x91c
[c:\toolchain\src\glib-2.22.4-1\glib-2.22.4\gobject\gsignal.c @ 2995]
00000001`3f3034ac : 00000000`01ea90c0 00000001`3f3092e0 00000000`001ff9b0
00000000`00000000 : gobject_2_0!g_signal_emit_by_name+0x18b
[c:\toolchain\src\glib-2.22.4-1\glib-2.22.4\gobject\gsignal.c @ 3075]
00000001`3f30208e : 00000000`00000000 00000001`3f309430 00000000`00000000
00000000`00000000 : vmtoolsd!ToolsCore_UnloadPlugins+0x4c [d:\build\ob\bora-
1065307\bora-vmsoft\services\vmtoolsd\pluginmgr.c @ 870]
00000001`3f302611 : 00000000`001ff950 00000001`3f3024c0 00000000`001ff950
00000001`3f308c00 : vmtoolsd!ToolsCoreCleanup+0x1e [d:\build\ob\bora-
1065307\bora-vmsoft\services\vmtoolsd\mainloop.c @ 47]
00000001`3f301e57 : 00000000`01fbfa30 00000000`01fbf680 00000000`01fbcc30
00000000`00000000 : vmtoolsd!ToolsCoreRunLoop+0xe1 [d:\build\ob\bora-
1065307\bora-vmsoft\services\vmtoolsd\mainloop.c @ 227]
00000001`3f307199 : 00000000`00000990 00000000`00000000 00000000`00000003
00000000`01fb5dc0 : vmtoolsd!CToolsService::Run+0x567 [d:\build\ob\bora-
1065307\bora-vmsoft\services\vmtoolsd\mainwin32.cpp @ 461]
00000001`3f301880 : 00000000`00000003 00000000`01fbcc30 00000000`01ea7370
00000000`01ea7376 : vmtoolsd!CNTService::StartServiceW+0x89
[d:\build\ob\bora-1065307\bora\lib\ntservice\ntservice.cpp @ 635]
00000001`3f30735a : 00000000`00000000 00000000`00000000 00000000`01eb87f0
00000000`01fb8630 : vmtoolsd!wmain+0x350 [d:\build\ob\bora-1065307\bora-
vmsoft\services\vmtoolsd\mainwin32.cpp @ 884]
00000000`76b4652d : 00000000`00000000 00000000`00000000 00000000`00000000
00000000`00000000 : vmtoolsd!__tmainCRTStartup+0x11a
[f:\dd\vctools\crt_bld\self_64_amd64\crt\src\crtexe.c @ 583]
00000000`76d7c521 : 00000000`00000000 00000000`00000000 00000000`00000000
00000000`00000000 : kernel32!BaseThreadInitThunk+0xd
00000000`00000000 : 00000000`00000000 00000000`00000000 00000000`00000000
00000000`00000000 : ntdll!RtlUserThreadStart+0x1d
```

3.51 Vprobe for Monitor

- Config vprobe enable
  - /etc/vmware/config.
  - vprobe.allow = TRUE
  - <path .vmx>
  - vprobe.enable = "TRUE"
  - vprobe.unsupportedEnable = "TRUE"
- /vmfs/volumes/4ff50132-5711faea-0e15-d4ae5264110b/ubuntu1204_64 # vim-cmd vmsvc/getallvms

```
Vmid                Name
File                                                    Guest OS      Version
Annotation
1       esx41u3                                 [LocalStore (1)]
esx41u3/esx41u3.vmx
vmkernel5Guest    vmx-08
2       ubuntu1204_64                           [LocalStore (1)]
ubuntu1204_64/ubuntu1204_64.vmx
ubuntu64Guest     vmx-07
3       ubuntu12.04-32bit                       [LocalStore (1)] ubuntu12.04-
32bit_1/ubuntu12.04-32bit.vmx                                     ubuntuGuest
vmx-08
5       [backup]fangchiw_rhel6_64_server    [LocalStore (1)]
[backup]fangchiw_rhel6_64_server/[backup]fangchiw_rhel6_64_server.vmx
rhel6_64Guest     vmx-07    Rhel 6 64bit Workstation edition.
```

- vprobe -c 'VMM1Hz printf("hi %d\n", VCPUID);' –m

- /vmfs/volumes/08f171bd-dbcf5569/bora/vmcore/support/vprobes/cookbook/vm # vprobe -m 2 vt-exit-fast.emt

3.52 Build Static monitor binary

- Get the execute commands and print the staticVMMFile

```
bora/vmcore/support/debug/vmmvmkstacksyms.pl
      $linkerScript = $buildTree .
                      "/bora/vmcore/support/debug/modular-to-static-
linker.pl";
      GDBWVMM::SetLinkerScript(0, $linkerScript);
   }
   GDBWVMM::SetLinker($GDBWUTIL::binary, undef);
   GDBWVMM::LinkStaticVMM(undef, $vmwareLog, $GDBWUTIL::binary);

   print($GDBWVMM::staticVMMFile);
/vmfs/volumes/50df5849-72c0f35b-8bd3-000c29396c43/valgrind_test #
$VMTREE/vmcore/support/debug/vmmvmkstacksyms.pl -l vmware.log
```

- Link the monitor dynamically

```
/vmfs/volumes/50df5849-72c0f35b-8bd3-000c29396c43/centos-5.5-32-sunk #
esxcfg-nas  -l
prod2013-stage-valgrind is /dbc/pek2-dbc202/hillzhao/src/prod2013-stage-
valgrind from pek2-dbc202.eng.vmware.com mounted available
toolchain is /toolchain from build-toolchain.eng.vmware.com mounted
available
valgrind is /dbc/pek2-
dbc202/hillzhao/src/cayman_esx_valgrind/build/obj/esx64/valgrind/install/opt
/valgrind/ from pek2-dbc202.eng.vmware.com mounted available
```

Linking monitor executable

```
/vmfs/volumes/50df5849-72c0f35b-8bd3-000c29396c43/centos-5.5-32-sunk #
/build/toolchain/lin32/perl-5.10.0/bin/perl /vmfs/volumes/acde89a9-
67ac4879/bora/vmcor
e/support/debug/modular-to-static-linker.pl --linker /vmfs/volumes/prod2013-
stage-valgrind/bora/build/esx/obj/vmcore-exported/obj/linker --log
vmware.log --p
hase last --vmm ./vmm64 --search /vmfs/volumes/prod2013-stage-
valgrind/bora/build/esx/obj/vmcore-exported/obj --find-binaries-script
/vmfs/volumes/prod2013-s
tage-valgrind/bora/support/scripts/find-binaries
```

For 50u3

```
perl $VMTREE/vmcore/support/debug/modular-to-static-linker.pl
--log vmware.log  --phase last --search $VMTREE/build/esx/
$VMBLD/vmcore-exported/$VMBLD --vmm myvmm
```

3.53 Build BIOS

- Use 32bit Windows
- Copy the BIOS2002 to the windows
- Correct the NUBIOS = c:\Hill\BIOS2002
  - Add set TCROOT=T:

```
hillzhao@pek2-dbc101:/dbc/pek2-
```

```
dbc101/hillzhao/express_patch_sb/vsphere51u2/BIOS2002$ cat build.bat
@echo off

REM
REM Set %NUBIOS% to the top level directory where you checked out BIOS2002
REM
set NUBIOS=c:\src\B2

REM
REM No user configurable parts below this line
REM
set OEM=%NUBIOS%\OEM\VMWARE\440BX338
set MTOOLS=%NUBIOS%\TOOLS600
set NUCORE=600
set PATH=%PATH%;%MTOOLS%
set TCROOT=T:


cd %oem%
echo Type 'makmaker' to rebuild the makefiles if make.mak has changed.
echo Type 'nmaker' to build all, 'nmaker quick' if only OEM changed.
echo Type 'mapconv' after building to generate a relocated BIOS.MAP.
echo Type 'verxpm' after generating BIOS.MAP to validate Win7 XP Mode
support.
```

- nmaker clean && makmaker && nmaker && mapconv /overwrite

- Type 'makmaker' to rebuild the makefiles if make.mak has changed.
- Type 'nmaker' to build all, 'nmaker quick' if only OEM changed.
  - Make sure there is no failures - this generates DEVEL1c3.ROM under BIOS2002\OEM\VMWARE\440BX338
- Type 'mapconv /overwrite' after building to generate a relocated BIOS.MAP.
  - This generates BIOS.MAP under BIOS2002\OEM\VMWARE\440BX338
- You can run all three commands above in one chain - full build is 'nmaker clean && makmaker && nmaker && mapconv /overwrite'
- Type 'verxpm' to verify that XP Mode license number region is correctly located
  - Unfortunately 'mapconv' exits with non-zero status code, so you cannot chain 'verxpm' to the command above.
- Compare generated NVRAM.LST with vmx/data/nvram/nvram.v4.lst.
  - If there is a difference, do not check-in change, and instead investigate what you did not change NVRAM layout, and change your code to not trigger NVRAM change.
- Rename the DEVEL1C3.ROM to BIOS.440.ROM and BIOS.MAP BIOS.440.MAP, and copy them to bora\vmx\data\bios - these are the files needs to check into bora.
  - If you are using Linux for checkin, do not forget to convert line endings in BIOS.440.MAP to Unix style.

Refer:

https://wiki.eng.vmware.com/MakingBIOSChanges

3.54 Build EFI

- Install 32bit dev in 64bit redhat
  - yum install glibc.i686
  - yum install libstdc++.so.6
- Download the source of EFI
  - Source is on perforce.eng.vmware.com:1666.
  - //depot/EFIROM/vmcore-main/EDK2/... //your-client/...
- Mount the toolchain in 64bit redhat linux
  - mkdir /build/toolchain
  - mount -t nfs build-toolchain.eng.vmware.com:/toolchain /build/toolchain
- Download efi toolchain, and build
  - perforce-toolchain.eng.vmware.com:1666.
  - //toolchain/Proj/EFIROM/lin32/... //your-client/lin32/...

- ln -s /home/hillzhao/toolchain/lin32 /build/efirom-toolchain/lin32
- ln –s /build/toolchain /build/mts/toolchain
- goto EDK2 directory
  - make toolchain (can build the toolchain in dbc, and then copy to redhat linux64) then make below
  - make VMBLD=release publish

Reference:

https://wiki.eng.vmware.com/VirtualEFI/Building

https://wiki.eng.vmware.com/BeatTheBugKBs/BuildBIOSandEFI

https://p4web.eng.vmware.com/@rev1=head@//depot/EFIROM/vmcore-main/EDK2/VmwPkg/Doc/Build.txt

[rhel-6-64-esx41 160] /home/hillzhao/efirom/EDK2/publish > pwd

/home/hillzhao/efirom/EDK2/publish

10.117.8.219

3.55 system tap

- yum install systemtap systemtap-runtime
- yum install kernel-devel
- rpm --force -ivh kernel-debuginfo-common-x86_64-2.6.32-71.el6.x86_64.rpm
- rpm --force -ivh kernel-debuginfo-2.6.32-71.el6.x86_64.rpm

```
[rhel-6-64-esx41 13] /home/hillzhao/linux/systemtap > cat irgnote.stp
#!/usr/bin/stap
/*
global action

function dumpstat() {
    foreach ( [a] in action- ) {
        printf ("%d:%d ", a, action[a])
    }
    printf ("\n")
//    delete action
}
```

```
*/
//probe begin { printf ("Starting probe, irq %d\n", $1) }
probe begin { printf ("Starting probe") }

probe kernel.function("scsi_io_completion") {
      print_backtrace();println("")
}

/*
probe jbd2.function("do_get_write_access") {
      print_backtrace();println("")
}
probe kernel.statement("handle_IRQ_event") {
      print_backtrace();println("")
}
probe kernel.function("note_interrupt") {
      action[$action_ret]++
      print_backtrace();println("")
}
*/

//probe timer.s(2) { dumpstat() }
[02:56 - 0.01]
```

```
0xffffffff8134a380 : scsi_io_completion+0x0/0x550 [kernel]
 0xffffffff81341812 : scsi_finish_command+0xc2/0x130 [kernel]
 0xffffffff8134649e : scsi_eh_flush_done_q+0x7e/0x160 [kernel]
 0xffffffff81363561 : ata_scsi_error+0x481/0x910 [kernel]
 0xffffffff81347846 : scsi_error_handler+0x126/0x630 [kernel]
 0xffffffff81091936 : kthread+0x96/0xa0 [kernel]
 0xffffffff810141ca : child_rip+0xa/0x20 [kernel]
 0xffffffff810918a0 : kthread+0x0/0xa0 [kernel] (inexact)
 0xffffffff810141c0 : child_rip+0x0/0x20 [kernel] (inexact)

 0xffffffff8134a380 : scsi_io_completion+0x0/0x550 [kernel]
 0xffffffff81341812 : scsi_finish_command+0xc2/0x130 [kernel]
 0xffffffff8134649e : scsi_eh_flush_done_q+0x7e/0x160 [kernel]
 0xffffffff81363561 : ata_scsi_error+0x481/0x910 [kernel]
 0xffffffff81347846 : scsi_error_handler+0x126/0x630 [kernel]
 0xffffffff81091936 : kthread+0x96/0xa0 [kernel]
 0xffffffff810141ca : child_rip+0xa/0x20 [kernel]
 0xffffffff810918a0 : kthread+0x0/0xa0 [kernel] (inexact)
 0xffffffff810141c0 : child_rip+0x0/0x20 [kernel] (inexac
```

3.56 Migrate History

- export VSISH="/dbc/pa-dbc1119/hillzhao/bin/automagicallyrun.sh vsish"
- migrateSummary <src vsicache> <dst vsicache>
- migrateHistory <vsicache>

3.57 Build driver For Linux

- Create Local.mk

```
export VERBOSE=4
export NUM_CPU=24
#export MAKE_CROSS=0
#export CROSSCOMPILE_TOP_DIR=/build/trees/crosscompile
#export DEBUG_FORCE_UNICODE=1
#export PRODUCT=tools-for-freebsd
#export PRODUCT=tools-for-windows
```

```
export PRODUCT=tools-for-linux
#export PRODUCT=tools-for-solaris
export ARCH=x86
#export ARCH=x64
#export OBJDIR=release
export OBJDIR=obj
```

- Make vmxnet3

4 Bugs

4.1 Windows

- MS KB
- Pm timer
- Windbg
- Hal
- Xperf
- smooth_acpi_timer
- Questions:
  - How to define which CPU wrong?
  -

4.2 VM hang pattern

- Jul 21 04:45:57.482: vmx| GuestRpcSendTimedOut: message to toolbox-dnd timed out.
- Jul 18 08:41:01.688: vmx| GuestRpcSendTimedOut: message to toolbox timed out

4.3 VM Freeze Ask Input

- VMsample
- GuestOS top
- When freeze, when unfreeze, then get which part time log and vmss useful.
- Can ping?
- Vm-support –Z(contain sample too) or –X, vm-support -Z will turn sample on before collect vmss file. What's parameter stands for each?
- Is really hang, how to define the hang?
- From esx, how to see the cpu usage of vm?
- Vcsupport, how to use it?
- Perfmon to help on this.
- logging in using the remote console, to see whether non-responsible.
- How to check windows newest SP up to date?
- Not SP up to date, there are some issue for SP2, it's hot fix.
- creating a batch file and running on all the VM's. The batch file can contain the steps of modifying the registry.

- "vm-support -k"
- OPT Build is the build-type kstats?

Reference:

Virtual machine console intermittently displays a black screen and reports high CPU usage

http://knova-prod-kcc-vip.vmware.com:8080/contactcenter/php/search.do?cmd=displayKC&docType=kc&externalId=2017851&sliceId=1&docTypeID=DT_KB_1_1&dialogID=407330748&stateId=1%200%20407474460

5 Products

Reference:

http://www.vmware.com/cn/products/datacenter-virtualization/vsphere

6 ESX

6.1 *Virtual Overview*

- Hypervisor VMBare Metal 此O 直ype 1”。
- Hosted 私Hypervisor 富2”。能 功 且 而

- 虚ll Virtulization)

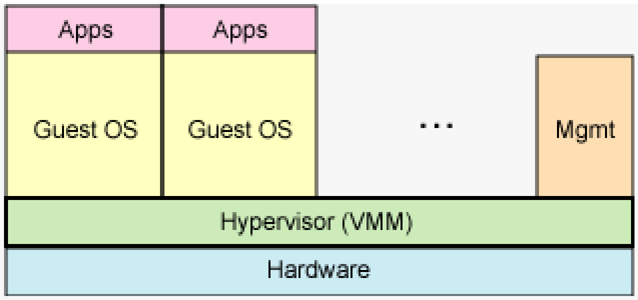
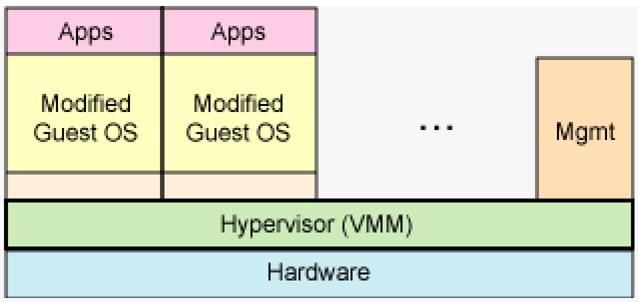


图2. 全虚拟化架构图

- 半虚拟化（Parairtulization）

  Xen



图

- 硬件辅助虚拟化（Hardware Assisted Virtualization）

  Intel 的VT-x 和AMD的AMD-V

  - 无VMM虚
  - 减VMM参
- 系A
  - CPU Ring 3)下
  - 特CPU 的
- Trap-And-Emulation

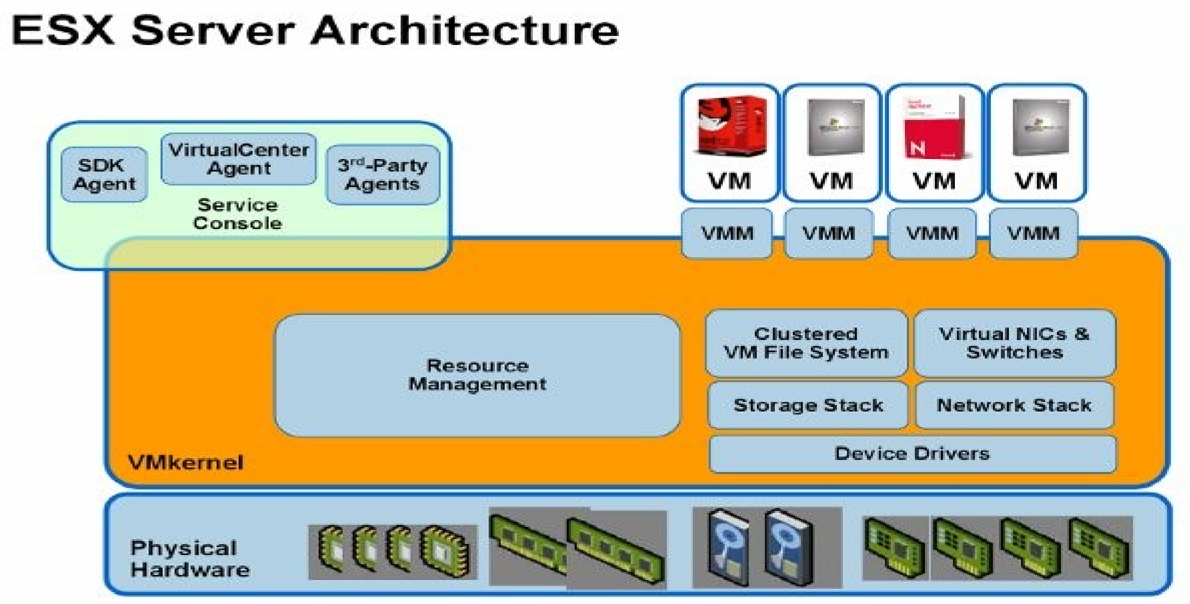
  Guest OS Guest OS Guest OS VT VMM VMM VMM Guest OS Guest OS R3 VMM VMM Guest OS，Guest OS VMM“Trap-And-Emulate”Guest OS Guest OS VMM

Reference:

http://en.wikipedia.org/wiki/Virtualization

http://it20.info/2007/06/a-brief-architecture-overview-of-vmware-esx-xen-and-ms-viridian/

6.2 ESX Overview



- VMware ESX 4
  - Infrastructure Service 就VM的IO 等CPU 和256GB 内存PU内
  - 增加frastructure Service 除frastructure Service 之ESX 4 通VMDirectPath 能 I/O 性能torage 的hin Provisioning 和inked Clone %右
  - Application Service 主VMware vCenter Agent 来VMware vCenter 同VMware vCenter vMotion 和A 类
- ESX4.0 版
- Service Console

Service Console 是edhat Enterprise OS虽然实现虚拟化能,但这

  - 启VMKernel 当SX Service Console 接inux runlevel 3 启VMKernel 之VMKernel 当VMKernel 启 Service Console 就MKernel 面
  - 提eb 以DK 接irtualCenter Agent 及irtual Center otion 和RS 等
  - 能VMkernel 的Service Console 的roc 录ervice Console 数 的SXTOP 命 能 性
  - 运ervice Console 提
  - 负D-ROM 等
  - 注ervice Console 提VMware rvice Console 经这 OS 的IB 内O Service Console 映virtual(关 磁 虚 Disk)

6.3 ESX Run (PXE Boot)

6.4 CPU 虚拟化

- g Compression ary Translation VM 和uest x86 架VM 运ing 0 下Guest 内ing 1 下Guest 应ing 3 下VMM uest 二 释,要 有 是
  - v pgtable,%r3" VM uest OS 不 间 OS eneral Protection Exception est OS 到VM 于VMM R3 寄 uest OS 将table est Physical Address 填R3 寄VM 置table 转est Physical Address 到R3 寄uest OS 中VM 通uest OS

成功age Fault，VMM 有Microsoft Virtual PC，VMware Workstation，Sun Virtual Box，Parallels Desktop for Mac 和QEMU。

- 硬件：Intel 的VT-x 和AMD的AMD-V 为VMM和Guest OS的VT-X 支持Root 模式Operation，VMM运行Non-Root 模式Operation，Guest OS运行Non-Root 模式，Guest 执行Root 模式VMM，让VMM来
- 进VMware 超虚拟
  - 纯x86 代码
  - 虚拟x86 指令
  - 集x86 虚拟x86 的 。 行 运

6.4.1 CPU 调度（Scheduler）

- CPU 的时间调CPU 虚拟VMware 采用CPU 调度Co-Scheduling”，是而“Gang-scheduling”，算法vSphere 中VMware 推出Co-Scheduling 和Relaxed Co-Scheduling，VMware 虚拟化的NUMA（Non-Uniform Memory Access），Hyperthreading，VM-Affinity 的 集 中 在

6.5 内存虚拟化

- 影子Shadow Page Table，Guest 和Guest OS 虚拟VMM 影MMU 进
- 硬件EPT（Extended Page Table），EPT 通EPT 虚拟Guest 的Host。
- **VMware 影**
  - 其Memory Overcommit 虚拟超 量 总 存 内
  - 其Page Sharing 它进Replica 页 之COW（Copy On Write）一的 内 为 的
  - 其Balloon Driver 其VMware Tools 可VMware Tools 有VMware 的Balloon Agent，Balloon Agent 在Balloon Driver 进程swapping 处 第 不 上 机

6.6 *I/O* 虚拟化

- 虚拟I/O 设Guest OS 而I/O 设Guest OS 对I/O 操VMM，让VMM 模Guest 而Guest 操SCSI 或IDE，ATA 的
- 硬件Intel 的VT-d，AMD的IOMMU 和PCI-SIG 的IOV（IO Virtulization）虚拟VT-d 其I/O 地DMA 的VT-d 通DMA 重Remapping 和I/O 页IOMMU 和VT-d 虚拟
- **VMKernel 的I/O 技术**
  - VMFS 是VMware 的I/O 而IO 的Distributed al 系统Block 为block 大小MB 的MB，MB 使 文 方 为 大 机 它
  - Virtual Switch 其VMKernel 的ESX 和STP（Spanning Tree protocol）协议Uplink，Server 交 中VMware 理 性 物 Virtual Switch 的Distributed Virtual Switch 它Virtual Switch 的
  - 虚拟VMDirectPath 虚拟IO 性PCI-SIG 的SR-IOV 虚拟PCIe 设Storage 的Thin Provisioning 和Linked Clone 虚拟%左右
  - 网UI 比vSphere Client 来
-

Reference:

[IO Virtualization]

http://www.intel.com/technology/itj/2006/v10i3/2-io/5-platform-hardware-support.htm

6.7 VMM

- 不VMM 虚拟

x86 处理器 Ring 0 ~ Ring 3，只有 Ring 0 ~ 2 级，Ring 0 级 x86 [illegible] Ring 0 和 Ring 3 [illegible] Ring 0 级 Ring 3 级 VMM 在 Ring 0 级 Guest OS 的 Guest OS 在 Ring 1 或 Ring 3 级 Ring 2 不使

- [illegible] Ring Compression）
  - VMM 使用 [illegible] Ring 0，1，2 为 VMM 的 Guest OS 的 Guest 在 Ring 3 级 VM 的 Guest OS 对 GDT，IDT [illegible] Guest OS 运行在 Ring 0 [illegible] Guest OS 对 Guest [illegible]
- [illegible] Ring Alias）
  - [illegible] Guest OS 在 VMM 的 Guest OS 在 x86 [illegible] Guest OS 的 push [illegible] mp [illegible] Guest OS 的 GDT，LDT，IDT 和 TR [illegible] push 自 %cs” [illegible] GDT 寄 gdt dest 值为 movl fake_gdt，dest”。 些
- 地址 Address Space Compression）
  - 地址 VM 的 Guest OS 的 GDT Register 中 Guest OS 的 VM 中 VM [illegible] Guest OS 的 VM 的 Guest OS 的 VM 内 Guest OS 的 GDT 等 VM 的 Guest OS 上
- [illegible] Guest OS 的 [illegible]
  - [illegible] VM 的 Guest OS 的 Guest Physical Address，Host Physical Address，Guest OS 的 VM [illegible] Guest 的 Guest Linear Address，Guest Physical Address 通 VM 为 Guest OS [illegible] Guest Linear Address 到 Host Physical Address 的 CR3 [illegible] VM 的 Guest OS 的 Guest Physical Address 到 Host Physical Address [illegible] Guest Linear Address，VM 的 Guest OS 的 Guest Physical Address [illegible] Host Physical Address 的 VM 的 VM 的 Guest OS 的 Guest OS 自
- [illegible] Guest OS 中的
  - 系统 SYSENTER/SYSEXIT [illegible] SYSENTER 指令 IA32_SYSENTER_CS，IA32_SYSENTER_EIP 和 IA32_SYSENTER_ESP 这 个 MSR（Model Specific Register）的 Ring 0 级 SYSEXIT 指令 Ring 0 级 VM 能 Trap-And-Emulate 的 指令
  - [illegible] VM 的 VM 的 Guest OS，VMM [illegible] VM 的 Guest OS 的 Guest OS 的 VM 的 Guest OS 的 VM 的 Guest OS 能 EFLAGS，VM 的 Guest OS 的 VM 需 。 转 发 断 中
- Guest OS [illegible]
  - Guest OS 对 VM [illegible] 到 Interrupts [illegible] Ring 4 [illegible] cycle，Local APIC，Task-Priority Register，IO-APIC [illegible] PR [illegible] PR [illegible] 性 将 优 （ 低

Reference:

http://www.tektalk.org/2010/04/12/%E5%89%96%E6%9E%90%E7%B3%BB%E7%BB%9F%E8%99%9A%E6%8B%9F%E5%8C%96%EF%BC%882%EF%BC%89-x86%E8%99%9A%E6%8B%9F%E5%8C%96%E6%8A%80%E6%9C%AF/

6.8 Intel-VT

- Root Operation & non-root Operation
  - VT-x 为 IA 32 [illegible] VMX root operation 和 VMX non-root operation，VMM 在 VMX root operation 模 VMX non-root operation 模 Guest OS [illegible] Ring 0 ~ Ring 3 这 VMM 和 Guest OS 都
  - 这 VMX 种操作模式可以互相转换 VMM 通过 VMLAUNCH 或 VMRESUME 指令 VMX non-root operation 模 Guest OS 的 Guest OS VM entry，Guest OS 通 VM 处理 VMCALL 指令 VM 的 Guest OS VMX 动调起 operation 模 挂 主 VMM VM exit，VMX root operation [illegible] x 从 VMX non-root operation [illegible] exit 指。 些 某
  - VMM 和 Guest OS 共享 VMCS [illegible] Guest State Area，Host State Area，VM entry 时 Guest OS [illegible] VM 的 VMM [illegible] Guest OS 的 VMM 将 Guest OS，而 VMM 的 VM entry 时 VMM 从 VM exit 时 Guest OS 的 VMM [illegible] VM entry 和 exit 的 VM exit，VM entry 时 Guest OS “暂停”
  - [illegible] CR3 [illegible] ESP 和 EIP 如 64 位 RSP，RIP，CS，SS，DS，ES，FS，GS [illegible] GDTR，IDTR 寄 器 IA32_SYSENTER_CS，IA32_SYSENTER_ESP，IA32_SYSENTER_EIP 和 IA32_PERF_GLOBAL_CTRL 等 MSR 寄 VMM 的 VM exit [illegible] 区 存 态 寄个 的 Active State 使 Guest OS 从 Active [illegible] Triple Fault [illegible] Shutdown 状态

  - [illegible] VM entry 和 VM exit 的条
    1. External-interrupt exiting 用 VM exit 而 Guest OS 是否
    2. Interrupt-window exiting 如 Guest OS [illegible] VM exit。

3. Use TPR shadow通CR8 访Task Priority RegisterTPR 的VMCS 中TPR 可VM exit TPR 值Guest OS 置R VM exit。
4. CR masks and shadows每个 VMCS est OS 致Guest OS。

- VMCS 涵
  1. Exception bitmap M exit，
  2. I/O bitmap 对16 位I/O 端VM exit。
  3. MSR bitmaps SR

致VM exit 时VMCS MM entry 时VM 或Guest OS VMCS 中Guest OS 的T 的Guest OS 的

Reference:

http://www.ibm.com/developerworks/cn/linux/l-cn-vt/

http://www.intel.com/content/www/us/en/processors/architectures-software-developer-manuals.html

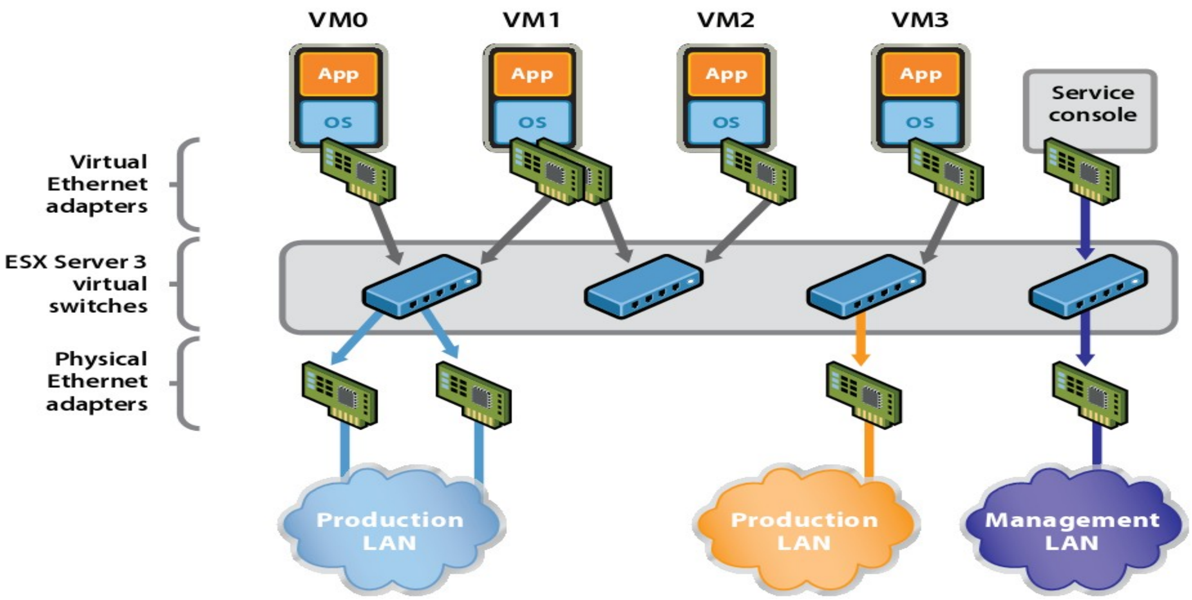
[tel VT-d 术

http://www.intel.com/technology/itj/2006/v10i3/2-io/7-conclusion.htm

[VMM ARCH]

http://www.intel.com/technology/itj/2006/v10i3/2-io/3-vmm-software-architecture.htm

6.9 *Virtual Networking*



- rtual Ethernet Adapter）
  - vLance MD Lance PCNet32 Mware Tools
  - e1000 tel E1000 0% ware Tools
  - vmxnet Mware ware Tools Mware nxnet2 和mxnet3 nxnet3 SIMessage Signaled Interrupt）MSI-X和MDirectPath v6。

装

- Port Group
  - Virtual Switch
  - VLAN 的D。
  - NIC Teaming

- Layer 2 的

6.9.1 Virtual Switch

- 基 Layer 2 forwarding)。
- 系 VLAN 的 VLAN 及 VLAN 发 Service Level 配置
- 除 Virtual Switch 还 NIC Teaming 负
- Layer 2 Forwarding，在 Virtual Switch 层 VLAN 的
  - Virtual Guest Tagging (VGT)：Guest OS 数 VLAN 标 tagging 来 VLAN 的
  - External Swicth Tagging (EST)：数 VLAN 在 VLAN 信
  - Virtual Switch Tagging (VST)：VMware 环 Virtual Switch 来 VLAN 标 Guest OS 是 Guest OS 要 VLAN 信

下 VLAN 的示

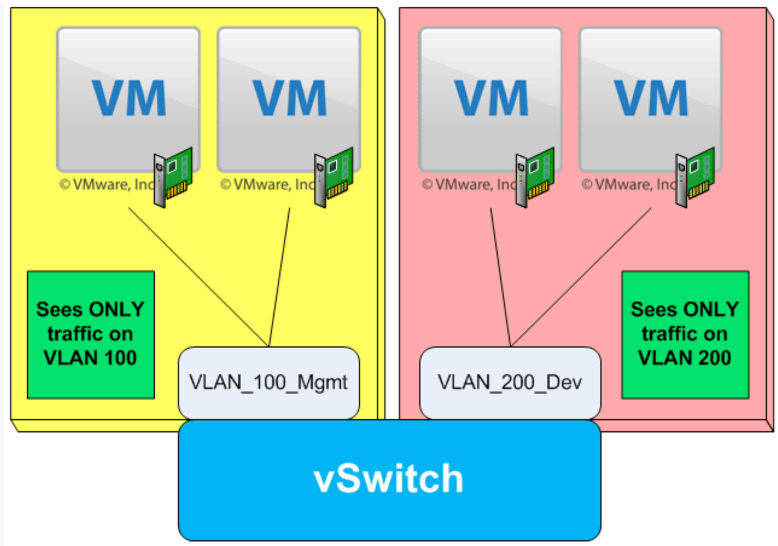
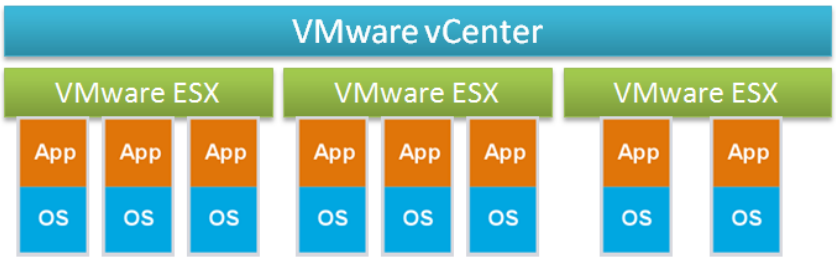


图 VLAN 示例)

- Virtual Switch 的
  - Promiscuous mode 混 VLAN 的
  - MAC Address Change 改 MAC 地 MAC 地 ARP Poisoning 攻 MAC 地
  - Forged Transmit 伪 MAC 地
  - 虚 Virtual Switch 的 VMware 的 Virtual Switch 以 Virtual Switch 流 outbound traffic only 带 Average Bandwidth 带 Peak Bandwidth 控 Burst Size 其 Average Bandwidthd 的 值
- NIC Teaming
  - NIC Teaming 是 VMware 在 VMware Infrastructure 3 中 Virtual Switch 进 的 移 转 障
  - 在 S 或 VS 要 在 Virtual Switch 上 据 送 Virtual Switch 式 MAC 地 IP Hash 其 Explicit Failover Order。
- vMotion
  - vMotion 是 VMware 资源 Virtual 化 Switch 优 参 用 作 等 来 面 合 页

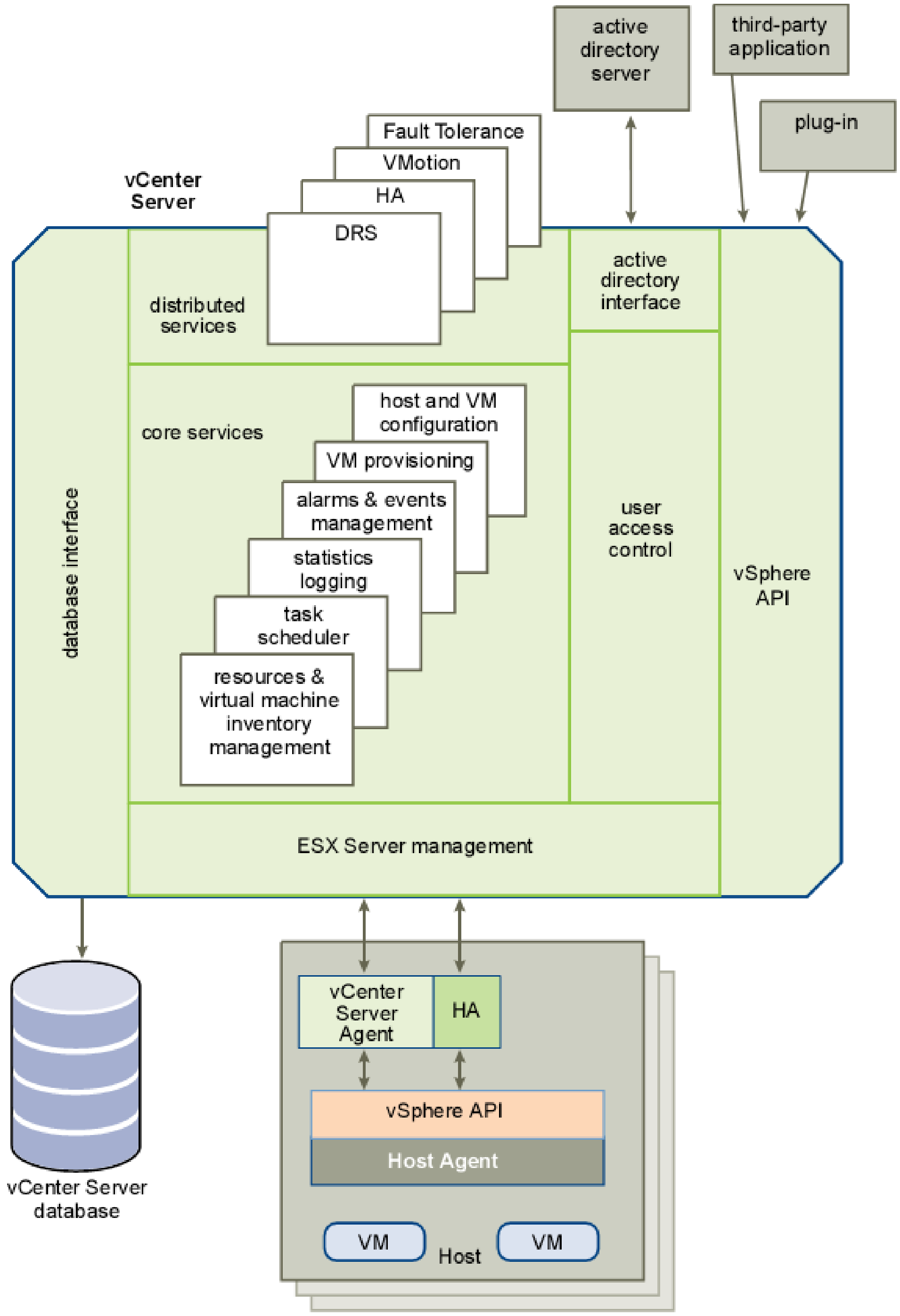
7 vSphere

- VMware vSphere 是VMware ESX 4 的VMware vCenter，vSphere 的



7.1 vCenter

- VMware vCenter 是VMware vSphere 环Sphere 可L Server Express Oracle 的Center 统
vSphere 机Center Server Agent ESX API 供Center 三 第 多 很



图Virtual Center 架构

- 资源Center 管CPU 和Memory 等
- 备Motion), 突冲 者 或 源 资
- 拔Wizard 业App 程部
- 动Mware 摆Motion 技Mware 在Mware Infrastructure 3.5 版Storage vMotion 通

- 在VMware 提供DRS(Distributed Resource Scheduler)分vSphere 和VMware 在VMware Infrastructure 3.5 版DPM(Distributed Power Management),DRS 的
- 在VMware 推VMsafe API 通API 能够otkit 和Checkpoint、IBM、McAfee、Symantec 和TrendMicro 以及VMSafe API 的VMware Shield Zones 它通过vShield 可以使用,可以监视记录
- 在VMware Fault Tolerance 是VMware 的Lockstep 技术Shadow 虚拟 ,
- 在VMware High Availability 技术Heartbeat 检Heartbeat 或者 响应 机 拟
- 在VMware 推VMCB(VMware Consolidated Backup 技Agent 的 装
- 在vApp 是VMware 最OVF(Open Virtualization Format 把vApp 便管理的 关 相 了
- vCenter ConfigControl 可以
- vCenter CapacityIQ 可以
- vCenter Chargeback 提供对虚拟机的成本数据跟踪使
- vCenter Orchestrator 可以
- vCenter AppSpeed 监控解 够来 能施 它 措
- vCenter 和vCenter 进程
- 在VMware 还vCenter 和VMware Lifecycle Manager 用Lab Manager 及Stage Manager 和Site Recovery Manager 等

8 Source Code

8.1 *Kstats*

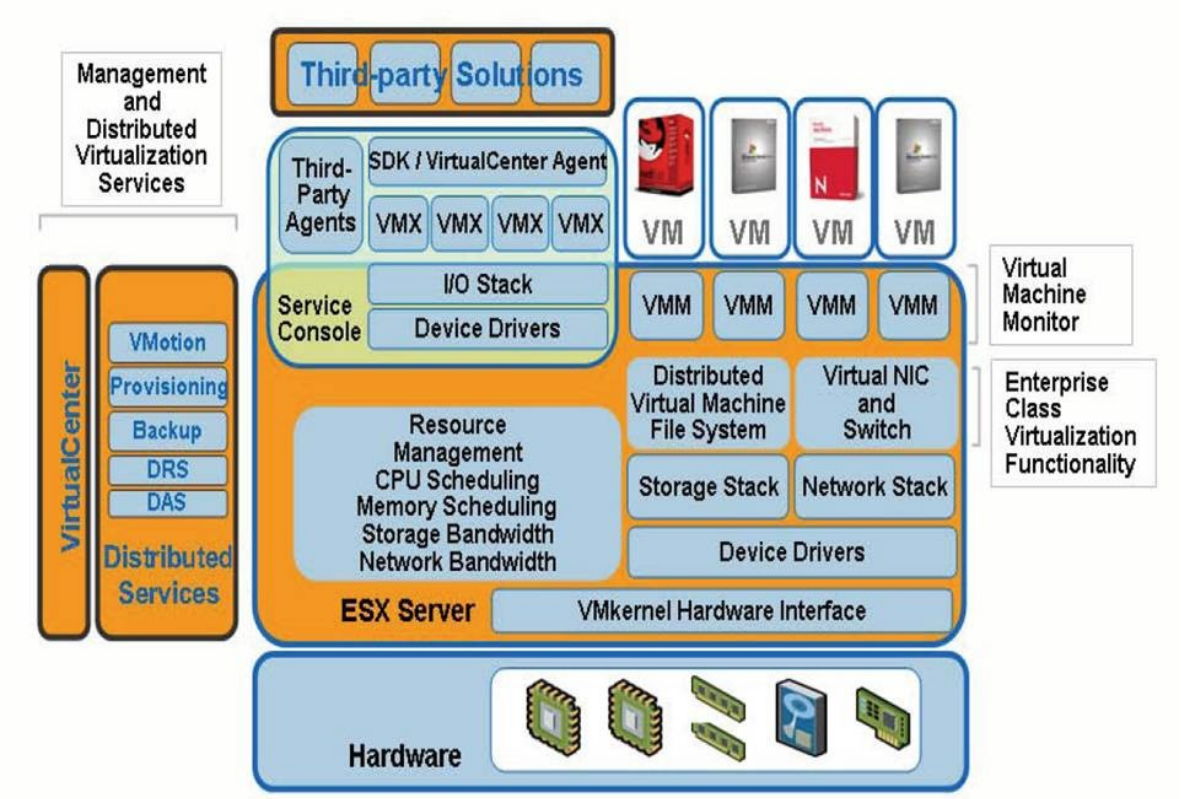
- **When you want information about VMM (or its calls to VMK/VMX) and its overheads**
- **Runs almost as fast as a release build, and collects a wealth of performance data**
- **Stats – Counts of how often important code is reached**
  - Run $VMTREE/support/scripts/getStats.pl
- **Kstats**
  - VMM service times/counts
  - VMM Semaphore/Lock Stats
  - MX User Lock Stats
  - Crosscall Stats
- **Callstack - https://wiki.eng.vmware.com/CallstackProfiling**
  - VM CPU time in great detail
- **Think of the monitor as a collection of services**
  - Exit HV, BT, or DE to perform a service (usually emulation) then back asap
- **Most kstats are instrumentation-based**
  - KSTATS_START – entering VMM to start a new service
  - KSTATS_PUSH/POP – stop previous kstat, start a new one
  - KSTATS_VECTOR – retroactively change the current kstat
  - Fast: no rdtsc on each push/pop to measure time
    - Time in a kstat is sample-based, 100/sec

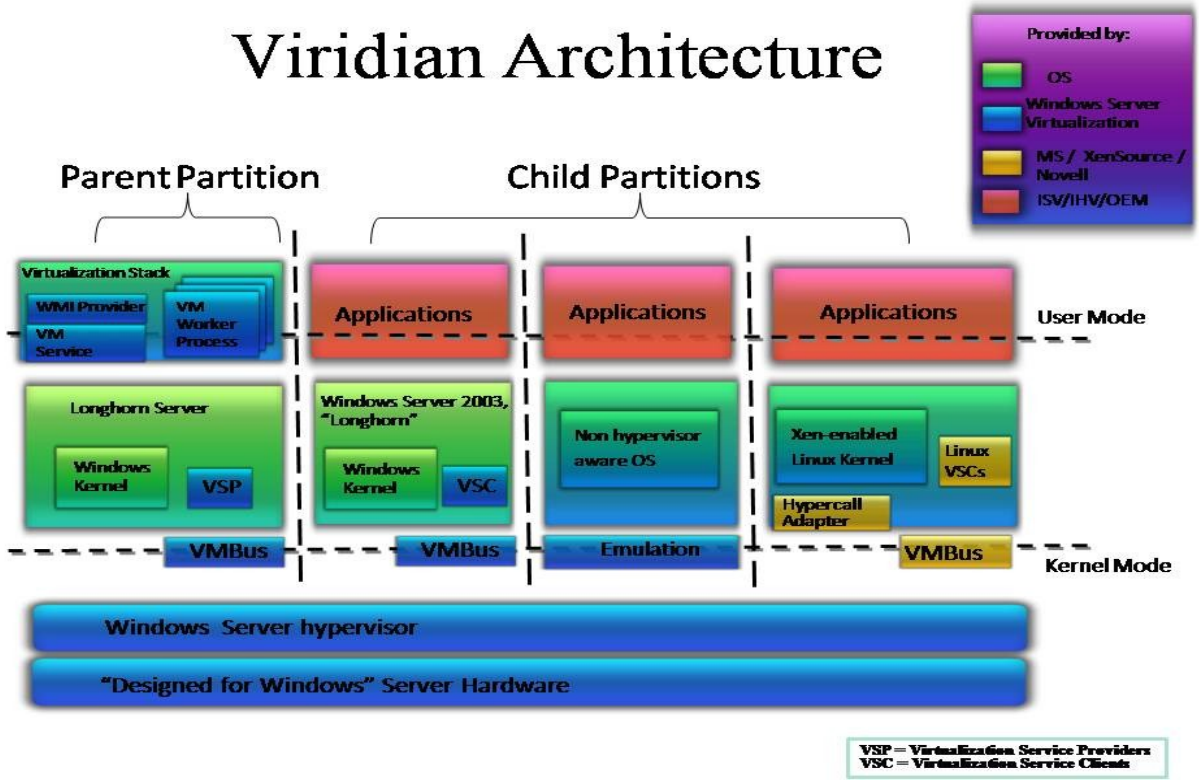
        - Instrumentation only needs to set current kstat and count # of invocation
- **Other kstats are sampled-only**
    - BT, DE, HV – where even modest push/pop is too expensive
    - Just profile such kstats to get time (sacrifice getting # of invocations)
- **Howto: after a run, cd to "stats" subdirectory**
    - $VMTREE/support/scripts/kstats.prl
- **Includes UserRPCs to VMX, VMMon ioctls, VMKernel calls**
- **Includes all of a VM's elapsed time (not just CPU time)**
    - e.g. Including I/O wait, blocked on locks, vmkernel scheduler delays
- **We see elapsed time in vmm & vmkernel, but…**
    - Is it CPU time, wait time, host I/O time, blocked on a lock, …?
    - Kstats doesn't "drill down" (by design, kept simple)
- **We have more in our quiver**
    - kstats.prl prints more things:
        - VMM SemaphoreStats – cpu + blocking time in all vmm locks/semaphores
        - Crosscall stats (contact: kevinc)
        - User-level MX Lock Stats (contact: mbellon)
    - Callstack
    - Vprobes (extensible custom instrumentation – contact: vprobes@vmware.com)

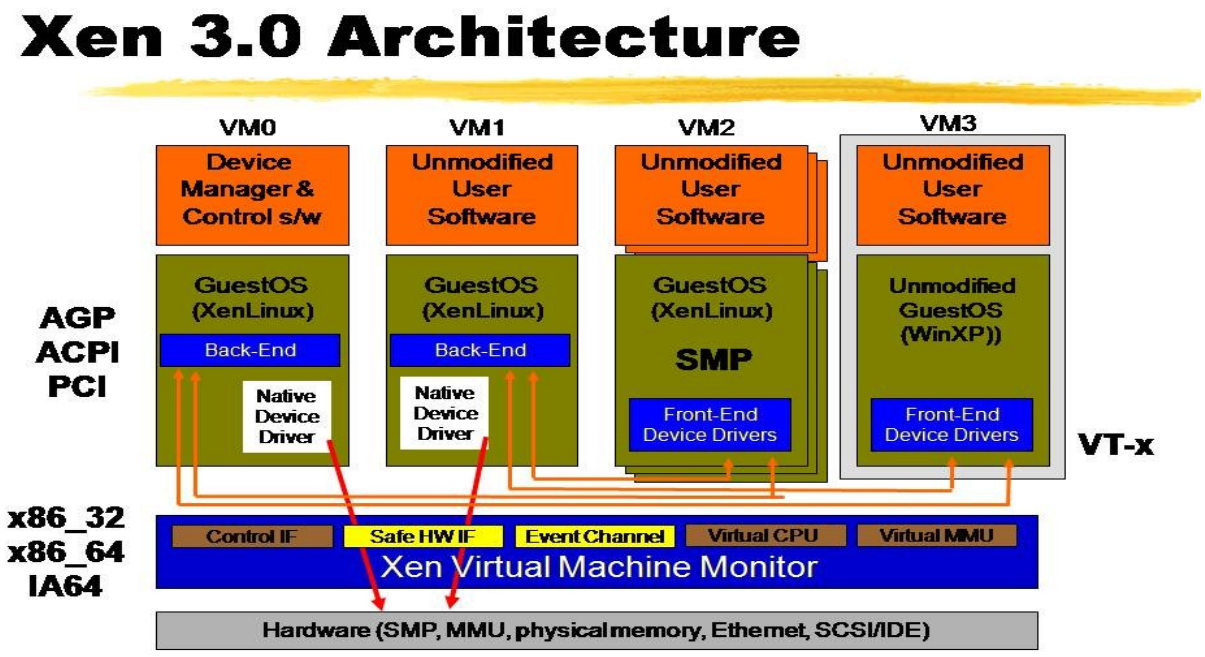
Reference:

https://wiki.eng.vmware.com/HowToGetKSTATSFromVMSupportPerformanceSnapshots

9 Other Virtual Machinesf







Appendix:

CPU stats:

VC counter name	Bug #	Stats Level	Stats Type	Aggregate (vCPUs for VM, pCPUs for host) or per-device?	Unit	Rollup Type	esxtop name	Label	V	H	H(V)	R	C	C(H)	C(R)	Entity	Description	Notes	VSI/vmkctl info
latency (NEW)		1	absolute	aggregate	%	average	%LAT_C	CPU Latency	N		DH	DVA				DVA	Percent of time the VM is unable to run because it is contending for access to the physical CPU(s)		VSI node: /sched/groups/<group id>/stats/cpuStatsDir/cpuStats:latencyStats.cpu

usage		1 (4)	rate	aggregate	%	average (min) (max) (none)	%USED	CPU Usage	X	X		X			X	CPU usage as a percentage during the interval.		
usage		3	''	per-cpu	''	average		''			X							
usagemhz		1 (4)	rate	aggregate	MHz	average (min) (max) (none)		CPU Usage in MHz	X	X		X			X	The amount of CPU used, as measured in megahertz, during the interval.		
usagemhz		3	''	per-cpu	''	average		''	X									
entitlement (NEW)		3	absolute	aggregate	MHz	latest	EMIN	CPU Entitlement	N			DV				CPU resources devoted by the ESX scheduler to the virtual machines and resource pools	ESX determines a VM's cpu entitlement by considering how much CPU a VM wants to use, plus its reservation, limit and shares.	VSI node: /sched/groups/<group id>/stats/cpuStatsDir/cpuStats:effectiveMin
cpuentitlement		1	absolute	aggregate	MHz	latest		CPU Worst case allocation	X			X					Is emin (in vmkernel terminology).	
demand (NEW)		3	rate	aggregate	MHz	average	%DMD	CPU Demand	N		DH	DV			DV	The amount of CPU resources a VM would use if there were no CPU contention or CPU limit		VSI node: /sched/groups/<group id>/stats/cpuStatsDir/cpuLoadHistory/cpuLoadHistory1MinInMhz:avgActive

ready		1	delta	aggregate	milliseconds	summation	%RDY	CPU Ready	X		DH	DV			DV	[NEW] Time the VM is ready to run, but there are no physical CPUs available	This is the basic CPU latency metric. It indicates CPU contention, but not necessarily poor performance.	
ready		3	"	per-vCPU	"	"		"	X							"	"	
costop (NEW)		2	delta	aggregate	milliseconds	summation	%CSTP	CPU Co-stop	DH		DH					Time the VM is ready to run, but is unable to due to co-scheduling constraints	This is different from CPU ready in that if this VM had less VCPUs it could run, but since it has as many as it does, it cannot. This may indicate either CPU overcommitment, that the VM owner should consider using less VCPUs, if possible or possibly memory overcommitment.	Aggregate over all per-vCPU times for the VM.
costop (NEW)		3	"	per-vCPU	"	"		"	N							"	"	VSI node: /sched/vcpus/<world id>/stats/stateTimes:coStopTime
maxlimited (NEW)		2	delta	aggregate	milliseconds	summation	%MLMTD	CPU Max limited	DH			DV			DV	Time the VM is ready to run, but is not run due to maxing out its CPU limit	This may indicate a VM's CPU limit setting should be	Aggregate over all per-vCPU times for the VM.

															setting	increased. However, assuming the admin who set the limit knew what they were doing, this may be expected behavior.	
maxlimited (NEW)		3	"	per-vCPU	"	"		"	N						"	"	VSI node: sched/Vcpus/<world id>/stats/stateTimes:maxLimitedTime
used		3	delta	aggregate	milliseconds	summation	%USED	CPU Used	X		DH	DV			DV	[NEW] Time accounted to the VM. If a system service runs on behalf of this VM, the time spent by that service (i.e. cpu.system) should be charged to this VM. If not, the time spent (i.e. cpu.overlap) should not be charged against this VM.	"used" = "run" + "system" - "overlap"	
used		3	"	per-vCPU	"	"		"	X		DH					"	"	
system		3	delta	aggregate	milliseconds	summation	%SYS	CPU System	X							[NEW] Time spent by system services on behalf of the VM.	The possible system services are interrupt handlers, bottom halves, and system worlds.	

system		3	"	per-vCPU	"	"		"	X						"	"	
overlap (NEW)		3	delta	aggregate	milliseconds	summation	%OVRLP	CPU Overlap	DH						Time the VM was interrupted to perform system services on behalf of that VM or other VMs.		Aggregate over all per-vCPU times for the VM.
overlap (NEW)		3	"	per-vCPU	"	"		"	N						"	"	VSI node: sched/Vcpus/<world id>/stats/stateTimes:sysOverlapTime
run (NEW)		3	delta	aggregate	milliseconds	summation	%RUN	CPU Run	DH						Time the VM is scheduled to run.	100% = "run" + "ready" + "co-stop" + "wait"	Aggregate over all per-vCPU times for the VM.
run (NEW)		3	"	per-vCPU	"	"		"	N						"	"	VSI node: sched/Vcpus/<world id>/stats/stateTimes:runTime
wait		3	delta	aggregate	milliseconds	summation	%WAIT	CPU Wait	X		DH	DV		DV	[NEW] CPU time spent in wait state. CPU Wait includes CPU Idle, CPU Swap Wait and CPU I/O Wait.	"idle", "swap wait" and "io wait" are included in "wait".	
wait		3	"	per-vCPU	"	"		"	X						"	"	

idle		2	delta	aggregate	milliseconds	summation	%IDLE	CPU Idle	X		DH	DV			DV	Total time that the CPU spent in an idle state (meaning that a virtual machine is not runnable). This counter represents the variance, in milliseconds, during theinterval.		
idle		3	"	per-vCPU	"	"		"	X		X					"	"	
swapwait		3	delta	aggregate	milliseconds	summation	%SWPWT	CPU Swap wait	X		DH	DV			DV	Time the VM is waiting for swap page-ins. CPU Swap Wait is included in CPU Wait.	This indicates memory overcommitment causing performance issues. The intent is for this to be used as part of the memory latency metric.	
swapwait		3	"	per-vCPU	"	"		"	X							"	"	
reservedCapacity		2	absolute	N/A	MHz	average		CPU Reserved capacity		X				X		Total CPU capacity reserved by the virtual machines		
totalmhz		1	rate	aggregate of all pCPUs in cluester	MHz	average		CPU Total					X			Aggregate available CPU resources of all the hosts within a cluster	This counter is identical to clusterServices.effectivecpu, I believe.	

~~guest.cpuRunQueueLength (NEW)~~		2	delta	aggregate	number	summation		Guest CPU Run Queue Length	N						Amount of guest processes ready to run in the guest operating system's run queue	This metric measures intra-VM CPU contention.	XXX: Will be available through tools->vmx->hostd channel.
managementAgent.cpuUsage (NEW)		3	rate	aggregate	%	average		Management agent CPU usage		N					Amount of Service Console CPU usage		tail -1 | awk '{print $13}' (returns a percent)

VM GuestOS hang

Problem	Approach	Windows	Linux
Guest Hang	1. Possible cause: VMX, VMM, slow, guest problem 2. Collect facts: does MKS work? can ping guest? esxtop outputs 3. If ping guest works, go to 6 4. Turn vmsample on and collect vmware.log file. If vmsample changes, go to 6 5. VMSample shows guets is not making progress (repeating same instruction), ask for vmm core 6. Guest is alive, ask for VMSS		
Guest BSOD/Panic	1. Ask for VMSS 2. Collect guest logs 3. Debug dump file	event logs windbg	var logs gdb/crash

- **VMM Debug Tools**

VMM Debug Tools

Name	Description	Source	Category
kstats	VM stats for pre-esx50 releases	Built in for obj/opt builds	VM Stats
vmx*3	Ability to switch to VM stats collection on release	esx50 and later	VM stats

	builds.		
vmsamples	Enable them at power-on time with "monitor_control.enable_vmsample = 1" in .vmx, or at run-time with `vmdumper -l`; `vmdumper <wid> samples_on`. They will appear in vmware.log for each VCPU.	Active since which esx release?	VM samples
vm-support -X	Used to examine state of hung VM. Convert vmss to core and load it up using debugger(windbg, crash) to examine state.		Analyze VM state
vmss2core	Convert vmss to core file. Core is loaded in native guest debuggers.	Always build from vmcore-main to get latest guest support. bora> make vmss2core	Core file converter
worldbacktracer.py	Analyze intermittent hangs on the host.	~fjacobs/scripts/worldbacktracer.py	Hang analysis
gdbWrapper.pl	GDB wrapper for VM debugging.	$VMTREE/vmcore/support/debug/gdbWrapper.pl	VMM debugger
gdb-macros	GDB macros for corequery	$VMTREE/bora/vmcore/support/debug	GDB macros
corequery	Query monitor related info.	$VMTREE/bora/vmcore/support/debug/corequery	Core analysis
vprobes	Probes for monitor	$VMTREE/bora/vmcore/support/vprobes/cookbook/vm	Dynamic VM probing

• Computer science or relative major

hands-on experience on Linux platform, system configuration, system admin, scripting etc
• Familiar with Linux Kernel or other Unix system kernel.

• layer 2 and layer 3 network protocols, TCPIP stack.
• C programming, POSIX/UNIX systems programming

• good communication (Chinese and/or English)

Responsibilities
• Develop a VM to attack the ESX host's network. Help to find the security vulnerabilities of the host and/or the virtual network system.

• Design and develop security attacking automation scripts/applications.

• Analyse the exploit and attacking tools from public security forum.

4237236:ISMFM305:5041428:vmx-mks:ISMFM305	4237236:ISMFM305:5041429:vmx-vcpu-0:ISMFM305
0.000	0.002
0.000	72.115
0.000	0.000
0.007	0.356
0.246	1.201
0.334	19.114
0.000	0.000
0.000	23.316
0.278	40.202
99.108	7.585
99.108	79.700

4237236:ISMFM305:5041430:vmx-vcpu-1:ISMFM305	4237236:ISMFM305:5041431:vmx-vcpu-2:ISMFM305
0.002	0.001
86.784	12.090
0.000	0.000
0.160	0.474
0.520	0.343
12.176	84.352
0.000	0.000
0.004	0.095
10.786	81.980
0.535	3.231
87.319	15.321

1. Monitor/VMX study and summary. VMX archtichure, hotplug, crash/hang, performance
2. Fix more VMX crash/hang/performance Monitor issue. Need search from guru queue.
3. GSS presentation, vmx performance document.
4. Coresummary intern project
5. Auto crossport project
6. Patent
7. RADIO
8. India GSS rotation
9. FreeBSD book writing
10. Child.
11. Family.
12. SDN/Openstack projects.
13. VForum, Innovation.