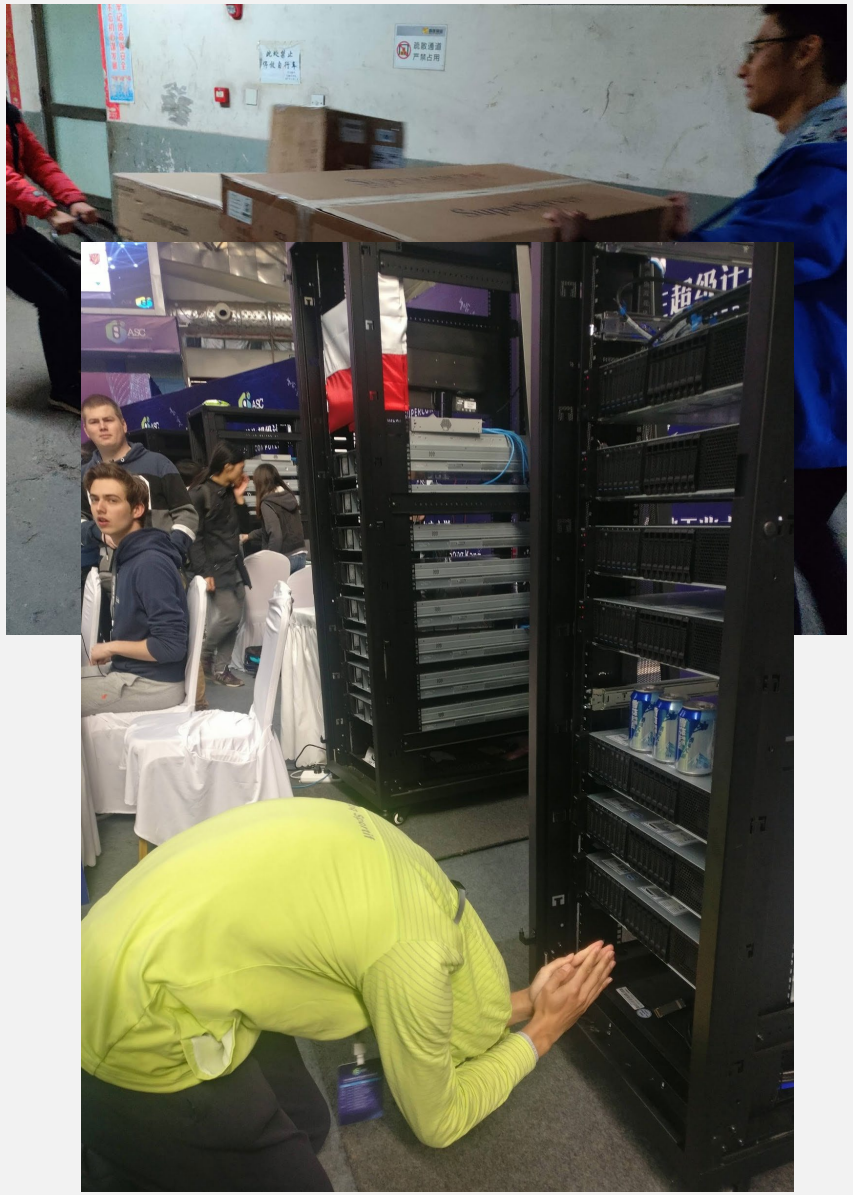
如何科学地在超算队当运维

——集群硬件管理与软件维护

陈晟祺
i@harrychen.xyz

运维需要做的事情

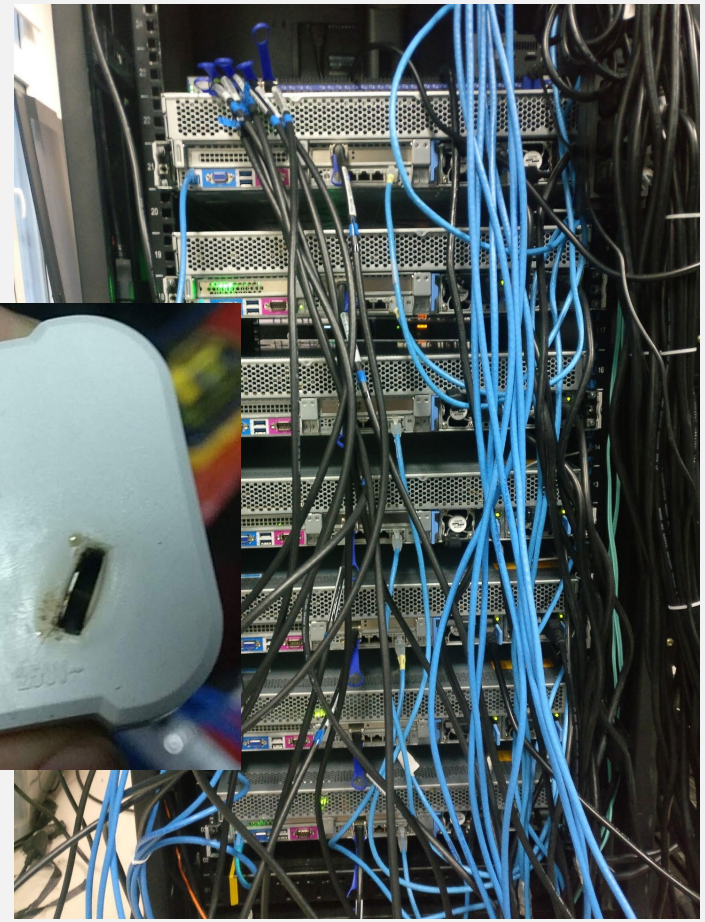
- 体力活
  - （每次比赛）组装和拆卸集群
  - （频繁地）搬运服务器、交换机、上架、下架
  - （更频繁地）装卸 CPU、GPU、IB卡、SSD 等硬件
- 脑力活
  - 安装与维护系统、更新软件、控制权限
  - 配 BMC、配网络、配存储
  - 在被别人搞坏的时候把机器修好
- 玄学活
  - 在集群装好之后施法以提高散热效率
- 还有……
  - 所有别人不愿意/不会做的



硬件管理

从机房说起

- 推荐的机器放置环境：
  - 良好的机房环境和质量较好的机柜
  - 服务器间隔放置（方便维护）
  - 整洁的布线（保持良好的心情）
- 下面是反例
- 千万注意功耗限制！
  - 不然真的会着火
  - （此处有真实的惨剧）



上架前

- 追踪手头有的所有资产
  - 服务器、交换机、显示器等整件
  - 独立的 CPU、显卡、硬盘、IB 卡等散件
  - 记录序列号、MAC地址等唯一标识
- 保留好各种容易丢的东西
  - 没有用到的显卡、IB 卡的长短挡板、电源转接线
  - 各种位置的螺丝（建议多买一些备用）
- 给机器编号，分配位置、IP 等

机器上架后

- 配置 BIOS
  - 检查 CPU 和内存的型号、**数量**、**频率**（又一桩血案）
  - 检查并按需调整 CPU 电源管理（C State、P State、Turbo Boost 等）
  - 确认引导选项（按需 UEFI/Legacy）
  - 按需配置硬盘 RAID
- 配置 BMC**（很重要）**
  - 带外（out-of-band）管理，无论系统是什么状态都能救你一命
  - 名称不本质：iLO (HPE) / iDRAC (Dell) / IPMI (General)
  - 都带远程控制台，甚至还能远程 mount 光驱
  - 都遵循 IPMI 标准，可以使用 ipmitool 在系统内/远程进行访问
    - （通用）读取电源状态、当前功耗、风扇转速等
    - （某些机器上）可以通过魔法命令控制风扇等硬件状态
  - 就算不用，用户名密码最好也要改掉（以防万一）

机器上架后（续）

- 设计网络拓扑
  - 傻瓜交换机：都插上去一把梭
  - 妥善利用管理交换机的各种功能（建议带好 console 线）
    - VLAN 隔离内外网（如果有外网，不然的话……）
    - ACL 控制访问权限
    - 还是要记得改密码啊
  - 建议留有冗余物理线路，防止突发情况（还是血案）
- 如果没有/不想要交换机
  - 以太网可以使用 Linux 的 Ethernet Bridge（小心 CPU 占用）
  - 较新的 VPI IB 卡可以切换到以太网模式使用 Host Chaining Mode
    - 但是坑可能比较多……
    - 后果自负

系统维护

操作系统

- 如果你足够强的话……
  - Proudly use Microsoft® Windows™!
- 当然我们一般还是会用 Linux
  - 强烈建议使用 Debian，Ubuntu 次之（点名批评 netplan）
  - 当然 CentOS / Fedora / RHEL 大概也能用（但是我不会）
  - 不推荐：Arch / Gentoo（除非你愿意 365 × 7 × 24 当运维）
- 稳定的系统总是好的
  - ……但是这并不意味着不需要及时更新
  - 特别是内核和 CPU microcode 等关键的包，有必要可以上 backports（又一桩血案）

存储与文件系统

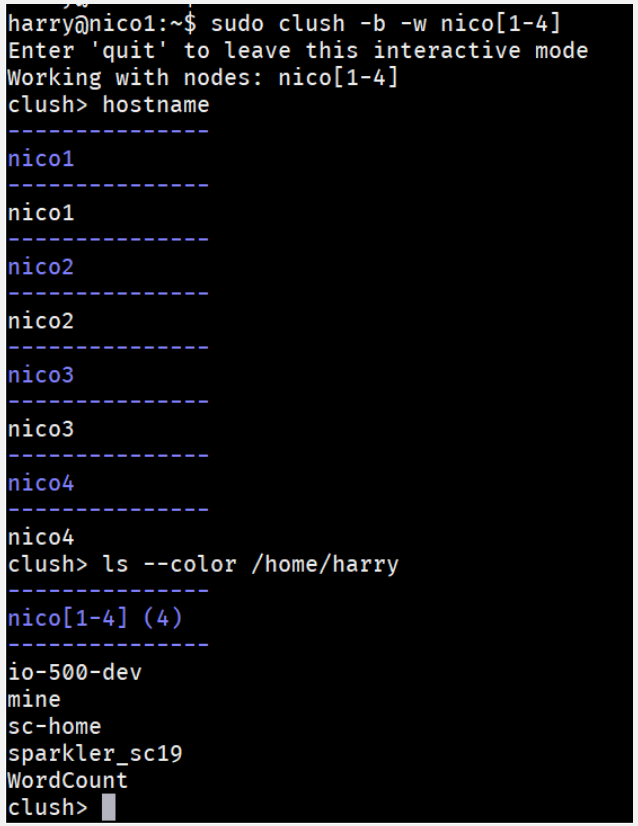
- 多盘可以组 RAID（提升性能或者可靠性）
  - 硬 RAID 卡
  - 软 RAID（mdadm）
- EXT4 是个好东西！
  - 无脑用就完事了
- 有兴趣可以尝试 ZFS on Linux
  - 通过 pool 管理存储设备，动态增减容量
  - pool 支持各种不同的 RAID
  - 实时 snapshot，copy-on-write，随时回滚（妈妈再也不怕我手残了）
- 但是**永远不要忘记**备份！！！
  - Intel 的企业级 SSD 也是可以莫名其妙坏掉的，还是两次

共享存储

- 穷人的解决方案：NFS (over Ethernet / IPoIB)
  - 糙快猛，一步解决问题
  - 注意安全问题：IP 白名单，是否开启 no_squash_root
  - 然而性能偶尔捉急，网络有问题还会 D 住
- 一个更好的尝试：NFS over RDMA
  - 但是我不会配
- 如果喜欢折腾，可以试试分布式/并行文件系统
  - CephFS / PVFS 等
  - 据说很鲁棒，但是很难配，我也没配过
  - 可以有很高的吞吐量，但是 latency 比较堪忧
- 小心共享存储带来的功耗波动！

用户权限与集群控制

- 统一用户系统
  - 科学的方案：LDAP ~~（但是据说配过 LDAP 的毕业都困难了）~~
  - 土制方案：手动同步 /etc/{passwd, shadow, group}
- 用户权限：视情况分配
  - 所有人都给 sudo：方便，然而可能出锅
  - 严格控制权限：什么事情都有人找你
- 统一控制集群
  - 使用 key 而非密码登录
  - 存在 clusterssh, tmux-cssh 等工具
  - 强推 ClusterShell（自动聚合，支持分组）
  - 可用于统一控制 CPU 频率/风扇转速等



软件/支持库版本维护

- 众所周知，超算题需要各种各样的工具和库（以及不同版本+编译选项）
  - 编译器：GCC / ICC / Clang / PGI
  - MPI：OpenMPI / Intel MPI / Mellanox HPC-X
    - 不同的 fabrics，CUDA awareness
  - 数学库：CuBLAS / MKL / OpenBLAS
  - 不同的编译选项：打开 AVX512？O3？
- 不会打架的可以用系统包管理器（有吗？）
- 会打架的需要分开管理
  - 下策：分别手动编译
  - 中策：手动编译后使用 environmental-modules 等工具管理
  - 上策：使用 Spack 享受一条龙服务

傲傲说：Spack 是个好东西！

- 一键安装：https://spack.io/
- 支持各种常用软件/库的不同版本
- 支持任意指定编译器/依赖
  - 将版本+编译选项+工具链进行 hash
  - 一键安装整条链
- 使用时一键 load 即可
- **它拯救了我的 CESM**

```
root@nico1:/home/harry# spack spec openmpi +cuda fabrics=ucx
Input spec
--------------------------------
openmpi+cuda fabrics=ucx

Concretized
--------------------------------
openmpi@3.1.5%gcc@9.2.0+cuda+cxx_exceptions fabrics=ucx ~java~legacylaunchers~memchecker~pmi schedulers=none ~sqlite3~thread_multiple+vt arch=linux-
    ^hwloc@1.11.11%gcc@9.2.0~cairo+cuda~gl+libxml2~nvml+pci+shared arch=linux-debian10-cascadelake
        ^cuda@10.2.89%gcc@9.2.0 arch=linux-debian10-cascadelake
        ^libpciaccess@0.13.5%gcc@9.2.0 arch=linux-debian10-cascadelake
            ^libtool@2.4.6%gcc@9.2.0 arch=linux-debian10-cascadelake
                ^m4@1.4.18%gcc@9.2.0 patches=3877ab548f88597ab2327a2230ee048d2d07ace1062efe81fc92e91b7f39cd00,fc9b61654a3ba1a8d6cd78ce087e7c96366c29
                    ^libsigsegv@2.12%gcc@9.2.0 arch=linux-debian10-cascadelake
            ^pkgconf@1.6.3%gcc@9.2.0 arch=linux-debian10-cascadelake
            ^util-macros@1.19.1%gcc@9.2.0 arch=linux-debian10-cascadelake
        ^libxml2@2.9.9%gcc@9.2.0~python arch=linux-debian10-cascadelake
            ^libiconv@1.16%gcc@9.2.0 arch=linux-debian10-cascadelake
            ^xz@5.2.4%gcc@9.2.0 arch=linux-debian10-cascadelake
            ^zlib@1.2.11%gcc@9.2.0+optimize+pic+shared arch=linux-debian10-cascadelake
        ^numactl@2.0.12%gcc@9.2.0 arch=linux-debian10-cascadelake
            ^autoconf@2.69%gcc@9.2.0 arch=linux-debian10-cascadelake
                ^perl@5.30.0%gcc@9.2.0+cpanm+shared+threads arch=linux-debian10-cascadelake
                    ^gdbm@1.18.1%gcc@9.2.0 arch=linux-debian10-cascadelake
                        ^readline@8.0%gcc@9.2.0 arch=linux-debian10-cascadelake
                            ^ncurses@6.1%gcc@9.2.0~symlinks~termlib arch=linux-debian10-cascadelake
            ^automake@1.16.1%gcc@9.2.0 arch=linux-debian10-cascadelake
    ^ucx@1.6.0%gcc@9.2.0 arch=linux-debian10-cascadelake
        ^rdma-core@20%gcc@9.2.0 build_type=RelWithDebInfo arch=linux-debian10-cascadelake
            ^cmake@3.16.2%gcc@9.2.0~doc+ncurses+openssl+ownlibs~qt arch=linux-debian10-cascadelake
                ^openssl@1.1.1d%gcc@9.2.0+systemcerts arch=linux-debian10-cascadelake
            ^libnl@3.3.0%gcc@9.2.0 arch=linux-debian10-cascadelake
                ^bison@3.4.2%gcc@9.2.0 patches=89aa362716d898edd0b5c6ae4208dc1b6992887774848a09e8021afd676f7d61 arch=linux-debian10-cascadelake
                    ^diffutils@3.7%gcc@9.2.0 arch=linux-debian10-cascadelake
                    ^help2man@1.47.11%gcc@9.2.0 arch=linux-debian10-cascadelake
                        ^gettext@0.20.1%gcc@9.2.0+bzip2+curses+git~libunistring+libxml2+tar+xz arch=linux-debian10-cascadelake
                            ^bzip2@1.0.8%gcc@9.2.0+shared arch=linux-debian10-cascadelake
                            ^tar@1.32%gcc@9.2.0 arch=linux-debian10-cascadelake
                ^flex@2.6.4%gcc@9.2.0+lex patches=09c22e5c6fef327d3e48eb23f0d610dcd3a35ab9207f12e0f875701c677978d3 arch=linux-debian10-cascadelake
                    ^findutils@4.6.0%gcc@9.2.0 patches=84b916c0bf8c51b7e7b28417692f0ad3e7030d1f3c248ba77c42ede5c1c5d11e,bd9e4e5cc280f9753ae14956c4e4
                        ^texinfo@6.5%gcc@9.2.0 patches=12f6edb0c6b270b8c8dba2ce17998c580db01182d871ee32b7b6e4129bd1d23a,1732115f651cff98989cb0215d85

root@nico1:/home/harry#
```

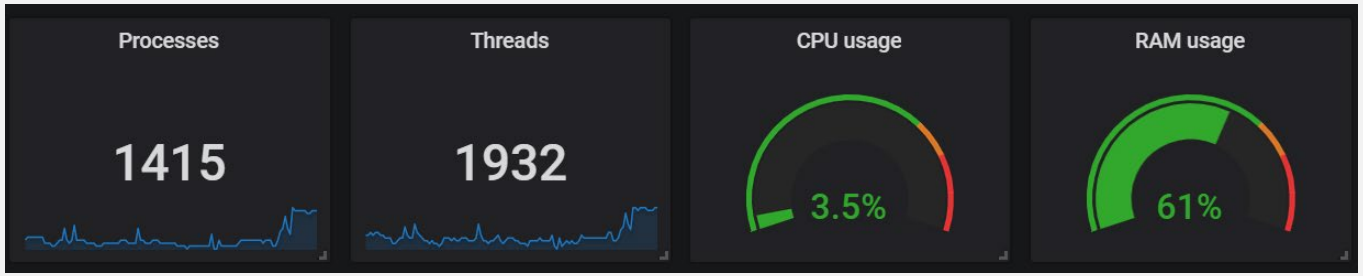
运行监控

监控的重要性

- 观察当前状态
  - 风扇转速、单机功耗、PDU 功耗等关键参数
  - 把爆功耗、高温扼杀在摇篮中
- 记录历史结果
  - 观察应用的功耗曲线以推断计算特征
  - 方便事后分析
- ~~看曲线不是很爽吗~~

搭建一套科学的监控系统

- 最重要的一点：**不能**用集群里的机器
- 全家桶：
  - 时序数据库：InfluxDB
  - 数据可视化：Grafana
  - 数据采集：Telegraf
    - Out-of-band：通过 IPMI 插件采集其他机器的功耗、风扇等数据
    - In-band：通过 cpu/disk/diskio/system 等插件采集本系统数据（**风险自负**）
    - 其他：通过 SNMP 插件采集 PDU 功耗等额外数据
- 可视化：开发脑洞！
  - 可以加入自动压功耗等功能



功耗控制

- CPU 功耗
  - 注意打开 C States / P States，Turbo Mode 按需开启
  - 使用 cpupower 命令可控制 CPU 频率
  - **小心**：随温度上升，同样频率的功耗也会上升
- GPU 功耗
  - nvidia-smi –pm 1 打开 Persistent Mode
  - nvidia-smi –pl / -ac 控制 Power Limit（比较慢）/ SM频率（不太稳）
- 风扇
  - 各种机器都有神秘的 IPMI 指令精细调整风扇
  - 千万别关了忘记开

另一些有用的小技巧

- 在线开关一些 CPU 核心：/sys/devices/system/cpu/cpu$i/online
  - 可以临时关掉超线程
  - 并不能降低功耗
- 检查 PCI-E 拓扑：nvidia-smi topo –m
  - GPU、CPU、IB 卡
- 检查显卡状态：nvidia-smi –q
  - 小心带宽不是 16x（血案 again）
- 检查系统配置：dmidecode

```
harry@nico1:~/sc-home/scripts$ nvidia-smi topo -m
        GPU0    GPU1    GPU2    GPU3    GPU4    GPU5    GPU6    GPU7    mlx5_0  CPU Affinity
GPU0     X      PIX     NODE    NODE    SYS     SYS     SYS     SYS     NODE    0-15,32-47
GPU1    PIX      X      NODE    NODE    SYS     SYS     SYS     SYS     NODE    0-15,32-47
GPU2    NODE    NODE     X      PIX     SYS     SYS     SYS     SYS     NODE    0-15,32-47
GPU3    NODE    NODE    PIX      X      SYS     SYS     SYS     SYS     NODE    0-15,32-47
GPU4    SYS     SYS     SYS     SYS      X      PIX     NODE    NODE    SYS     16-31,48-63
GPU5    SYS     SYS     SYS     SYS     PIX      X      NODE    NODE    SYS     16-31,48-63
GPU6    SYS     SYS     SYS     SYS     NODE    NODE     X      PIX     SYS     16-31,48-63
GPU7    SYS     SYS     SYS     SYS     NODE    NODE    PIX      X      SYS     16-31,48-63
mlx5_0  NODE    NODE    NODE    NODE    SYS     SYS     SYS     SYS      X

Legend:

  X    = Self
  SYS  = Connection traversing PCIe as well as the SMP interconnect between NUMA nodes (e.g., QPI/UPI)
  NODE = Connection traversing PCIe as well as the interconnect between PCIe Host Bridges within a NUMA node
  PHB  = Connection traversing PCIe as well as a PCIe Host Bridge (typically the CPU)
  PXB  = Connection traversing multiple PCIe bridges (without traversing the PCIe Host Bridge)
  PIX  = Connection traversing at most a single PCIe bridge
  NV#  = Connection traversing a bonded set of # NVLinks
```

应用调度

- 不推荐使用 slurm 等作业系统
  - 粒度太粗，难以控制
  - 运行 daemon 有功耗尖峰风险
- 人工调度（基本靠吼）
  - 能够精细控制
  - 但是有冲突的风险（不小心一起跑就爆了）
  - 建议使用双重确认机制
- 建议使用脚本等工具智能调度
  - 配合上述的监控和功耗控制
  - 一个例子：傲傲的 HPL

最后……

一些提醒

- 出发前写一个 checklist，记下所有要带的东西
- 如果东西较多，分摊到人看管
- 及时把踩过的坑记下来
- 多睡觉，防止脑子不清醒
- **运维不能只有一个（这里还有血案）**

Q & A