宁夏培训内容的总结

1. 链接的总结
    a. 系统下载

        正常系统下载使用官方网站

            Ubuntu: https://ubuntu.com/download/server

            CentOS: https://www.centos.org/download/

            RockyLinux: https://rockylinux.org/download

        特殊的系统下载使用google搜索页可以找到对应的下载页面。

            例如，会议上提到的ubuntu22.04.4，可以使用下面这个链接
            https://old-releases.ubuntu.com/releases/22.04.4/

            这个链接也是通过google搜索得来。

    b. IB相关驱动

        - 各种工具及固件匹配关系的最新版本查询的链接

          https://www.nvidia.com/en-us/networking/products/lts-releases/

          https://developer.nvidia.com/networking/infiniband-software

        - IB网卡驱动及对应的工具包

          IB网卡的驱动可以通过关键词搜索 `OFED download` ，链接如下

          https://network.nvidia.com/products/infiniband-drivers/linux/mlnx_ofed/
          选择LTS（长期支持）版本和对应的系统和架构即可。下载时选择ISO或者tar

          未来IB网卡会从OFED过渡到DOCA，链接如下

          https://developer.nvidia.com/doca-downloads

        - MFT (Mellanox Firmware Toolkit)

          https://network.nvidia.com/products/adapter-software/firmware-tools/

        - 交换机固件及网卡固件

          https://network.nvidia.com/support/firmware/firmware-downloads/

        - CPLD的说明文档

          https://docs.nvidia.com/networking/display/mftv4301508/cpldupdate+–+tool+for+programming+on-board+cplds+on+nvidia+devices

    c. GPU相关驱动以及CUDA包

        - CUDA工具

          https://developer.nvidia.com/cuda-toolkit-archive

          选择对应版本，选择runfile。如何安装下面会有对应的介绍。

        - GPU驱动

          https://www.nvidia.cn/drivers/lookup/

        - nvlink驱动

          https://developer.download.nvidia.cn/compute/cuda/repos/ubuntu2204/x86_64/

          找到cuda对应的fabricmanager的驱动版本。配套版本查找方法如下
          https://docs.nvidia.com/cuda/cuda-toolkit-release-notes/index.html#id9

        - container（docker）的支持包

          https://developer.download.nvidia.cn/compute/cuda/repos/ubuntu2204/x86_64/

          nvidia-container-toolkit_1.17.8-1_amd64.deb

          nvidia-container-toolkit-base_1.17.8-1_amd64.deb

libnvidia-container1_1.17.8-1_amd64.deb

libnvidia-container-tools_1.17.8-1_amd64.deb

安装方法：`dpkg -i *.deb`

- 关于apt源，也可以使用cuda的源来进行安装cuda

```
wget https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2404/x86_64/cuda-keyring_1.1-1_all.deb
sudo dpkg -i cuda-keyring_1.1-1_all.deb
sudo apt-get update
sudo apt-get -y install cuda-toolkit-13-0
```

d. apt源，一般使用清华或者阿里云

清华源举例

```
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy main restricted
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-updates main restricted
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy universe
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-updates universe
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-updates multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-backports main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-security main restricted
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-security universe
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-security multiverse
```

阿里源举例

```
deb https://mirrors.aliyun.com/ubuntu/ jammy main restricted universe multiverse
deb-src https://mirrors.aliyun.com/ubuntu/ jammy main restricted universe multiverse
deb https://mirrors.aliyun.com/ubuntu/ jammy-security main restricted universe multiverse
deb-src https://mirrors.aliyun.com/ubuntu/ jammy-security main restricted universe multiverse
deb https://mirrors.aliyun.com/ubuntu/ jammy-updates main restricted universe multiverse
deb-src https://mirrors.aliyun.com/ubuntu/ jammy-updates main restricted universe multiverse
deb-src https://mirrors.aliyun.com/ubuntu/ jammy-backports main restricted universe multiverse
```

2. 操作系统安装步骤的总结

a. 安装操作系统，一些基础软件的推荐。

```
apt install -y openssh-server curl wget vim htop tree ansible ipmitool lldpd net-tools unzip
```

```
- openssh-server
- curl
- wget
- vim
- htop
- tree
- ansible
- ipmitool
- lldpd
- net-tools
- unzip
```

b. 更新内核（视需求更新）

```
# 查询内核
root@UFM-StorNet-Node01:/# cat /proc/version
Linux version 5.15.0-151-generic (buildd@lcy02-amd64-092) (gcc (Ubuntu 11.4.0-1ubuntu1~22.04) 11.4.0, GNU ld (GNU Binutils for Ubuntu) 2.38) #161-Ubuntu SMP Tue Jul 22 14:25:40 UTC 2025
root@UFM-StorNet-Node01:/# uname -r
5.15.0-151-generic
root@UFM-StorNet-Node01:/# uname -a
Linux UFM-StorNet-Node01 5.15.0-151-generic #161-Ubuntu SMP Tue Jul 22 14:25:40 UTC 2025 ×86_64 ×86_64 ×86_64 GNU/Linux

# 查看可用内核版本
apt search linux-image

# 安装特定内核版本
sudo apt install linux-image-[版本号]
sudo apt install linux-headers-[版本号]

# 更新引导加载器，更改下次启动的加载的内核
sudo update-grub
```

c. 关闭系统自动更新

```
sed -i.bak 's/1/0/' /etc/apt/apt.conf.d/10periodic

vi /etc/apt/apt.conf.d/50unattended-upgrades
加入
Unattended-Upgrade::Package-Blacklist {
    "linux-image-*";
    "linux-headers-*";
};
```

d. 设置BIOS

需要设置的项目

1. iommu / vt-d
2. cpu改为性能模式
3. 关闭 ACS

查询的命令

```
dmesg | grep -i iommu
cpupower frequency-info
lspci -vvv | grep ACSCtl
```

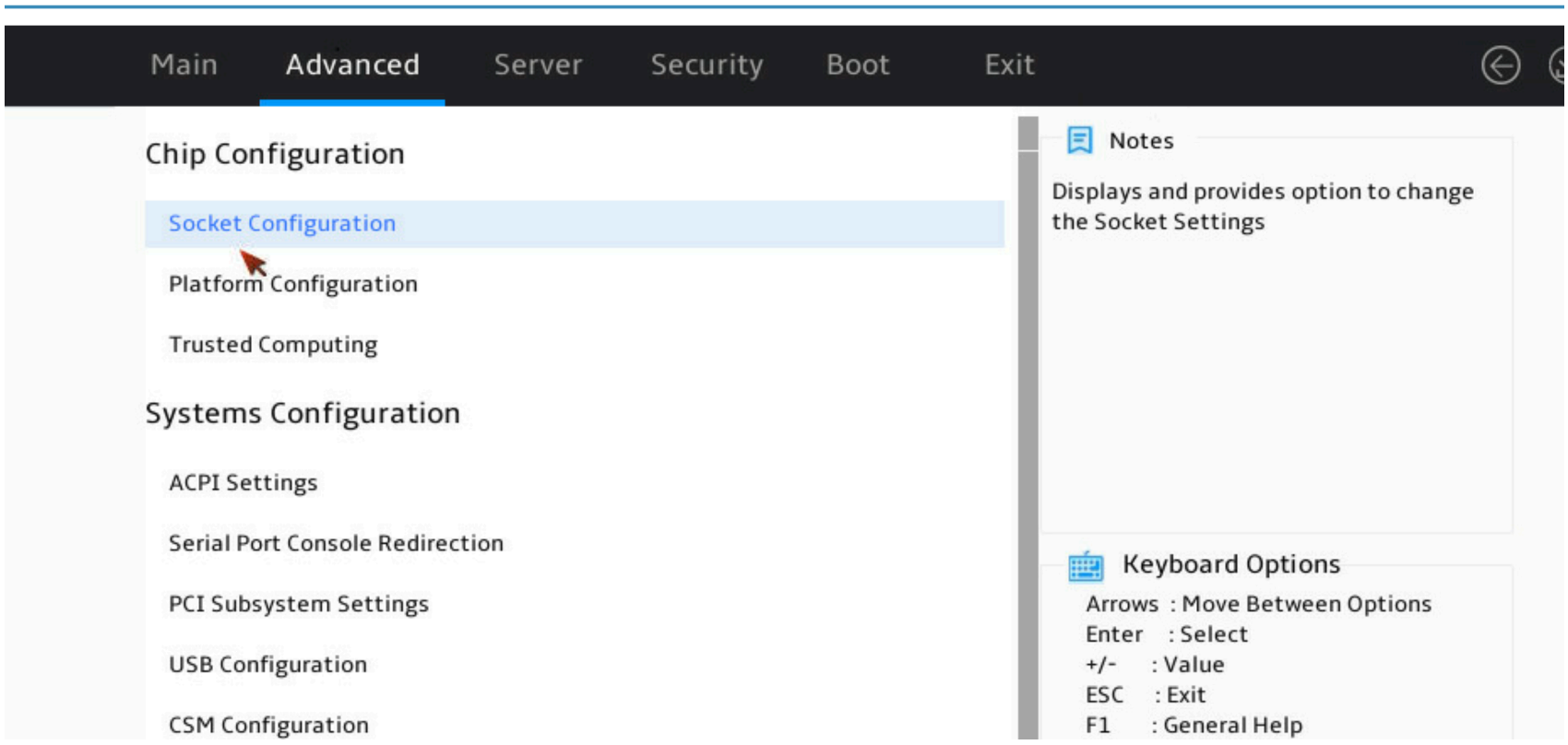
查询结果

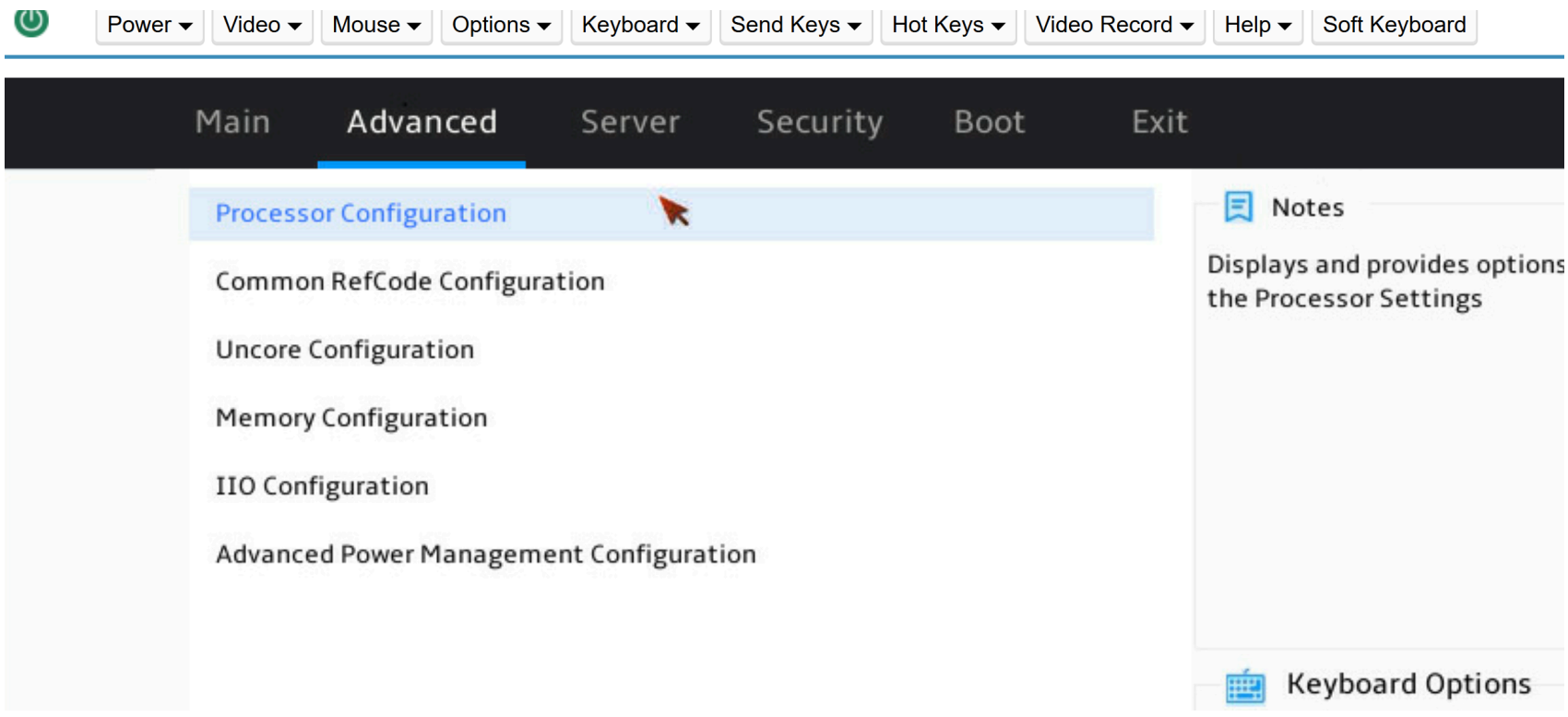
```
root@H800GPU-ComServer-Node09:~# dmesg | grep -i iommu
[   11.287745] DMAR-IR: IOAPIC id 8 under DRHD base  0x9c3fc000 IOMMU 19
[   16.839506] iommu: Default domain type: Translated
[   16.839506] iommu: DMA domain TLB invalidation policy: lazy mode
```

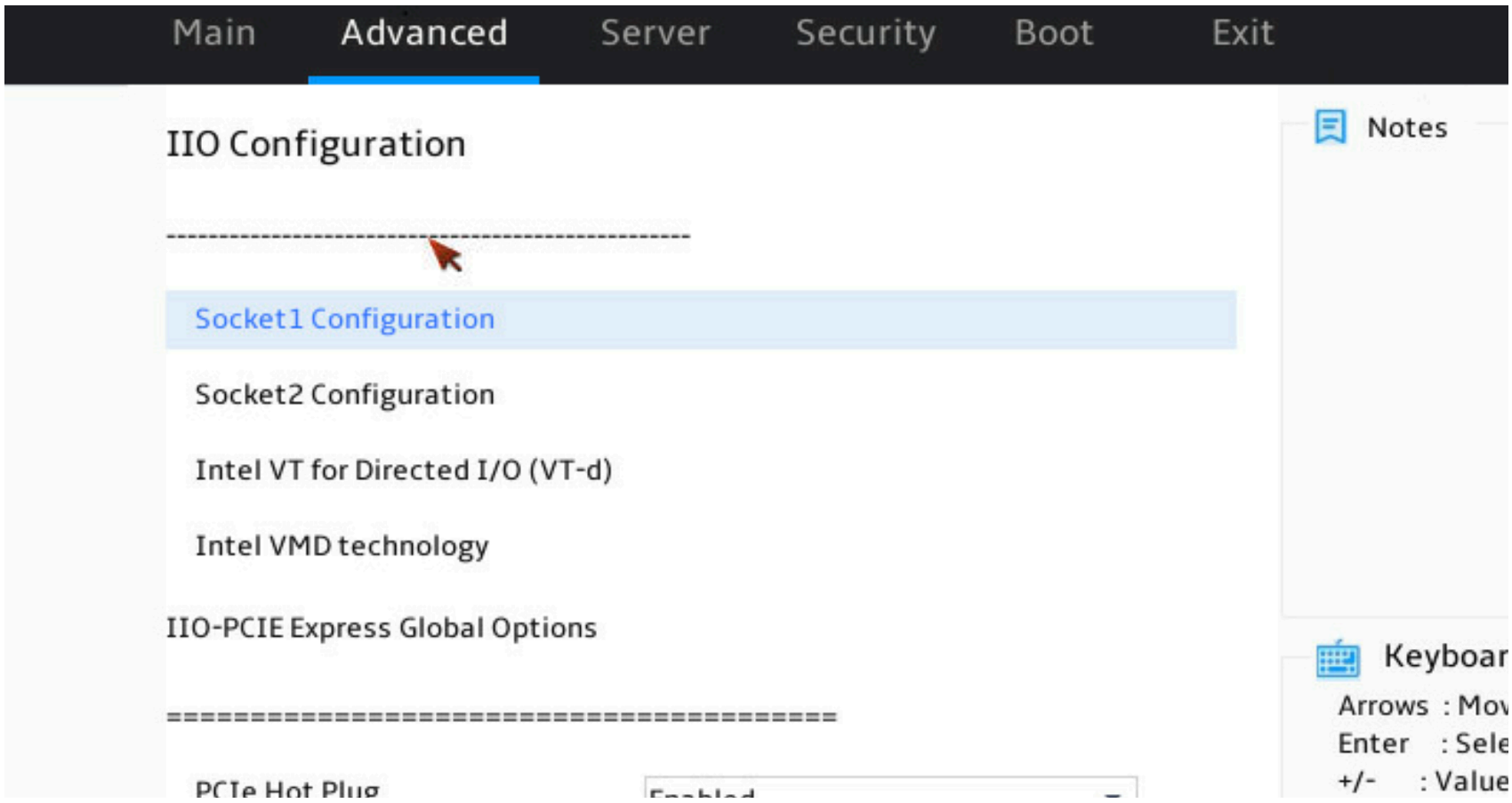
```
root@H800GPU-ComServer-Node09:~# cpupower frequency-info
analyzing CPU 0:
  driver: intel_pstate
  CPUs which run at the same hardware frequency: 0
  CPUs which need to have their frequency coordinated by software: 0
```

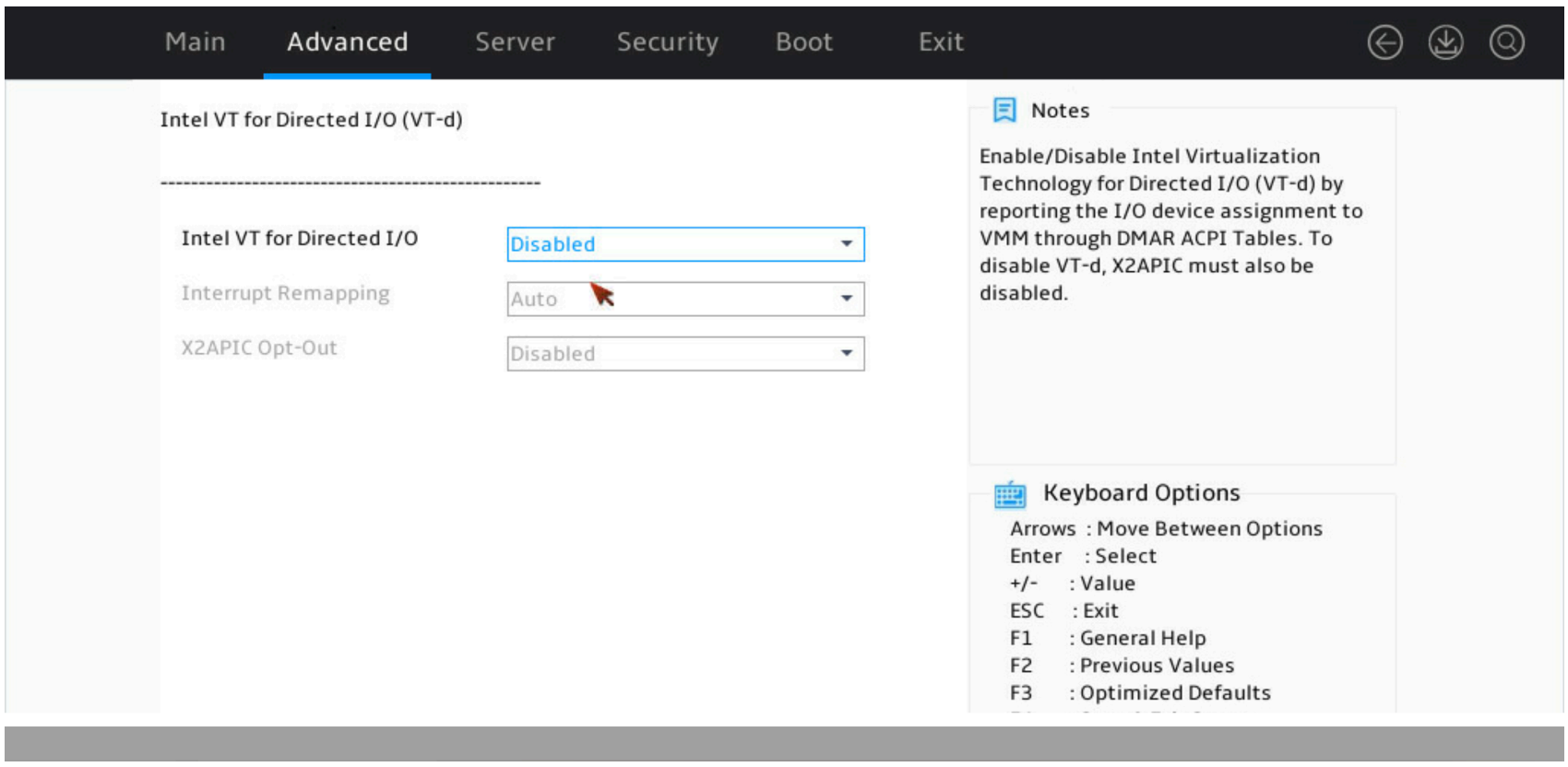
```
maximum transition latency:  Cannot determine or is not supported.
hardware limits: 800 MHz - 3.40 GHz
available cpufreq governors: performance powersave
current policy: frequency should be within 800 MHz and 3.40 GHz.
                The governor "powersave" may decide which speed to use
                within this range.
current CPU frequency: Unable to call hardware
current CPU frequency: 768 MHz (asserted by call to kernel)
boost state support:
  Supported: yes
  Active: yes
```

▼ 关闭vt-d

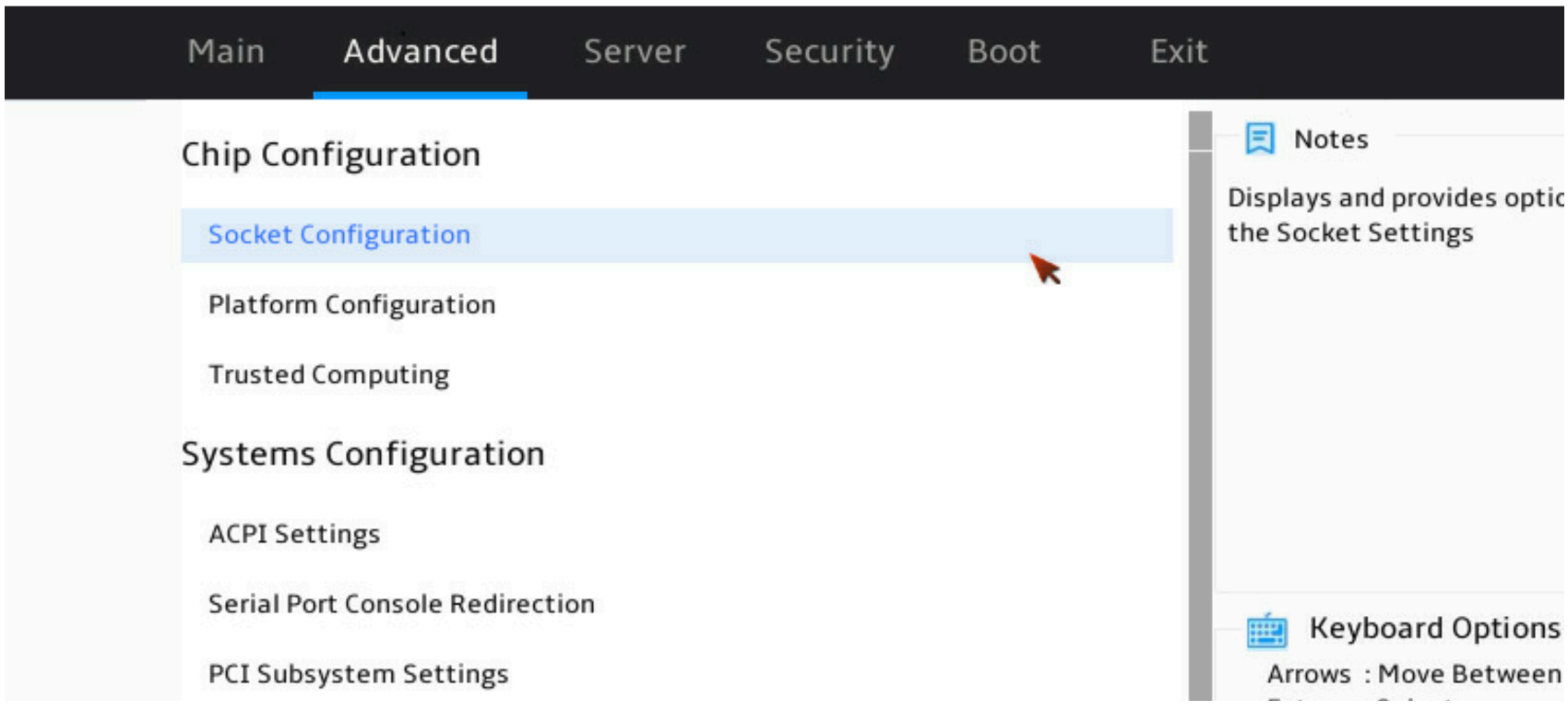


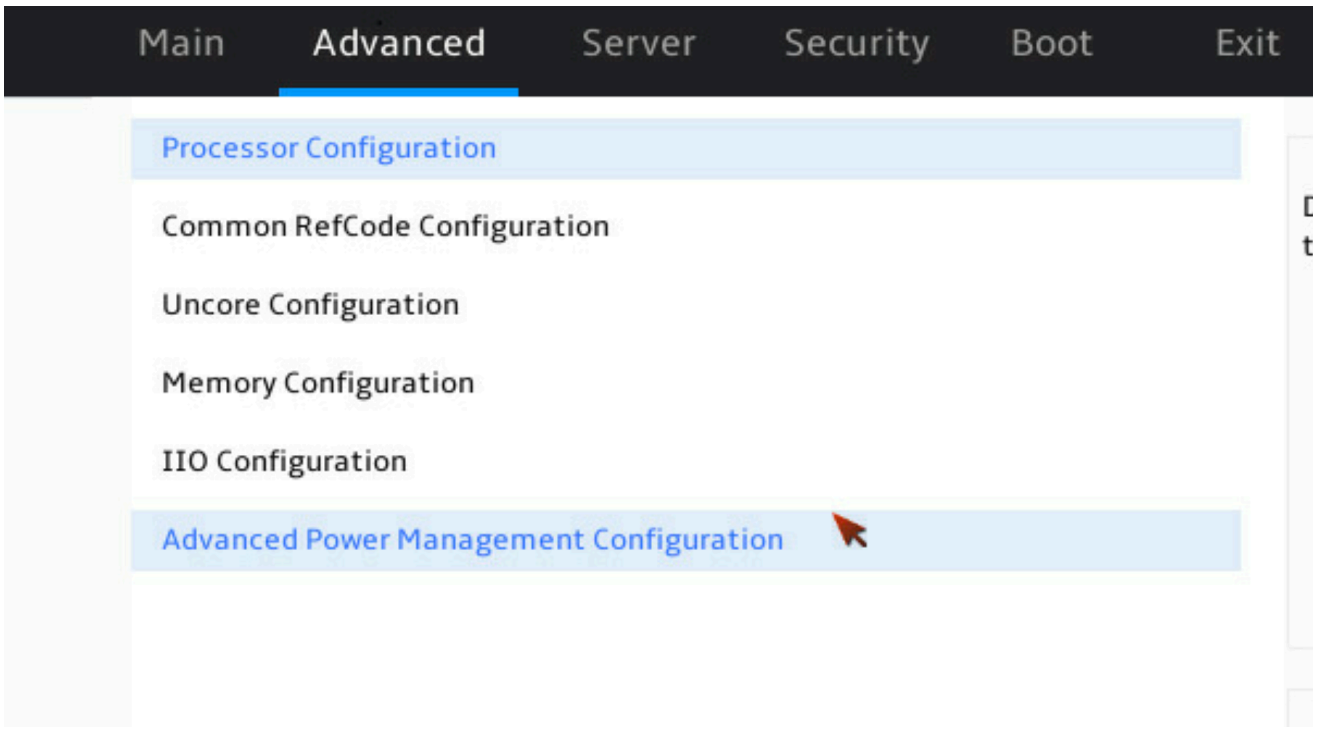


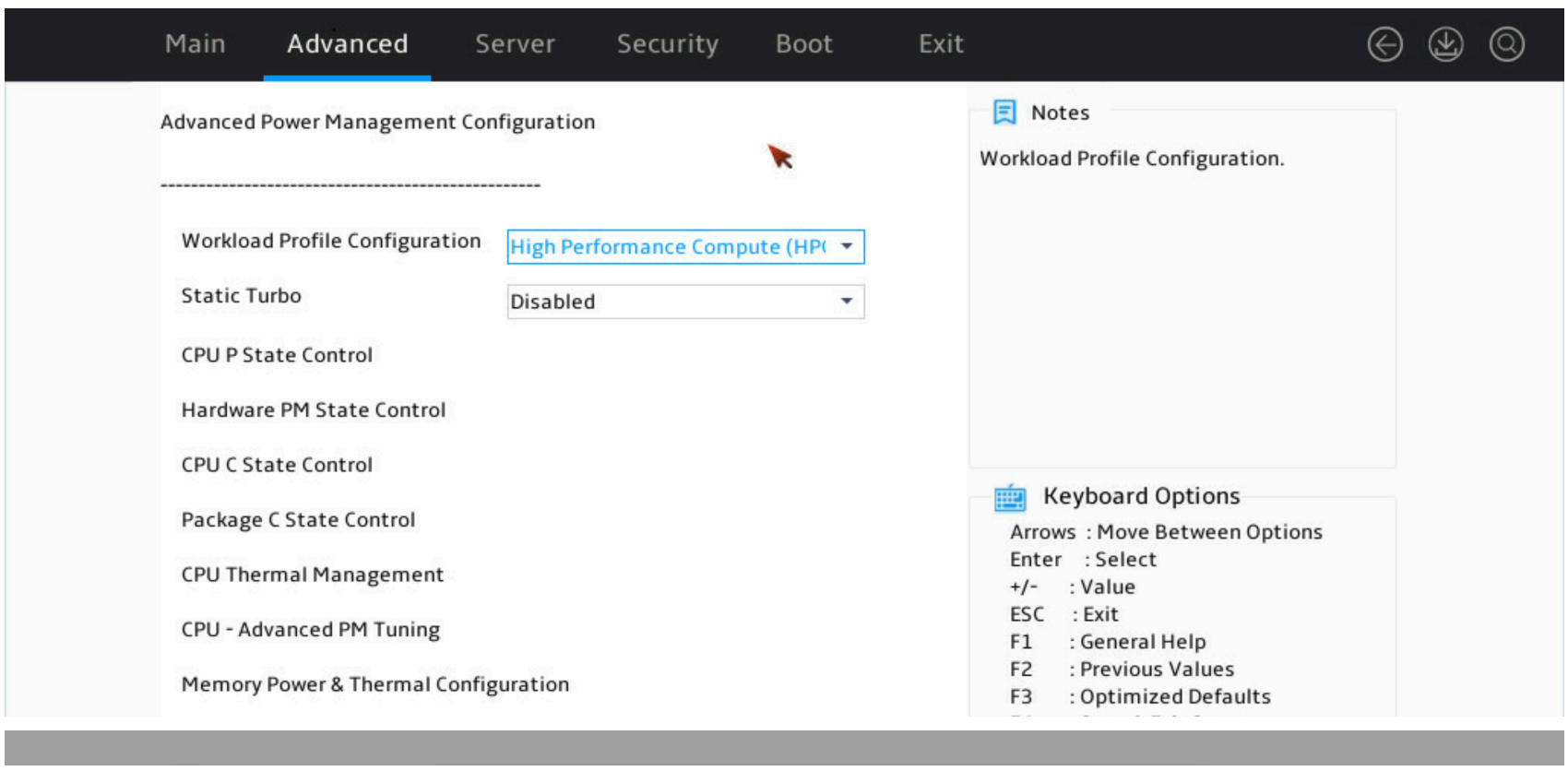

Main
Advanced
Server
Security
Boot
Exit
IIO Configuration
Socket1 Configuration
Socket2 Configuration
Intel VT for Directed I/O (VT-d)
Intel VMD technology
IIO-PCIE Express Global Options
Notes
Keyboar
Arrows : Mov
Enter : Sele
+/- : Value


Main
Advanced
Server
Security
Boot
Exit
Intel VT for Directed I/O (VT-d)
Intel VT for Directed I/O
Disabled
Interrupt Remapping
Auto
X2APIC Opt-Out
Disabled
Notes
Enable/Disable Intel Virtualization Technology for Directed I/O (VT-d) by reporting the I/O device assignment to VMM through DMAR ACPI Tables. To disable VT-d, X2APIC must also be disabled.
Keyboard Options
Arrows : Move Between Options
Enter : Select
+/- : Value
ESC : Exit
F1 : General Help
F2 : Previous Values
F3 : Optimized Defaults

▼ 设置CPU的性能模式


Main
Advanced
Server
Security
Boot
Exit
Chip Configuration
Socket Configuration
Platform Configuration
Trusted Computing
Systems Configuration
ACPI Settings
Serial Port Console Redirection
PCI Subsystem Settings
Notes
Displays and provides optic
the Socket Settings
Keyboard Options
Arrows : Move Between





3. IB网卡的驱动及固件

   a. 安装网卡驱动OFED（开局安装驱动，包含了固件）

      下载链接见上

      ```
      tar xf MLNX_OFED_LINUX-<version>-<os>-x86_64.tar.gz
      cd MLNX_OFED_LINUX-<version>-<os>
      ./mlnxofedinstall --all --force
      /etc/init.d/openibd restart
      ```

   b. 更换网卡后，单独刷IB网卡固件

      ```
      flint -d mlx5_0 query
      flint -d mlx5_0 -i fw-ConnectX7-rel-28_43_2566-MCX75310AAS-NEA_Ax-UEFI-14.37.13-FlexBoot-3.7.500.signed.bin burn
      ```

4. 升级IB交换机固件（包括FW和CPLD）

   a. 对交换机的操作建议在对应网络（计算、存储）的UFM上进行操作。因为在拥有两个网络的设备上操作的时候，无法指定对应的网卡，从而无法升级对应的网络的交换

   b. 升级固件（只能升级不带管理的，升级带管理的需联系代理商）

      i. 检查交换机固件

         查询交换机lid 列表 `ibswitches`

查询基础命令`flint -d lid-496 query`

查询批量命令

```
ibswitches | awk '{print $11}' | grep -o '[0-9]\+' | xargs -I {} sh -c 'echo "LID: {}" && flint -d lid-{} query | grep "FW Version"'
```

ii. 升级的步骤

```
wget https://www.mellanox.com/downloads/firmware/fw-Quantum-2-rel-31_2016_1028-MQM9790-NS2X_Ax.signed.bin.zip
unzip fw-Quantum-2-rel-31_2016_1028-MQM9790-NS2X_Ax.signed.bin.zip
flint -d lid-496 -i fw-Quantum-2-rel-31_2016_1028-MQM9790-NS2X_Ax.signed.bin burn
```

iii. 重启交换机生效

`flint -d lid-19 swreset`

c. 升级CPLD固件

i. 安装MFT（找一台安装即可）

```
tar xzf mft-4.xx.x-xxx.tgz
cd mft-4.xx.x-xxx
./install.sh --oem
mst start
mst status
```

ii. 检查CPLD的固件（一般查看完以后都需要升级到最新的）

CPLD工具需要向代理商获取

基础命令`./updateswitchcpld --unmanaged -d lid-26 --check_cpld --verbose`

批量命令

```
ibswitches | awk '{print $11}' | grep -o '[0-9]\+' | xargs -I {} sh -c 'echo "LID: {}" &&  ./updateswitchcpld --unmanaged -d lid-{} --check_cpld --verbose'
```

iii. 升级CPLD的固件（永远升级最新版即可，没有配套关系）

```
#不带管理的交换机
./updateswitchcpld --unmanaged -d lid-<lid of switch>
#带管理的交换机
./updateswitchcpld --managed -t <switch's IP> -u <username> -p <password> --os mlnx-os
```

iv. 重启交换机

`flint -d lid-19 swreset`

5. 安装显卡驱动（CUDA）

a. 下载CUDA

```
wget https://developer.download.nvidia.com/compute/cuda/13.0.0/local_installers/cuda_13.0.0_580.65.06_linux.run
```

b. 安装CUDA(gpu的驱动）

i. 先检查是否有默认的（系统自带的）显卡驱动

```
lsmod | grep nouveau
```

如果没有输出证明没有加载这个模块。如果有输出，需要关闭

```
echo "blacklist nouveau " >>/etc/modprobe.d/blacklist.conf
echo "options nouveau modeset=0">>/etc/modprobe.d/blacklist.conf
```

ii. 安装CUDA

```
sh cuda_12.9.1_575.57.08_linux.run --silent --driver --toolkit --override-driver-check
```

iii. 配置环境变量

```
# 添加环境 cuda 变量
# echo "export PATH=/usr/local/cuda-12.x/bin${PATH:+:${PATH}} " >> ~/.bashrc
echo "export PATH=/usr/local/cuda-12.6/bin${PATH:+:${PATH}}" >> ~/.bashrc
# echo "export LD_LIBRARY_PATH=/usr/local/cuda-12.x/lib64${LD_LIBRARY_PATH:+:${LD_LIBRARY_PAT
H}}" >> ~/.bashrc
echo "export LD_LIBRARY_PATH=/usr/local/cuda-12.6/lib64${LD_LIBRARY_PATH:+:${LD_LIBRARY_PATH}}"
>> ~/.bashrc

source ~/.bashrc
# 选择修改对应的cuda版本
# 注：export CUDA_HOME=/usr/local/cuda
```

iv. 持久化

```
cd /usr/share/doc/NVIDIA_GLX-1.0/samples/
# 解压nvidia-persistenced-init.tar.bz2
tar -xvf nvidia-persistenced-init.tar.bz2
# 进入nvidia-persistenced-init文件夹
cd nvidia-persistenced-init/
# 执行
./install.sh #,安装nvidia-persistenced
#启动持久化服务
systemctl start nvidia-persistenced.service
#设置nvidia-persistenced服务开机启动
systemctl enable nvidia-persistenced.service
```

6. 安装nv switch驱动（fabricmanager）

a. 安装驱动之前，查看各个GPU之间的连接状态

▼ 图片如下

```
root@H800GPU-ComServer-Node09:/home/ubuntu/startup# nvidia-smi topo -p2p a
        GPU0    GPU1    GPU2    GPU3    GPU4    GPU5    GPU6    GPU7
 GPU0   X       NS      NS      NS      NS      NS      NS      NS
 GPU1   NS      X       NS      NS      NS      NS      NS      NS
 GPU2   NS      NS      X       NS      NS      NS      NS      NS
 GPU3   NS      NS      NS      X       NS      NS      NS      NS
 GPU4   NS      NS      NS      NS      X       NS      NS      NS
 GPU5   NS      NS      NS      NS      NS      X       NS      NS
 GPU6   NS      NS      NS      NS      NS      NS      X       NS
 GPU7   NS      NS      NS      NS      NS      NS      NS      X

Legend:

  X    = Self
  OK   = Status Ok
  CNS  = Chipset not supported
  GNS  = GPU not supported
  TNS  = Topology not supported
  NS   = Not supported
  U    = Unknown
root@H800GPU-ComServer-Node09:/home/ubuntu/startup#
```

可以看到连接状态是NS

b. 安装

```
wget https://developer.download.nvidia.cn/compute/cuda/repos/ubuntu2204/x86_64/nvidia-fabricmanager-560_560.28.03-1_amd64.deb
sudo dpkg -i nvidia-fabricmanager-560_560.28.03-1_amd64.deb
systemctl start nvidia-fabricmanager
systemctl enable nvidia-fabricmanager
```

c. 配置对应的服务

```
systemctl start nvidia-fabricmanager # 启动 nvidia-fabricmanager 服务
systemctl status nvidia-fabricmanager # 查看 nvidia-fabricmanager 服务
systemctl enable nvidia-fabricmanager # 设置开机自启
```

d. 安装完以后再次观察

▼ 图片如下

```
root@H800GPU-ComServer-Node09:/home/ubuntu/startup# nvidia-smi topo -p2p a
        GPU0    GPU1    GPU2    GPU3    GPU4    GPU5    GPU6    GPU7
 GPU0   X       OK      OK      OK      OK      OK      OK      OK
 GPU1   OK      X       OK      OK      OK      OK      OK      OK
 GPU2   OK      OK      X       OK      OK      OK      OK      OK
 GPU3   OK      OK      OK      X       OK      OK      OK      OK
 GPU4   OK      OK      OK      OK      X       OK      OK      OK
 GPU5   OK      OK      OK      OK      OK      X       OK      OK
 GPU6   OK      OK      OK      OK      OK      OK      X       OK
 GPU7   OK      OK      OK      OK      OK      OK      OK      X

Legend:

  X    = Self
  OK   = Status Ok
  CNS  = Chipset not supported
  GNS  = GPU not supported
  TNS  = Topology not supported
  NS   = Not supported
  U    = Unknown
root@H800GPU-ComServer-Node09:/home/ubuntu/startup#
```

7. 安装docker支持(视需要而定)

```
wget https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2204/x86_64/cuda-keyring_1.1-1_all.deb
sudo dpkg -i cuda-keyring_1.1-1_all.deb
sudo apt-get update
apt install libnvidia-container-tools nvidia-container-toolkit
```

8. 做系统基础配置例如network ntp dns log压缩 配置
9. 多数情况下，可以把相关的配置写成一个ansible的playbook
10. 打流测试包括IB的，以太的
    a. IB打流测试
        i. 带宽
            - 前提条件两台设备要可以相互ping。这个ip不一定必须要配置在IB网卡上面，只需要相互ping通就可以了
            - 一端设备开始被测试（这台设备的IP是`10.232.87.102`）

              `ib_write_bw -d -q 10 mlx5_ 0 --report_gbits`
            - 一端设备开始测试

              `ib_write_bw -d mlx5_0 -q 10 --report_gbits --run_infinitely 10.232.87.102`
        ii. 延时
            - 前提条件两台设备要可以相互ping。这个ip不一定必须要配置在IB网卡上面，只需要相互ping通就可以了
            - 一端设备开始被测试（这台设备的IP是`10.232.87.102`）

              `ib_write_lat -d mlx5_0`

- 一端设备开始测试

  `ib_write_lat -d mlx5_0  10.232.87.102`

- 关于结果，主要参考绿色部分的三个值

```
root@UFM-StorNet-Node01:~# ib_write_lat -d mlx5_0  10.232.87.102
---------------------------------------------------------------------------------------
                    RDMA_Write Latency Test
 Dual-port       : OFF          Device         : mlx5_0
 Number of qps   : 1            Transport type : IB
 Connection type : RC           Using SRQ      : OFF
 PCIe relax order: OFF          Lock-free      : OFF
 ibv_wr* API     : ON
 TX depth        : 1
 Mtu             : 4096[B]
 Link type       : IB
 Max inline data : 220[B]
 rdma_cm QPs     : OFF
 Data ex. method : Ethernet
---------------------------------------------------------------------------------------
 local address: LID 0x01 QPN 0x002f PSN 0x5c5957 RKey 0x1ffcbb VAddr 0x0056000d5ac000
 remote address: LID 0x13d QPN 0x002f PSN 0x6ac2da RKey 0x1ff5b3 VAddr 0x00555bbb6cf000
---------------------------------------------------------------------------------------
 #bytes #iterations    t_min[usec]    t_max[usec]  t_typical[usec]    t_avg[usec]    t_stdev[usec]   99% per
centile[usec]   99.9% percentile[usec]
Conflicting CPU frequency values detected: 2100.000000 != 1957.948000. CPU Frequency is not max.
 2       1000         1.60         3.88         1.84                1.84           0.03          1.89                3.88
---------------------------------------------------------------------------------------
```

b. 以太打流测试

i. 带宽

- 前提条件两台设备要可以相互ping。这个ip不一定必须要配置在IB网卡上面，只需要相互ping通就可以了
- 一端设备开始被测试（这台设备的IP是`10.232.87.102`）

  `iperf3 -s`

- 一端设备开始测试

  `iperf3 -c 10.232.87.102`

- 注意

  lacp模式下，不加参数只能测得一根链路的带宽，如果想要测试bond链路的带宽，需要根据lacp的模式调整测试方案

11. gpu burn测试

a. 下载并编译gpu burn

```
git clone https://github.com/wilicc/gpu-burn.git
cd gpu-burn
make
```

b. 指定使用那张卡进行测试

`CUDA_VISIBLE_DEVICES=0,1,2,3,4,5,6,7 ./gpu_burn 300`

c. 查看GPU的状态

`watch -n 1 nvidia-smi`

d. 结果

12. nccl 安装及测试
    a. 先安装mpirun

```
apt-get install openmpi-bin openmpi-common libopenmpi-dev
```

    b. nccl

    下载和编译

```
git clone https://github.com/NVIDIA/nccl.git
cd nccl
make -j src.build
```

    安装

```
# Install tools to create debian packages
sudo apt install build-essential devscripts debhelper fakeroot
# Build NCCL deb package
make pkg.debian.build
# 还在nccl那个工作目录下
ls build/pkg/deb/
# 还在nccl那个工作目录下
dpkg -i build/pkg/deb/*.deb
```

    c. nccl_test

    下载并编译

```
git clone https://github.com/NVIDIA/nccl-tests.git
# 这里如果如果CUDA没有安装到/usr/local/cuda需要执行CUDA的安装目录
# 这里如果没有指定nccl的安装命令，需要指定nccl的安装目录
# 如果都是默认安装这不需要指定
# make CUDA_HOME=/path/to/cuda NCCL_HOME=/path/to/nccl
#进入nccl-tests目录
cd nccl-tests
make MPI=1 MPI_HOME=/usr/lib/x86_64-linux-gnu/openmpi
```

    d. 使用nccl_test进行测试
        i. 单点nccl-tests执行

`./build/all_reduce_perf -b 128M -e 8G -f 2 -g 8`

        或者

```
mpirun --allow-run-as-root \
```

```
-np 8 ./build/all_reduce_perf -b 128M -e 8G -f 2
```

结果

```
root@H800GPU-ComServer-Node01:/home/ubuntu/startup/nccl-tests#
root@H800GPU-ComServer-Node01:/home/ubuntu/startup/nccl-tests# mpirun --allow-run-as-root \
-np 8 ./build/all_reduce_perf -b 128M -e 8G -f 2
# Collective test starting: all_reduce_perf
# nThread 1 nGpus 1 minBytes 134217728 maxBytes 8589934592 step: 2(factor) warmup iters: 5 iters: 20 agg iters: 1 validation
: 1 graph: 0
#
# Using devices
#  Rank  0 Group  0 Pid 3928604 on H800GPU-ComServer-Node01 device  0 [0000:19:00] NVIDIA H800
#  Rank  1 Group  0 Pid 3928605 on H800GPU-ComServer-Node01 device  1 [0000:3a:00] NVIDIA H800
#  Rank  2 Group  0 Pid 3928606 on H800GPU-ComServer-Node01 device  2 [0000:4c:00] NVIDIA H800
#  Rank  3 Group  0 Pid 3928607 on H800GPU-ComServer-Node01 device  3 [0000:5c:00] NVIDIA H800
#  Rank  4 Group  0 Pid 3928608 on H800GPU-ComServer-Node01 device  4 [0000:9a:00] NVIDIA H800
#  Rank  5 Group  0 Pid 3928610 on H800GPU-ComServer-Node01 device  5 [0000:ba:00] NVIDIA H800
#  Rank  6 Group  0 Pid 3928612 on H800GPU-ComServer-Node01 device  6 [0000:ca:00] NVIDIA H800
#  Rank  7 Group  0 Pid 3928614 on H800GPU-ComServer-Node01 device  7 [0000:da:00] NVIDIA H800
#
#                                                     out-of-place                       in-place
#       size         count      type   redop    root     time   algbw   busbw #wrong     time   algbw   busbw #wrong
#        (B)    (elements)                               (us)  (GB/s)  (GB/s)            (us)  (GB/s)  (GB/s)
   134217728      33554432     float     sum      -1   1184.7  113.29  198.27      0   1185.6  113.20  198.11      0
   268435456      67108864     float     sum      -1   2282.2  117.62  205.83      0   2281.8  117.64  205.87      0
   536870912     134217728     float     sum      -1   4469.0  120.13  210.23      0   4464.5  120.25  210.44      0
  1073741824     268435456     float     sum      -1   8825.5  121.66  212.91      0   8824.3  121.68  212.94      0
  2147483648     536870912     float     sum      -1    17530  122.50  214.38      0    17525  122.54  214.45      0
  4294967296    1073741824     float     sum      -1    34979  122.79  214.88      0    34972  122.81  214.92      0
  8589934592    2147483648     float     sum      -1    69877  122.93  215.13      0    69846  122.98  215.22      0
# Out of bounds values : 0 OK
# Avg bus bandwidth    : 210.255
#
# Collective test concluded: all_reduce_perf
```

ii. 多机nccl测试

官方的troubleshooting页面

https://docs.nvidia.com/deeplearning/nccl/user-guide/docs/troubleshooting.html#pci-access-control-services-acs

1. 提前配置

   查询SrcValid+

   ```
   lspci -vvv | grep ACSCtl  | grep SrcValid+
   ```

   关闭SrcValid+

   ```
   lspci    -vvv | grep 'PCI bridge' | cut -d' ' -f 1 | xargs -I{} setpci -s {}  ECAP_ACS+06.w=0000
   ```

2. 文件准备 `myhosts2`

   ```
   root@H800GPU-ComServer-Node09:/home/ubuntu/startup/nccl-tests# cat myhosts2
   gpu2 slots=8
   gpu4 slots=8
   gpu5 slots=8
   gpu6 slots=8
   gpu1 slots=8
   ```

3. 通过mpirun运行

   ```
   mpirun --allow-run-as-root \
   -hostfile myhosts2 \
   -np 32 \
   -npernode 8 \
   -x LD_LIBRARY_PATH \
   -x NCCL_DEBUG=INFO \
   -x NCCL_IB_HCA=mlx5_1,mlx5_2,mlx5_4,mlx5_7 \
   -x NCCL_SOCKET_IFNAME=bond0 \
    ./build/all_reduce_perf -b 128M -e 8G -f 2 -g 1
   ```

   ```
   root@H800GPU-ComServer-Node09:/home/ubuntu/startup/nccl-tests# mpirun --allow-run-as-root -h
   ostfile myhosts2 -np 32 -npernode 8 -x LD_LIBRARY_PATH -x NCCL_DEBUG=INFO -x NCCL_IB_HCA=
   ```

```
mlx5_1,mlx5_2,mlx5_4,mlx5_7 -x NCCL_SOCKET_IFNAME=bond0  ./build/all_reduce_perf -b 769M -e 8G -f 2 -g 1 -i 0
```

```
root@H800GPU-ComServer-Node09:/home/ubuntu/startup/nccl-tests# mpirun --allow-run-as-root -hostfile myhosts2 -np 32 -npernode 8 -x LD_LIBRARY_PATH -x NCCL_DEBUG=INFO -x NCCL_IB_HCA=mlx5_1,mlx5_2,mlx5_4,mlx5_7 -x NCCL_SOCKET_IFNAME=bond0  ./build/all_reduce_perf -b 769M -e 8G -f 2 -g 1 -i 0
```

总结

1. 需要安装的驱动

   ib卡驱动OFED，gpu卡驱动CUDA，nvswitch驱动fabric manager

2. 概念的总结

   IB网卡需要的安装包OFED是一个包含了固件、库、驱动、工具集的合集

   GPU网卡安装包 CUDA是一个包含了固件、库、驱动、工具集的合集

   正常安装驱动，一般就是一个deb包，当然这个包也有对应的依赖

3. 关于硬件选型

   400G QSFP112 OSFP112 （APC端面，端面是指的模块和线缆的接口）

   200G 都是Q(sfp)56的。Q的是向后兼容的。Q112向下兼容Q56 Q28

   Q112 400G APC

   Q56 200G UPC/APC（其实是Q112降速）较贵

   Q28 100G UPC

   VR SR DR FR LR这个选型要清晰

   多模中OM3 OM4

   模式中，单模虽然贵，但是贵在稳定。即使短距离使用单模也还是稳定的。

4. 应该如何升级关注什么

   a. 交换机要和网卡的固件版本匹配

   b. 在官方建议中（见上“查询最新版本的连接”）可以看到最新的版本中对应的交换机的和网卡的固件版本。建议升级到最新的LTS版本。因为如果遇到问题开case nvidia基本上会要求先升版本

   c. 交换机需要关注的点：固件、CPLD（其实就是电源风扇模块的固件）。带管理的交换机，需要升级系统。升级系统的同时，就升级了固件，所以不能单独升级固件。

      交换机的固件要统一

   d. 网卡需要关注的点：OFED的版本应该和固件所匹配。尤其是在换网卡的时候需要注意。

      网络中的网卡固件要统一。

5. 网卡的对应关系

   a. PCI插槽和网卡的对应关系

```
root@UFM-StorNet-Node01:~# lspci | grep -i mellanox
17:00.0 Infiniband controller: Mellanox Technologies MT28908 Family [ConnectX-6]
31:00.0 Infiniband controller: Mellanox Technologies MT28908 Family [ConnectX-6]
32:00.0 Ethernet controller: Mellanox Technologies MT27710 Family [ConnectX-4 Lx]
32:00.1 Ethernet controller: Mellanox Technologies MT27710 Family [ConnectX-4 Lx]
```

   网卡的状态

```
root@UFM-StorNet-Node01:~# ibstat
CA 'mlx5_0'
        CA type: MT4123
        Number of ports: 1
        Firmware version: 20.43.3608
        Hardware version: 0
        Node GUID: 0x946dae03000ed0a0
```

```
        System image GUID: 0x946dae03000ed0a0
        Port 1:
                State: Active
                Physical state: LinkUp
                Rate: 200
                Base lid: 1
                LMC: 0
                SM lid: 1
                Capability mask: 0xa651e84a
                Port GUID: 0x946dae03000ed0a0
                Link layer: InfiniBand
CA 'mlx5_1'
        CA type: MT4123
        Number of ports: 1
        Firmware version: 20.40.1000
        Hardware version: 0
        Node GUID: 0xb83fd203007b92c2
        System image GUID: 0xb83fd203007b92c2
        Port 1:
                State: Down
                Physical state: Polling
                Rate: 10
                Base lid: 65535
                LMC: 0
                SM lid: 0
                Capability mask: 0xa651e848
                Port GUID: 0xb83fd203007b92c2
                Link layer: InfiniBand
CA 'mlx5_bond_0'
        CA type: MT4117
        Number of ports: 1
        Firmware version: 14.32.1010
        Hardware version: 0
        Node GUID: 0x30b9300300354afd
        System image GUID: 0x30b9300300354afd
        Port 1:
                State: Active
                Physical state: LinkUp
                Rate: 10
                Base lid: 0
                LMC: 0
                SM lid: 0
                Capability mask: 0x00010000
                Port GUID: 0x28de01fffe4f2803
                Link layer: Ethernet
```

网卡和PCI-e的关系

ls -l /sys/class/infiniband/mlx5_1/device

```
root@UFM-StorNet-Node01:~# ls -l /sys/class/infiniband/mlx5_1/device
lrwxrwxrwx 1 root root 0 Aug  6 11:30 /sys/class/infiniband/mlx5_1/device → ../../../0000:31:00.0
root@UFM-StorNet-Node01:~# ls -l /sys/class/infiniband/mlx5_0/device
lrwxrwxrwx 1 root root 0 Aug  6 11:30 /sys/class/infiniband/mlx5_0/device → ../../../0000:17:00.0
```

b. 网卡和对应的以太的关系

```
root@UFM-StorNet-Node01:~# ibdev2netdev
mlx5_0 port 1 ⟹ ibs5 (Down)
mlx5_1 port 1 ⟹ ibs3 (Down)
mlx5_bond_0 port 1 ⟹ bond0 (Up)
```

6. IP和IB的的对应关系

   两种概念

   a. 我的网卡工作在IP模式下

      这种情况下，可以吧IB网卡就理解为正常的IP网卡就可以了。完全一模一样

   b. 我的网卡工作在IB模式下

      例如上面我们看到IB和IP的对应关系

```
root@UFM-StorNet-Node01:~# ibdev2netdev
mlx5_0 port 1 ⟹ ibs5 (Down)
mlx5_1 port 1 ⟹ ibs3 (Down)
mlx5_bond_0 port 1 ⟹ bond0 (Up)
```

首先在这里看到Down与Up需要配置一个IP地址，且执行`ip link set dev ibs5 up` 就可以up对应的以太网端口了。

IP地址是跑在IP over IB的模式下。正常的RDMA访问仍然是IB数据包。IP仅用于ping或者找到主机方便或者握手或者心跳。