science + computing

an atos company

High-Performance-Computing mit Docker

Sebastian Klingberg

science + computing ag
IT-Dienstleistungen und Software für anspruchsvolle Rechnernetze
Tübingen I München I Berlin I Düsseldorf

Agenda

science + computing
an atos company

- Motivation & Ziel der Thesis
- PoC:
  Integration der Docker Technologie in traditionellem High-Performance-Computing
- Fazit

science + computing

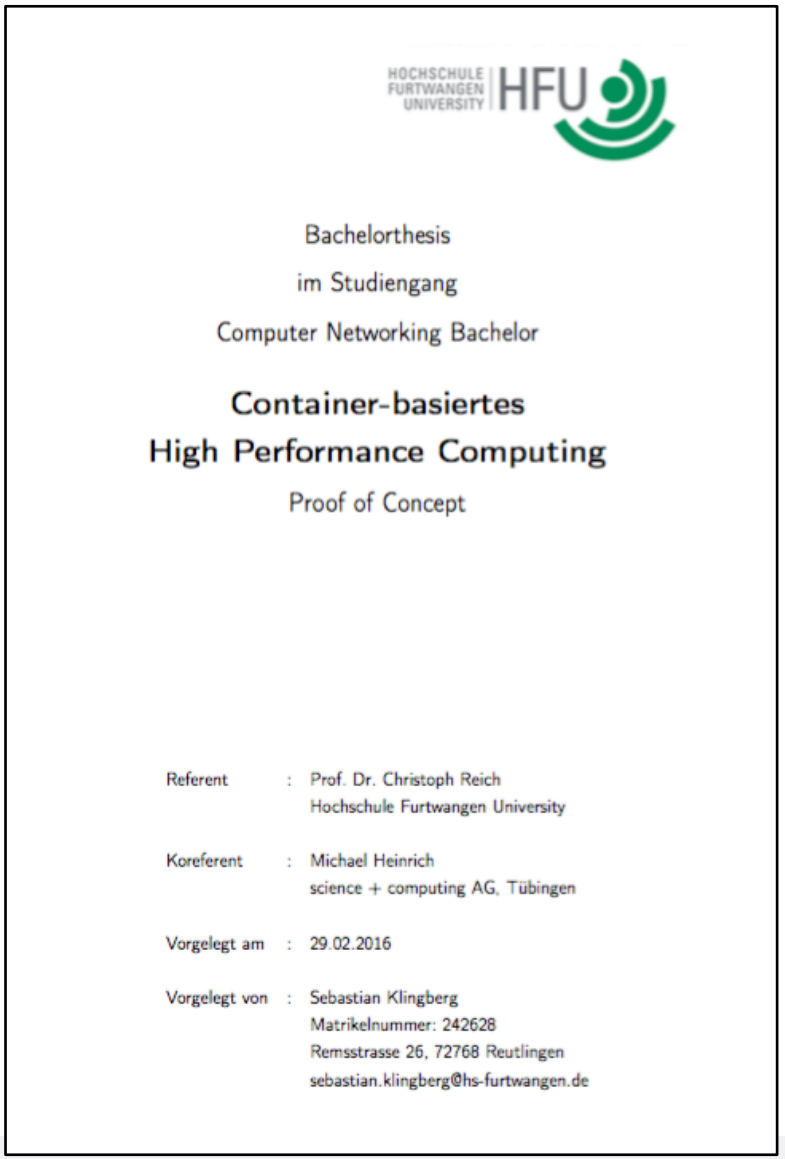
an atos company

MOTIVATION & ZIEL DER THESIS

Motivation & Forschungsziel

science + computing
an atos company

„[…] Design und Integration einer umfassenden Lösung für BatchJob-basiertes, paralleles Technical Computing über Interconnect-basiertes SM–MIMD in HPC Clustersystemen, durch den Einsatz verteilter Linux-Container …“



HOCHSCHULE FURTWANGEN UNIVERSITY HFU

Bachelorthesis

im Studiengang

Computer Networking Bachelor

Container-basiertes High Performance Computing

Proof of Concept

Referent : Prof. Dr. Christoph Reich
Hochschule Furtwangen University

Koreferent : Michael Heinrich
science + computing AG, Tübingen

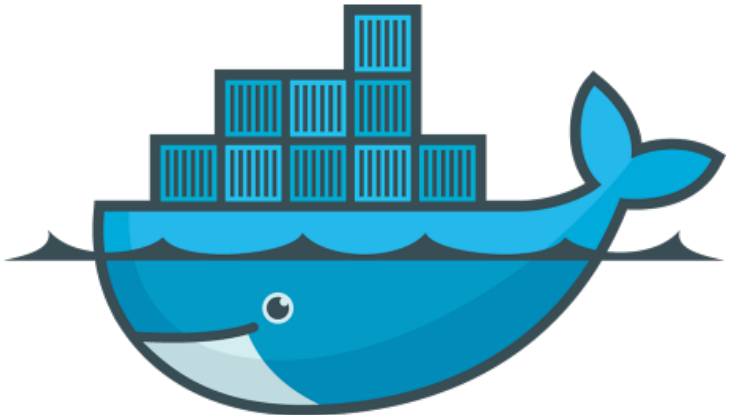
Vorgelegt am : 29.02.2016

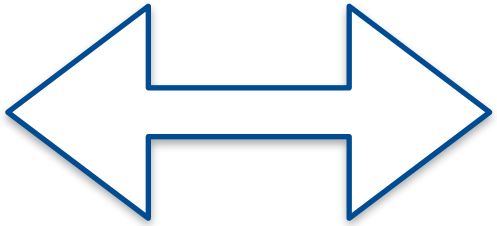
Vorgelegt von : Sebastian Klingberg
Matrikelnummer: 242628
Remsstrasse 26, 72768 Reutlingen
sebastian.klingberg@hs-furtwangen.de

Motivation & Forschungsziel ... Kurz!

science + computing
an atos company

Paralleles Rechnen über mehrere Hosts mit Docker in bestehender Umgebung!!!





HPC

Auszug:
Anforderungen & Rahmenbedingungen

science + computing
an atos company

- Aufbau & Umgebung nach „realen“ Anforderungen
  - Integration
- Einsatz von OpenSource Technologien
- Portiertbarkeit auf andere Systeme/Lösungen
- Identische Anwenderprozesse beim Jobsubmit
- Benutzer haben keinen direkten Dockerzugriff
  - Sicherheitsproblematik (Bsp.: NFSv3 UID-Spoofing)
- …

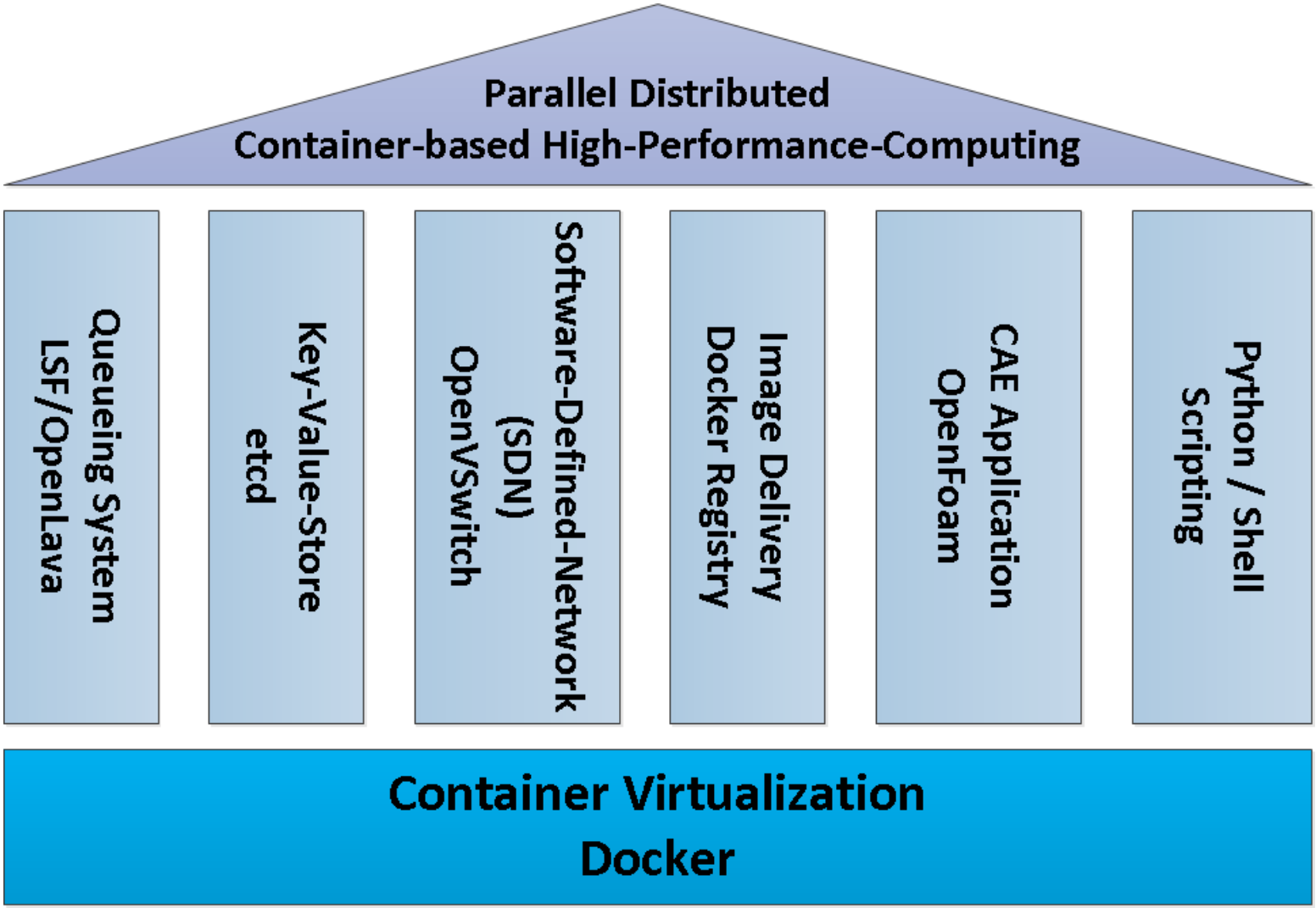
science + computing

an atos company

POC:
INTEGRATION DOCKER HPC

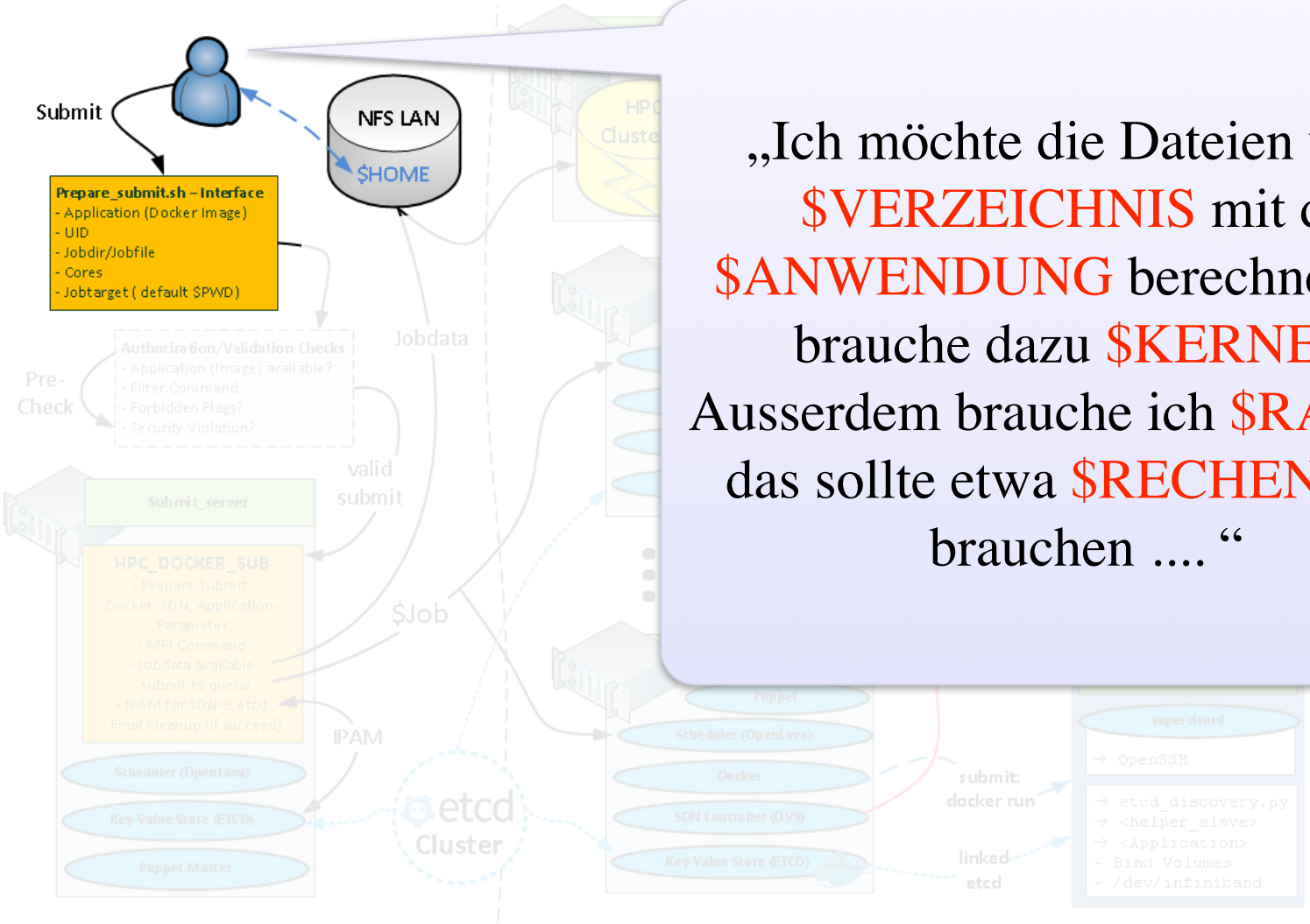
Verwendete Technologien im PoC

science + computing
an atos company



1. Anwender Submit eines parallelen Job

science + computing
an atos company



„Ich möchte die Dateien unter $VERZEICHNIS mit der $ANWENDUNG berechnen. Ich brauche dazu $KERNE ... Ausserdem brauche ich $RAM und das sollte etwa $RECHENZEIT brauchen .... “

1. Anwender Submit eines parallelen Job: Submitskript

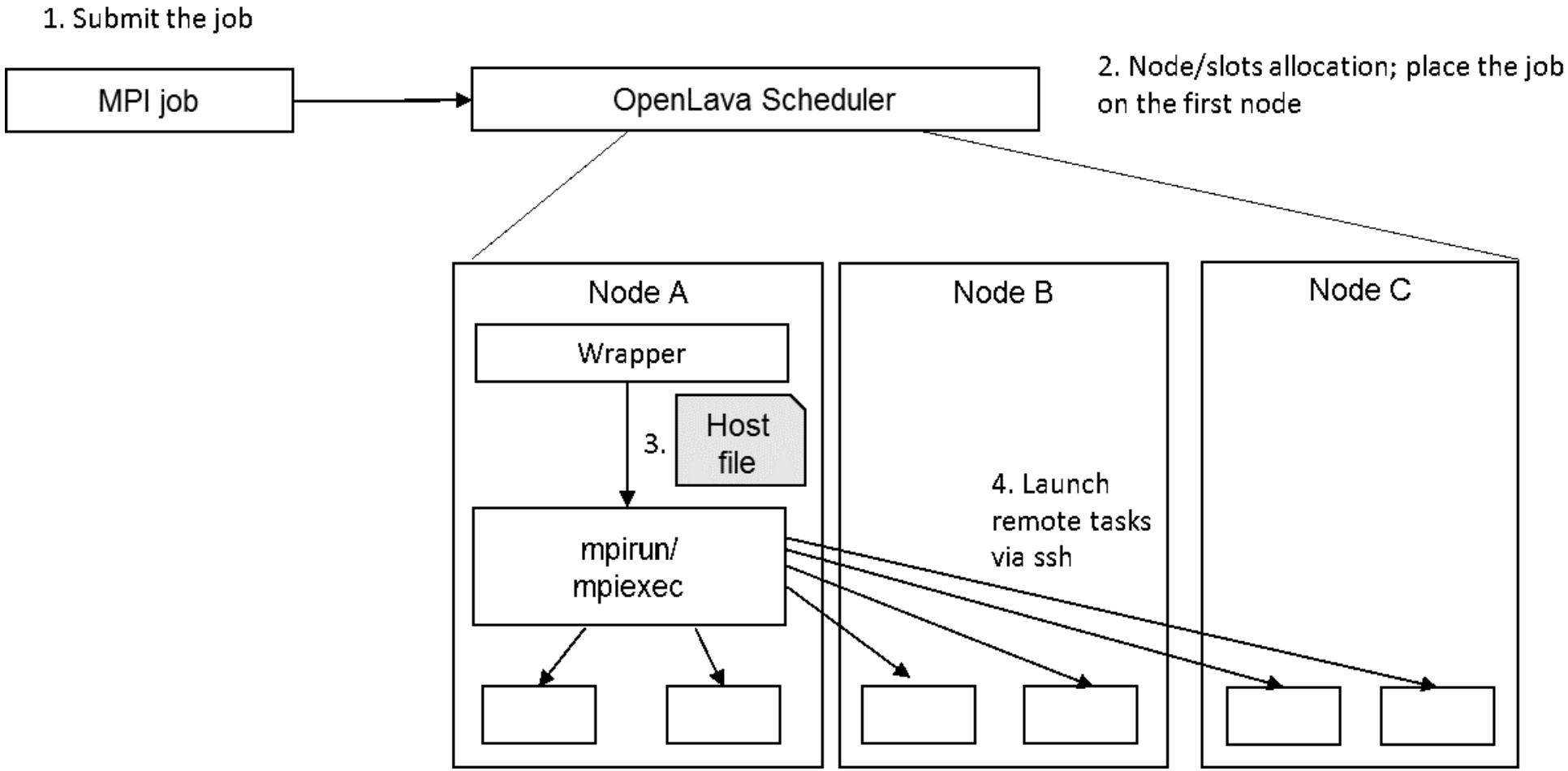
science + computing
an atos company

- Submitskript (LSF):

```
#!/bin/bash
#BSUB -P IB_testrun_openfoam_8core
#BSUB -J IB_pitzDaily_OF
#BSUB -W 00:05
#BSUB -n 8
#BSUB -o job-%J.out
#BSUB -e job-%J.err

### exec params
export APPLICATION=openfoam:2.3.0

export COMMAND="mpirun -v -hostfile system/machines -x PATH=\$PATH -x
LD_LIBRARY_PATH=\$LD_LIBRARY_PATH -x WM_PROJECT_DIR=\$WM_PROJECT_DIR  -x
MPI_BUFFER_SIZE=2000000000 -report-bindings -bind-to-core -mca btl ib
simpleFoam -parallel"

export JOBDIR="/scratch/lustre/OF_pitzDailyExptInlet_IB"

### Docker-Middleware-Wrapper
/share/bin/hpc_submit_dockerjob
```

Einschub: Parallelrechnen mit Message-Passing-Interface (MPI)

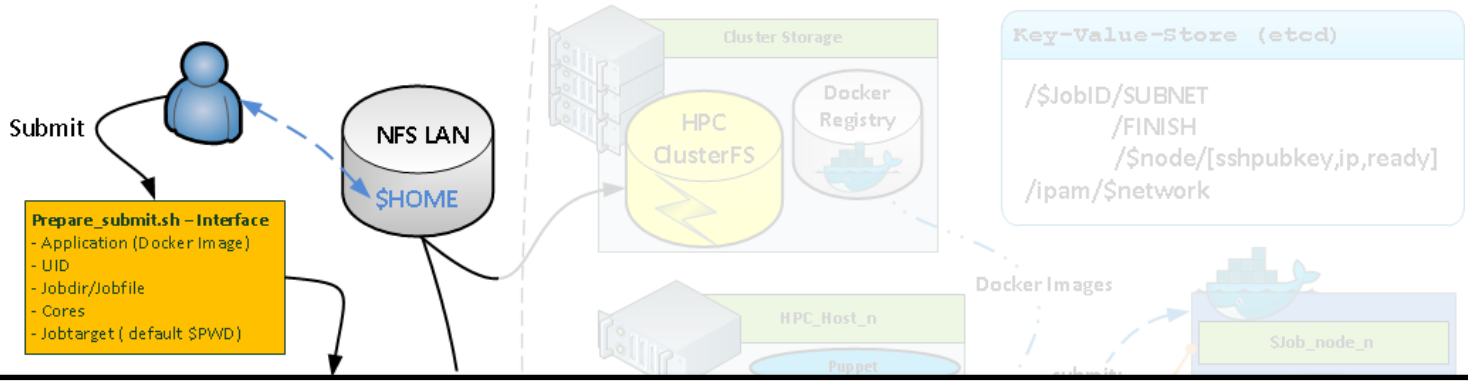
science + computing
an atos company



Quelle: http://www.teraproc.com/teraproc-blog/mpi/

1. Anwender Submit eines parallelen Job: Queueing System

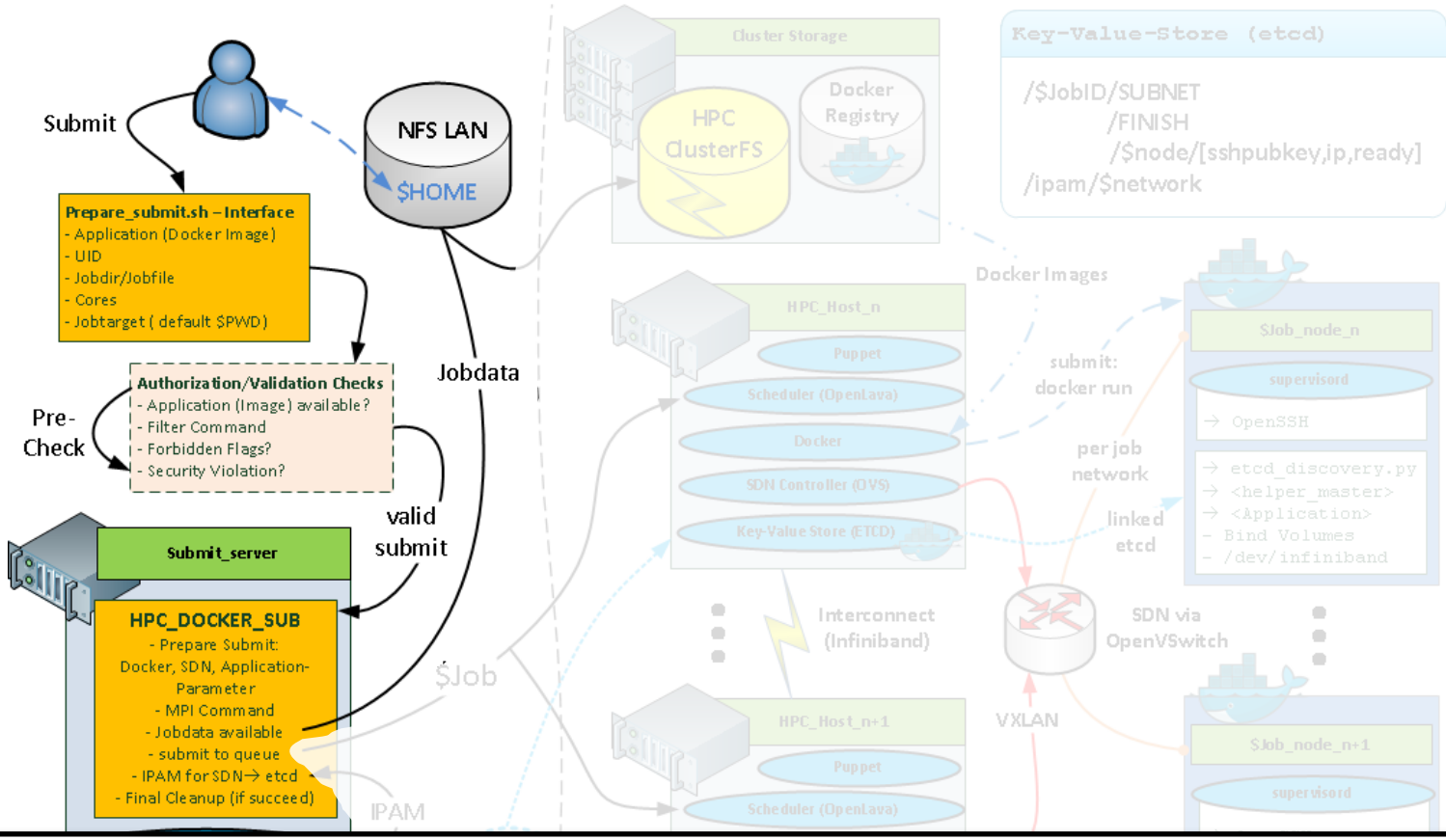
science + computing
an atos company




```
[myuser@hpc_node ~]$ bqueues
QUEUE_NAME      PRIO      STATUS      MAX  JL/U JL/P JL/H NJOBS  PEND  RUN  SUSP
normal           30    Open:Active      -    -    -    -     0     0     0     0
[myuser@hpc_node ~]$ lshosts
HOST_NAME      type    model  cpuf ncpus maxmem maxswp server RESOURCES
centos7_0     linux  IntelI5 100.0     2   986M  1999M    Yes ()
centos7_1     linux  IntelI5 100.0     2   986M  1999M    Yes ()
[myuser@hpc_node ~]$ bhosts
HOST_NAME          STATUS       JL/U    MAX  NJOBS    RUN  SSUSP  USUSP    RSV
centos7_0          ok              -      2      0      0      0      0      0
centos7_1          ok              -      2      0      0      0      0      0
```

1. Anwender Submit eines parallelen Job: Submit erfolgt!

science + computing
an atos company




```
[myuser@hpc_node ~]$ bsub < myjobsubmit
Job <772> is submitted to default queue <normal>.
```

2. Batch Job im Queueing System: Jobumgebung wird erzeugt

science + computing
an atos company

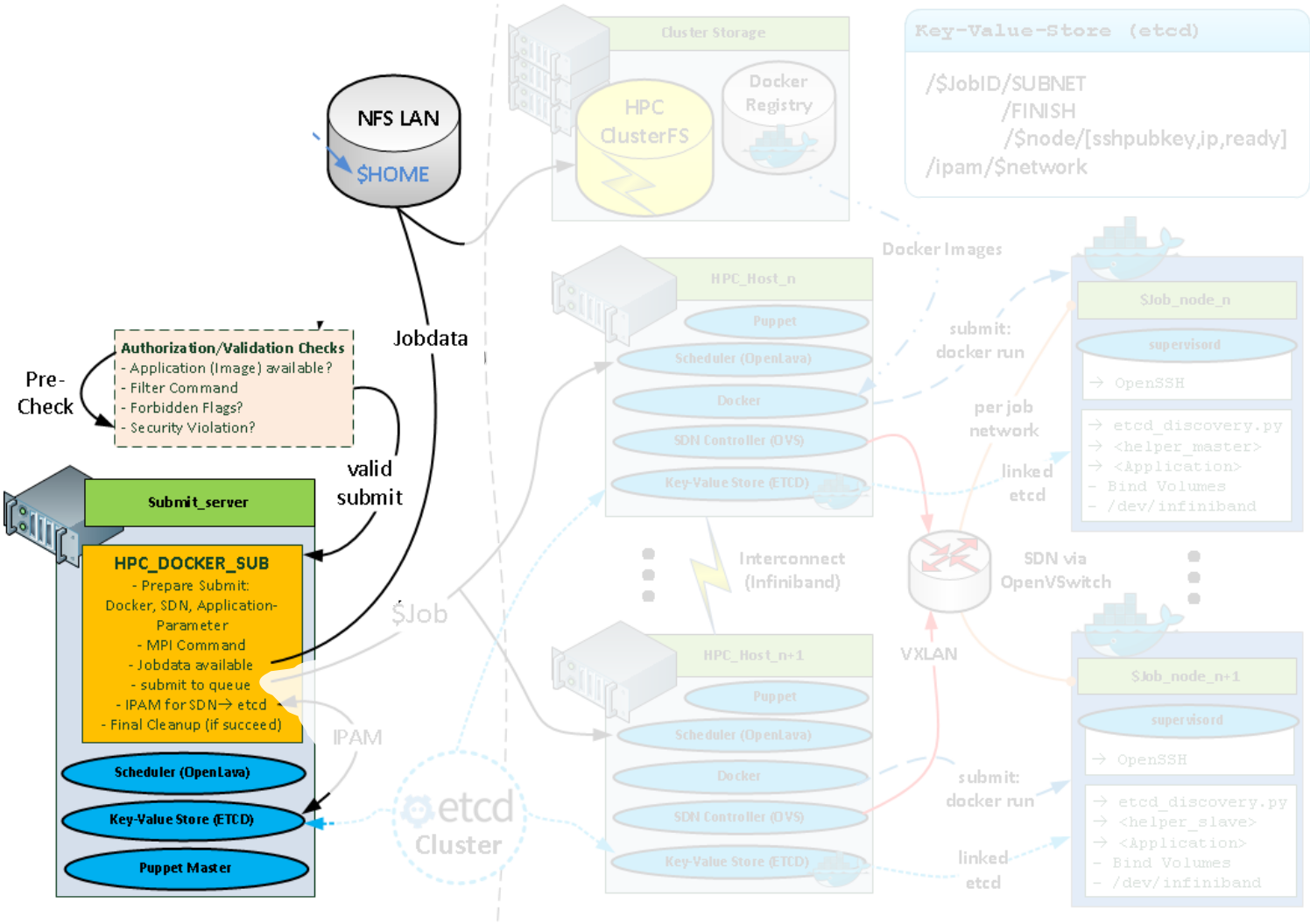
- Verfügbare Informationen der Jobumgebung (z.B. LSF)

Environment Variable	Description
LSB_JOBNAME	The name of the job
LSB_JOBID	The job ID assigned by LSF
LSB_JOBINDEX	The job array index
LSB_JOBINDEX_STEP	Step at which single elements of the job array are defined
LSB_JOBINDEX_END	Contains the maximum value of the job array index
LSB_HOSTS	The list of hosts selected by LSF to run the job
LSB_MCPU_HOSTS	The list of the hosts and the number of CPUs used
LSB_DJOB_HOSTFILE	Path to the hostfile
LSB_DJOB_NUMPROC	The number of slots allocated to the job

Quelle: https://doc.itc.rwth-aachen.de/display/CC/LSF+environment+variables

2. Batch Job im Queueing System: Bearbeiten, Filtern und Prüfen

science + computing
an atos company



2. Batch Job im Queueing System: Bearbeiten, Filtern und Prüfen

science + computing
an atos company

- Ist die geforderte Applikation als Docker Image vorhanden?
- Anpassen des Submit Befehls
  - Vorgaben (Laufzeit)
  - Ressourcen (Lizenzen, Host-spezifisch, ...)
- Sicherheit
  - Command-String filtern
    - Verwendung korrekter Executables
  - UserID für Docker Prozesse übernehmen
    - User-Spoofing unbedingt unterbinden !!!
    - Lösung z.B. NFS4 mit Kerberos

Einschub:
User Spoofing mit Docker

science + computing
an atos company

- User-Spoofing unbedingt unterbinden !!!
  - auch auf verteilten Dateisytemen!

```
[evil@docker ~]# id
uid=1234(evil) gid=1234(evil) Gruppen=1234(evil),1337(docker)
[evil@docker ~]# id skling
uid=1000(skling) gid=1000(skling) Gruppen=1000(skling)
[evil@docker ~]# cd /home/skling
bash: cd: /home/skling: Keine Berechtigung
[evil@docker ~]# docker run -it -v /home:/nfs3home -u 1000 busybox sh
/ $ id
uid=1000 gid=0(root)
/ $ touch /nfs3home/skling/EVIL_WAS_HERE && exit
```

2. Batch Job im Queueing System: Jobumgebung —> Middleware

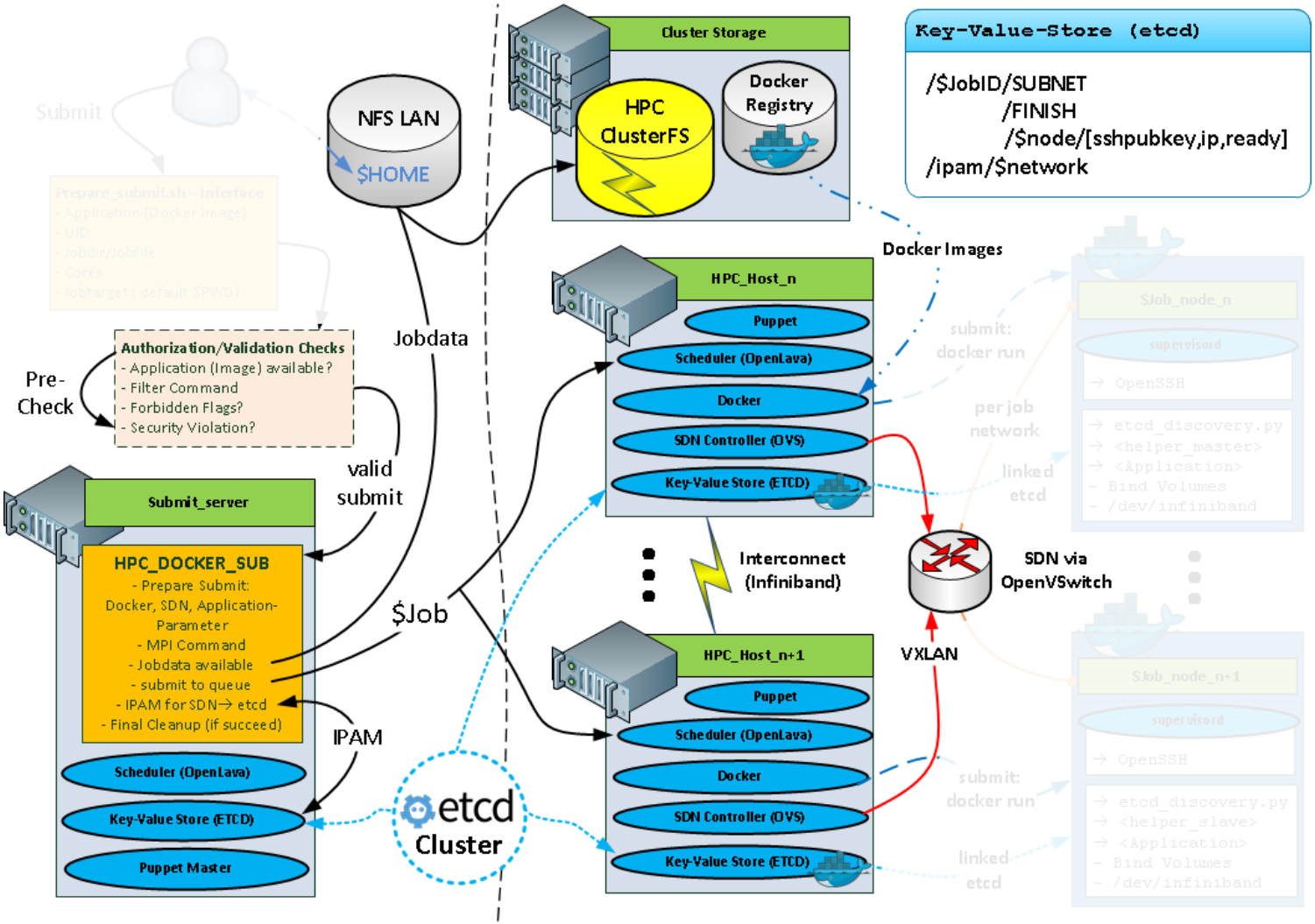
science + computing
an atos company

- Zentraler Middleware-Prozess zwischen allen beteiligten Backend-Komponenten
  - Jobumgebung wird weitergeben

```
#!/bin/bash
…

### exec params
export APPLICATION=hpc_ubuntu_openfoam:2.3.0

export COMMAND="mpirun -v -hostfile system/machines … simpleFoam -parallel"

export JOBDIR="/scratch/lustre/OF_pitzDailyExptInlet_IB"

### Docker-Middleware-Wrapper
/share/bin/hpc_submit_dockerjob
```

3. Containerumgebung vorbereiten (Middleware)

science + computing
an atos company



3. Middleware

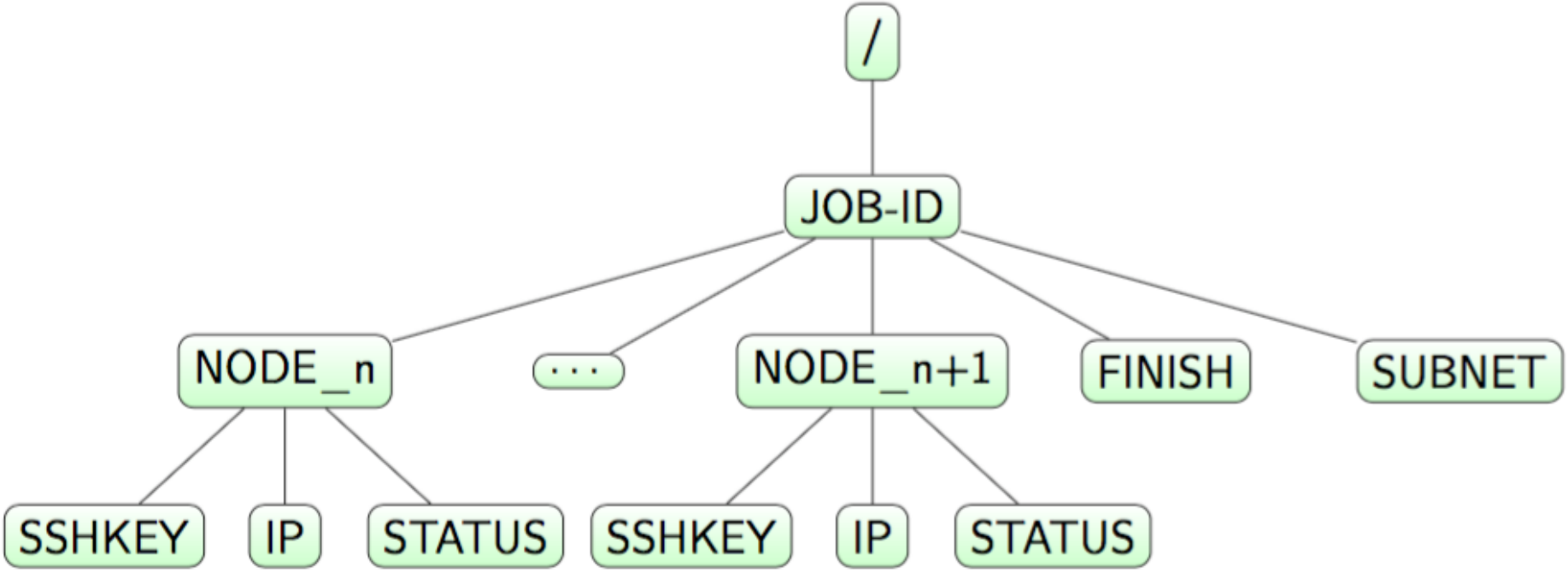
science + computing
an atos company

- Aufgaben:
  1. Check: Docker Images, Parameter, Validität
  2. Bereitstellen der Jobdaten (z.B. auf Cluster FS)
  3. Vorbereitung und Kontrolle der temporären Container-Rechenumgebung
  4. Steuerung der Container und weiteren Komponenten
  5. Reaktion/Meldung an Queueing System
  6. Aufräumen der Umgebung nach Jobende

3. Middleware: Initialisierung KVS Kommunikation

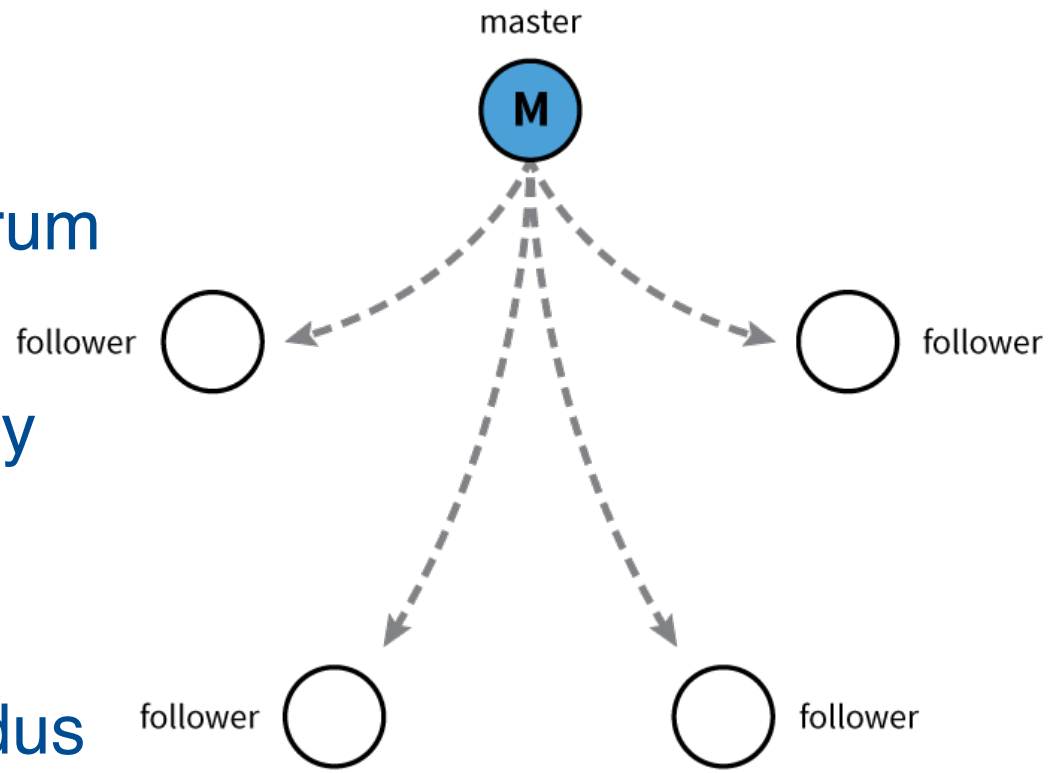
science + computing
an atos company

- Zentraler Kommunikationspunkt in der Jobumgebung
- Anlegen Jobverzeichnis im Key-Value-Store
- Reservierung eines freien IP-Netz für die Container Rechenumgebung (Alternative: IPAM)



Einschub: Key-Value-Store (KVS) etcd

- KVS aus dem CoreOS Projekt
- Robust und Schnell
- Raft Algorithmus
  - Leader + Member = Quorum
- REST-API (HTTPS)
  - Value TTLs + Watch-a-key
- Auto TLS (+ClientCerts)
- Einfacher Clusterbau
  - Member- oder Proxy-Modus
- Großer Docker Support



Quelle: https://coreos.com/etcd/

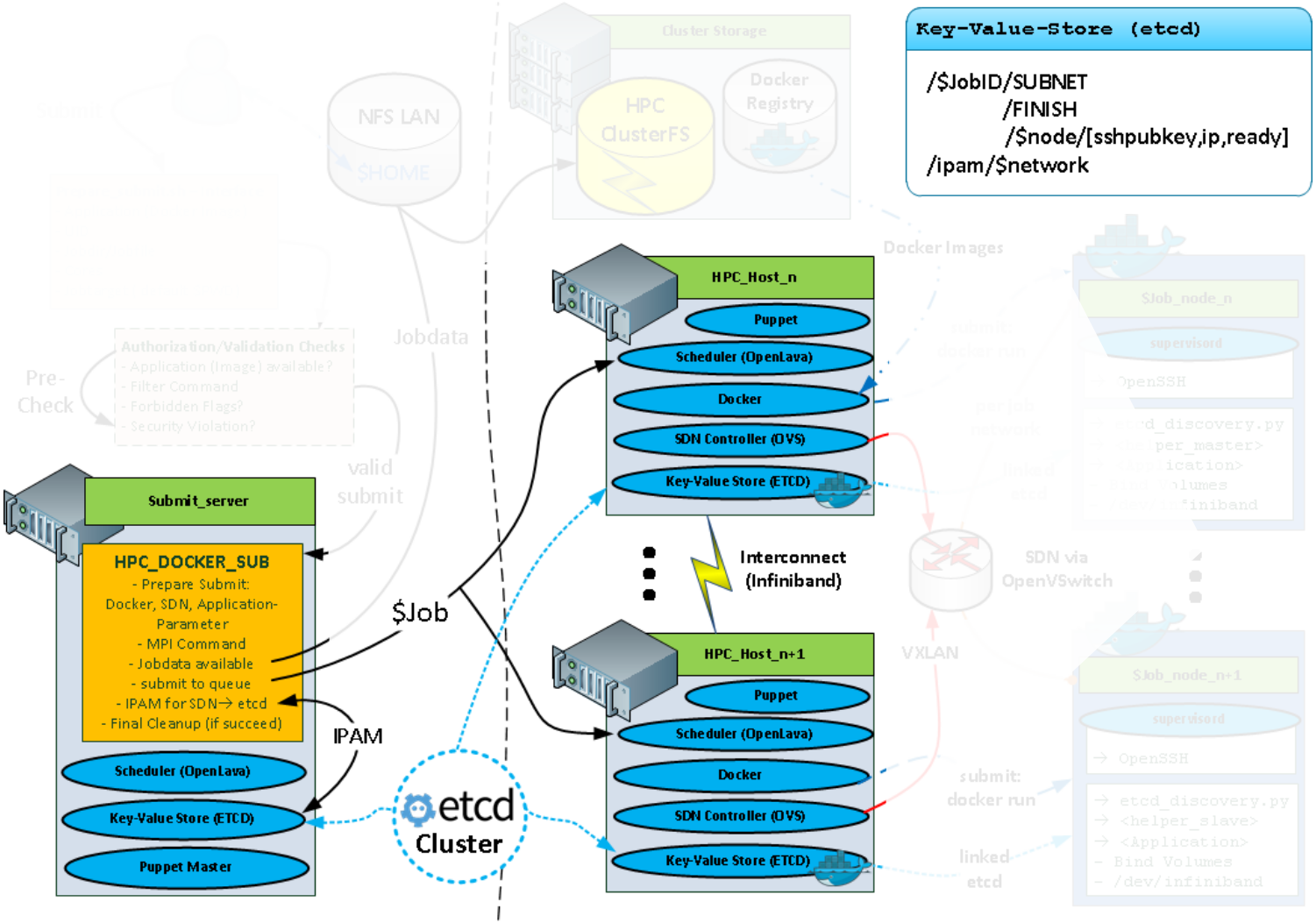
3. Middleware: etcd KVS Cluster „gedockert“

science + computing
an atos company

```
docker run -d --name=${DOCKERNAME} --restart=always \
  -p ${PEER_PORT}:${PEER_PORT} -p ${REST_API}:${REST_API} etcd:2.1 \
        etcd -name $NAME \
        -advertise-client-urls http://${HOST_ADDRESS}:${REST_API},http://${HOST_ADDRESS}:${REST:API} \
        -listen-client-urls http://0.0.0.0:${CLIENT_API2},http://0.0.0.0:${CLIENT_API1} \
        -initial-advertise-peer-urls http://${HOST_ADDRESS}:${PEER_PORT} \
        -listen-peer-urls http://0.0.0.0:${PEER_PORT} \
        -initial-cluster-token $TOKEN \
        -initial-cluster $PEER_URL_CLUSTER \
        -initial-cluster-state new
```

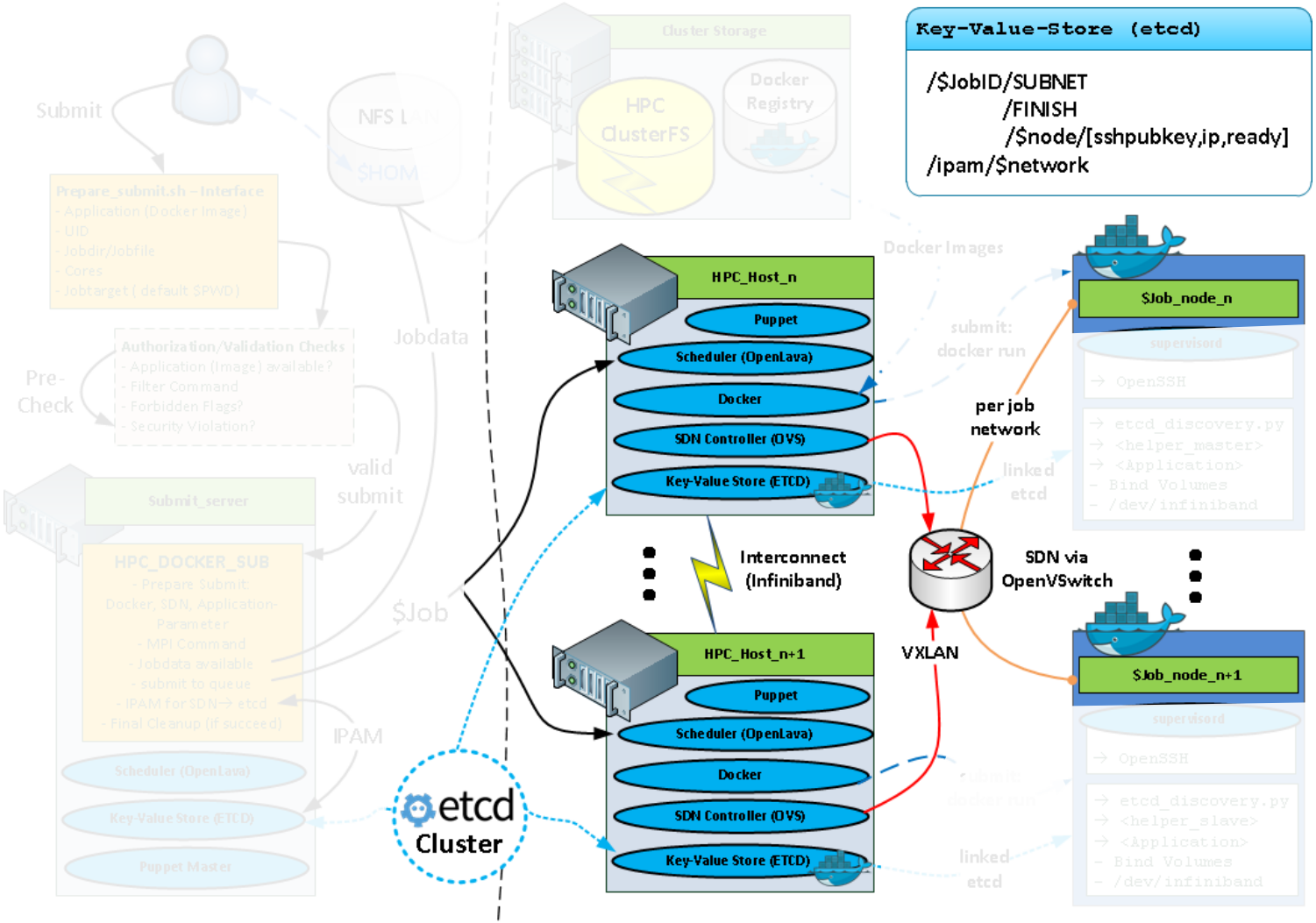
3. Middleware: etcd KVS Cluster

science + computing
an atos company



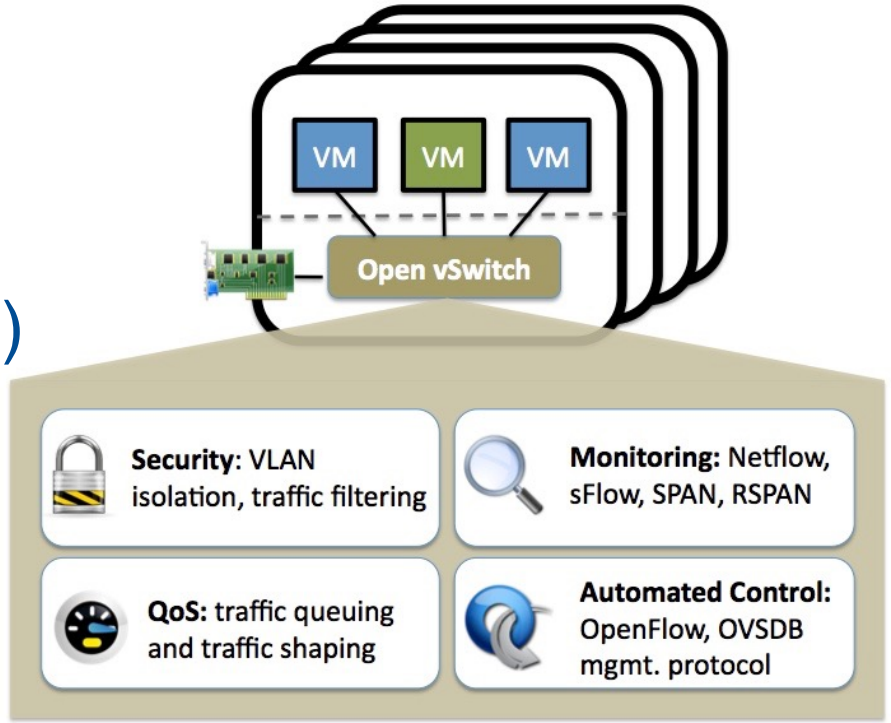
3. Middleware: Netzwerk Infrastruktur Pro Job?!

science + computing
an atos company



Einschub: OpenVSwitch

- Virtueller, software-basierter Switch für Linux Systeme
- OVS-Datapath-Modul (Performance) ab Kernel 3.10
- Einsatz:
  OpenStack Neutron, Xen, KVM,
  Virtualbox, libvirt, ...
- Software-Defined-Networks (SDN)
- OVS als Docker „networkdriver“



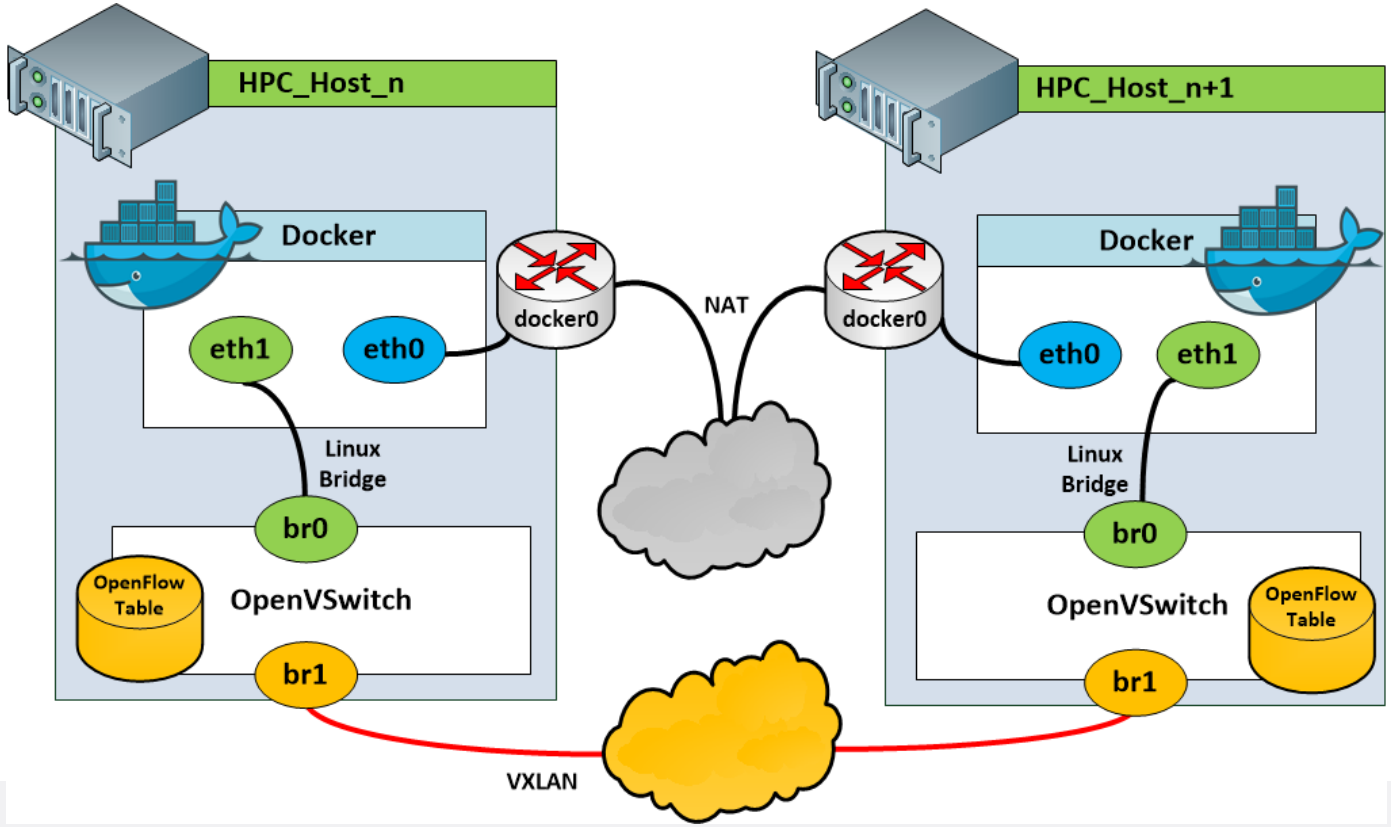

```
$ ovs-docker add-port ovs-br1 eth0 123_node_0 --ipaddress=10.0.190.1/16
```

Quelle: https://openvswitch.org

3. Middleware: Initialisierung Netzwerk Infrastruktur

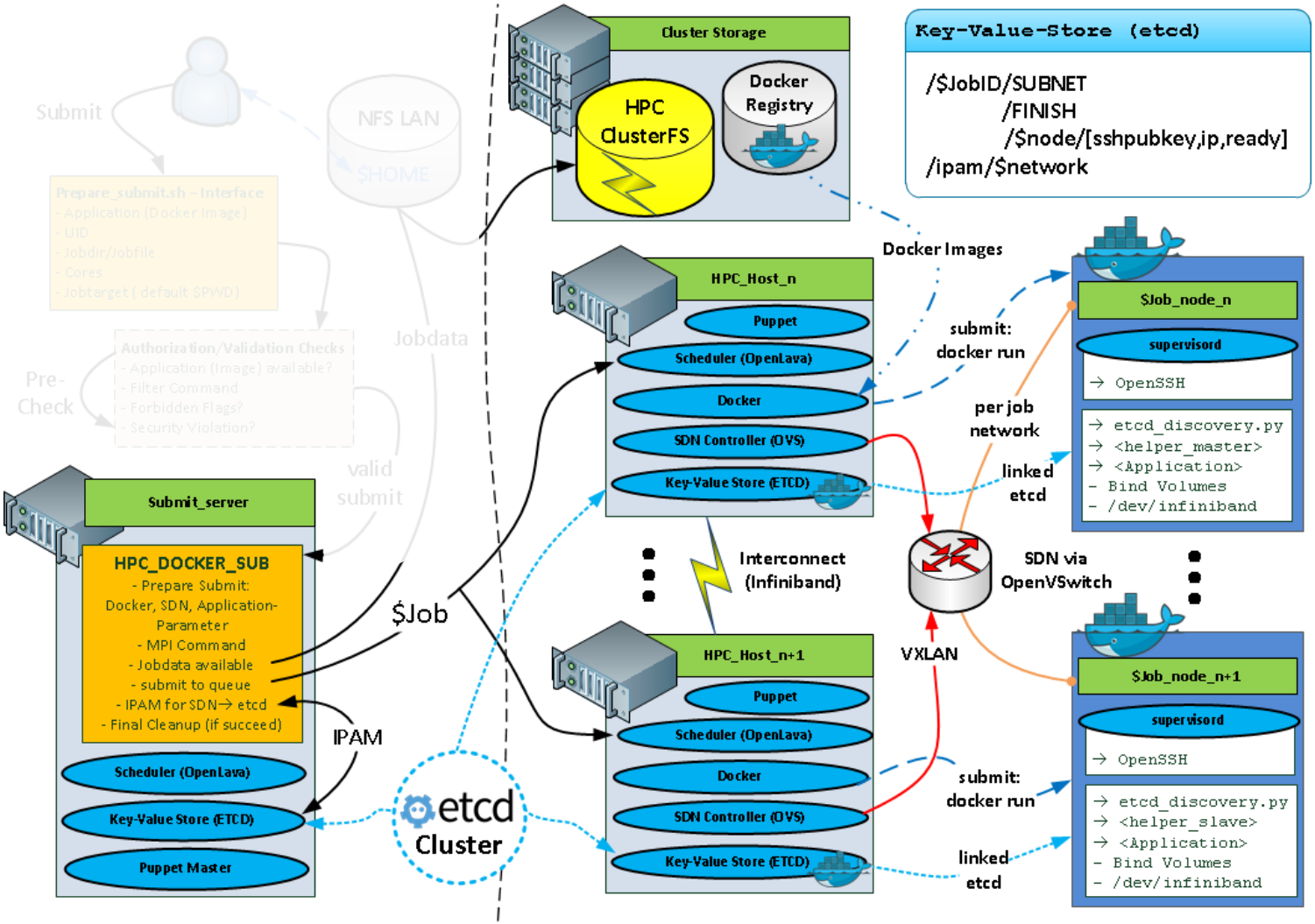
science + computing
an atos company

- Basis: OpenVSwitch + Bridges + Tunnelverbindungen
- IP aus Jobnetz an Container vergeben (+ VLAN + ...)
- Temporäre Kommunikation ohne Socket Problematik!



4. Start HPC Docker Container

science + computing
an atos company



4. Start HPC Docker Container: Ausgangssituation

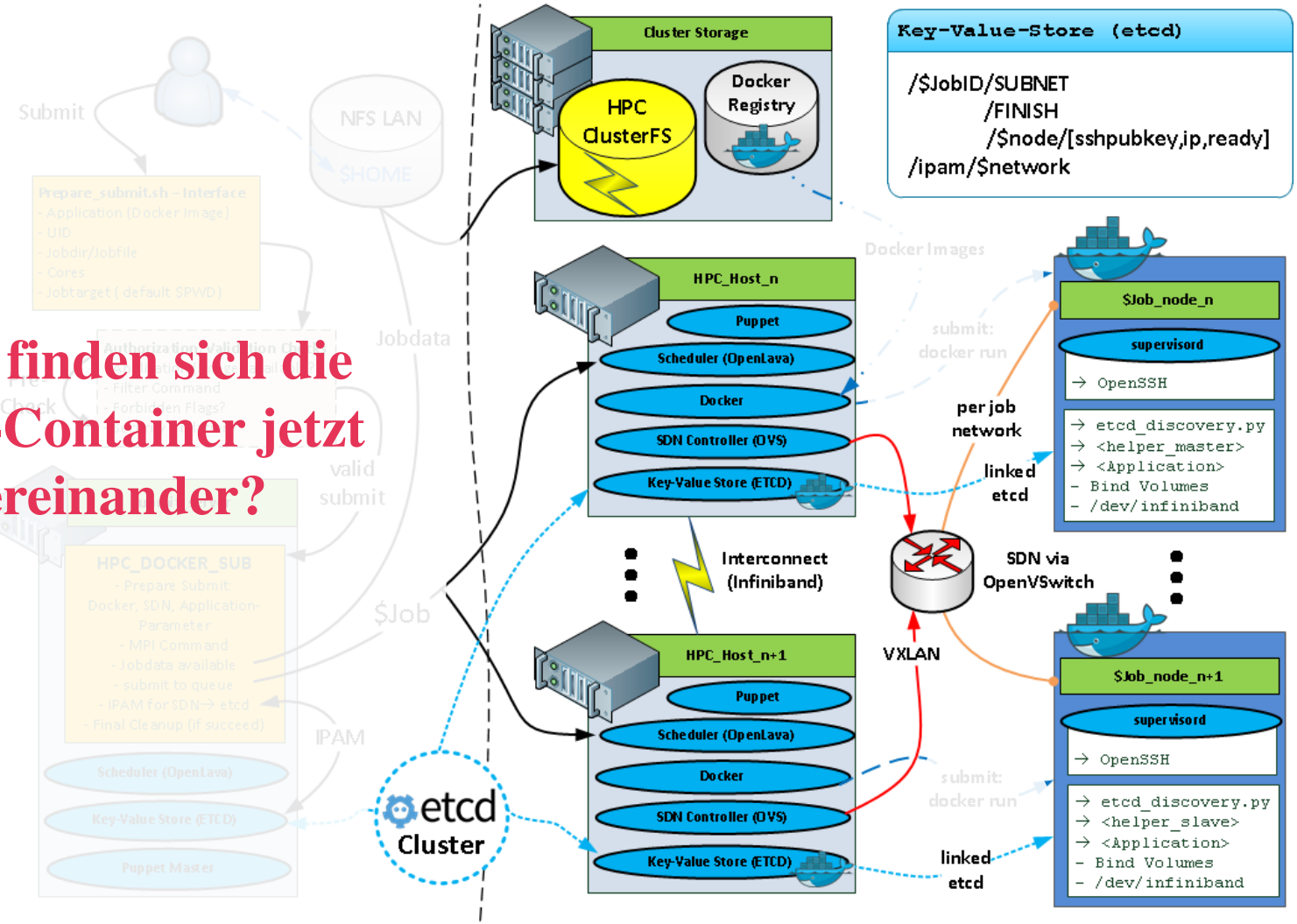
science + computing
an atos company

- Jeder Jobcontainer kennt JobID, seinen Hostname, IP und die Anzahl der Teilnehmer (falls verteilt)
- Middleware prüft Jobstatus über Meldungen der Docker Container am KVS:
  - Abbruch bei Ablauf TTL --> z.B. Job WallClock
  - Exitcode
- Queueing System prüft auf den Status des ausgeführten Middleware-Skript

5. Autonome Koordinierung

science + computing
an atos company

Wie finden sich die Job-Container jetzt untereinander?

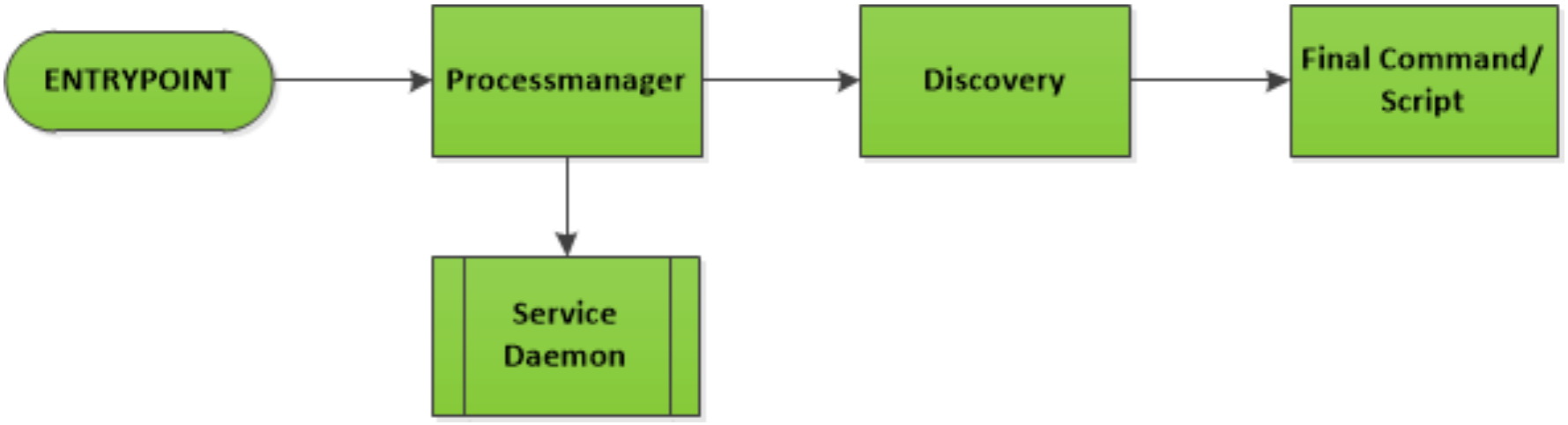


Einschub:
Lebenszeit eines Docker Containers

science + computing
an atos company

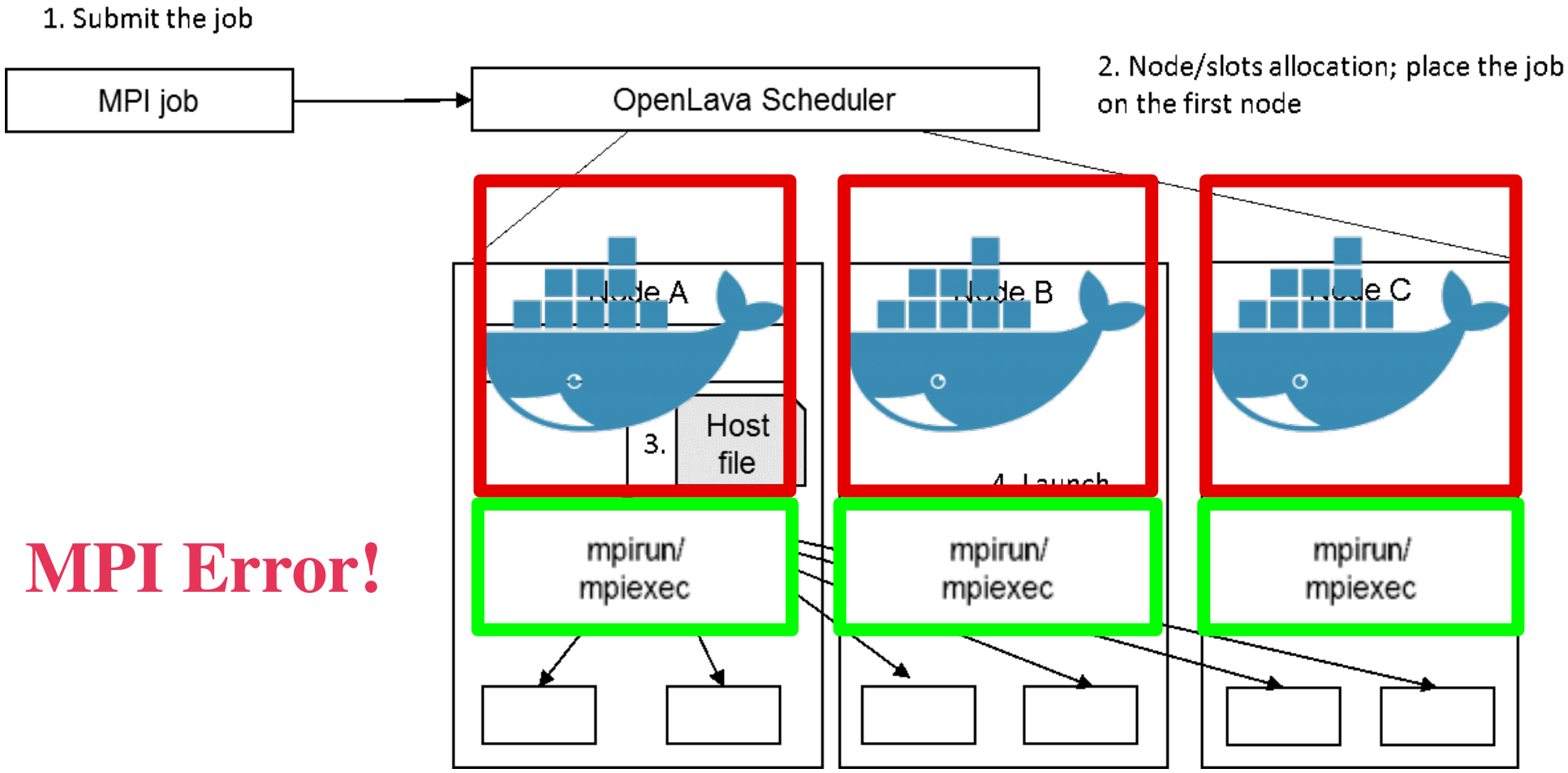
Container Lebenszeit ist an einen Init-Prozess gekoppelt!

- Service Deamons bei Docker im Vordergrund
- Prozessmanager für benötigte Services verwenden (z.B. sshd) --> Supervisord empfohlen



Problem: Parallelrechnen mit Docker

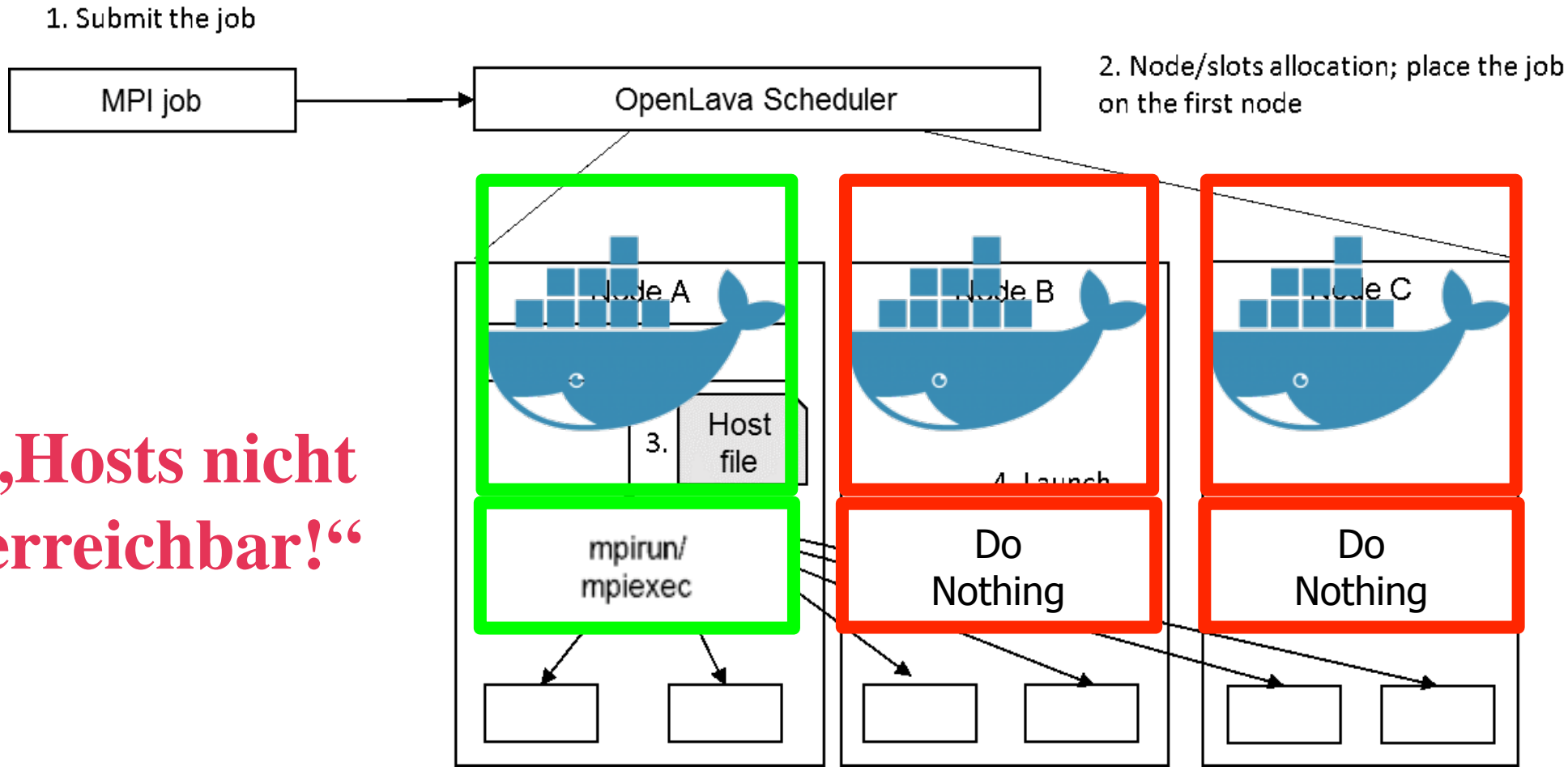
science + computing
an atos company



MPI Error!

Quelle: http://www.teraproc.com/teraproc-blog/mpi/

Problem: Parallelrechnen mit Docker



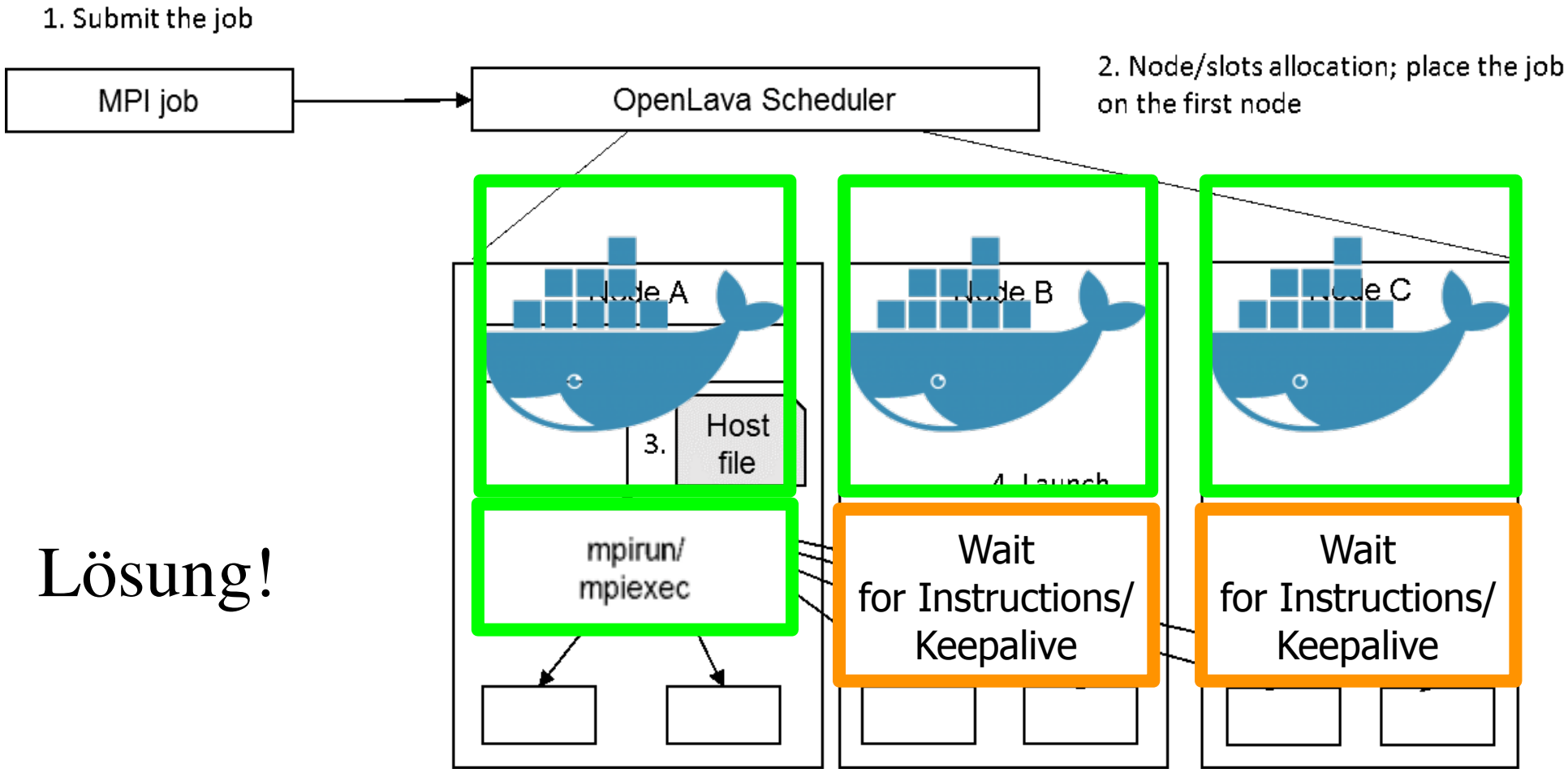
„Hosts nicht erreichbar!“

(Container beenden sich)

Quelle: http://www.teraproc.com/teraproc-blog/mpi/

Problem: Parallelrechnen mit Docker

science + computing
an atos company



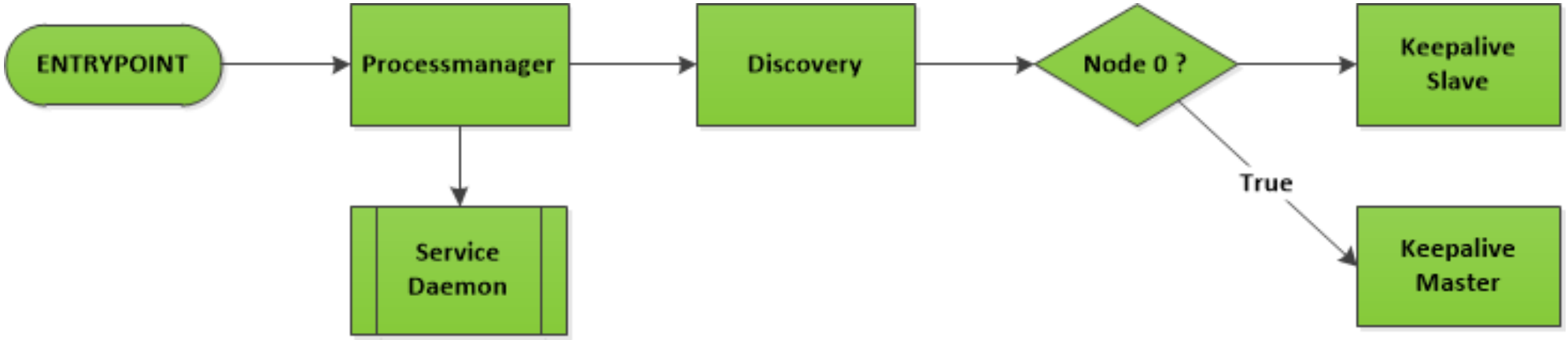
Lösung!

Quelle: http://www.teraproc.com/teraproc-blog/mpi/

5. Autonome Koordinierung: Ablauf Containerprozess I

science + computing
an atos company

- Master/Slave Konzept für autonome Container:
  - <job>_node_0 —> Master
  - <job>_node_n —> Slave
  - Kommunikation über KVS



5. Autonome Koordinierung: Ablauf Containerprozess II

science + computing
an atos company

1. Fester Start in Infrastruktur Skript (ENTRYPOINT)
2. sshd start über supervisord; Generierung SSH-Schlüssel
3. Jobdocker teilt dem KVS seine Eigenschaften mit:
   - `/$JobID/$node/[IP,SSHPUBKEY,READY]`
   - Ready = False
4. Container kontaktiert den KVS fordert IP & SSHPUBKEY aller anderen Teilnehmer an --> „wait-for-key“

5. Autonome Koordinierung: Ablauf Containerprozess III

science + computing
an atos company

5. Warten auf Empfang der Daten:
    1. Teilnehmer IP:
        - Pingtest
        - Eintragen in die /etc/hosts
    2. Teilnehmer SSH Schlüssels:
    - SSH Test (+KnownHosts)
    - Eintragen Authorized Keys
6. Alle Informationen erhalten und verifiziert?:
   --> Ready = True
7. Abfragen nach READY=True aller Teilnehmer

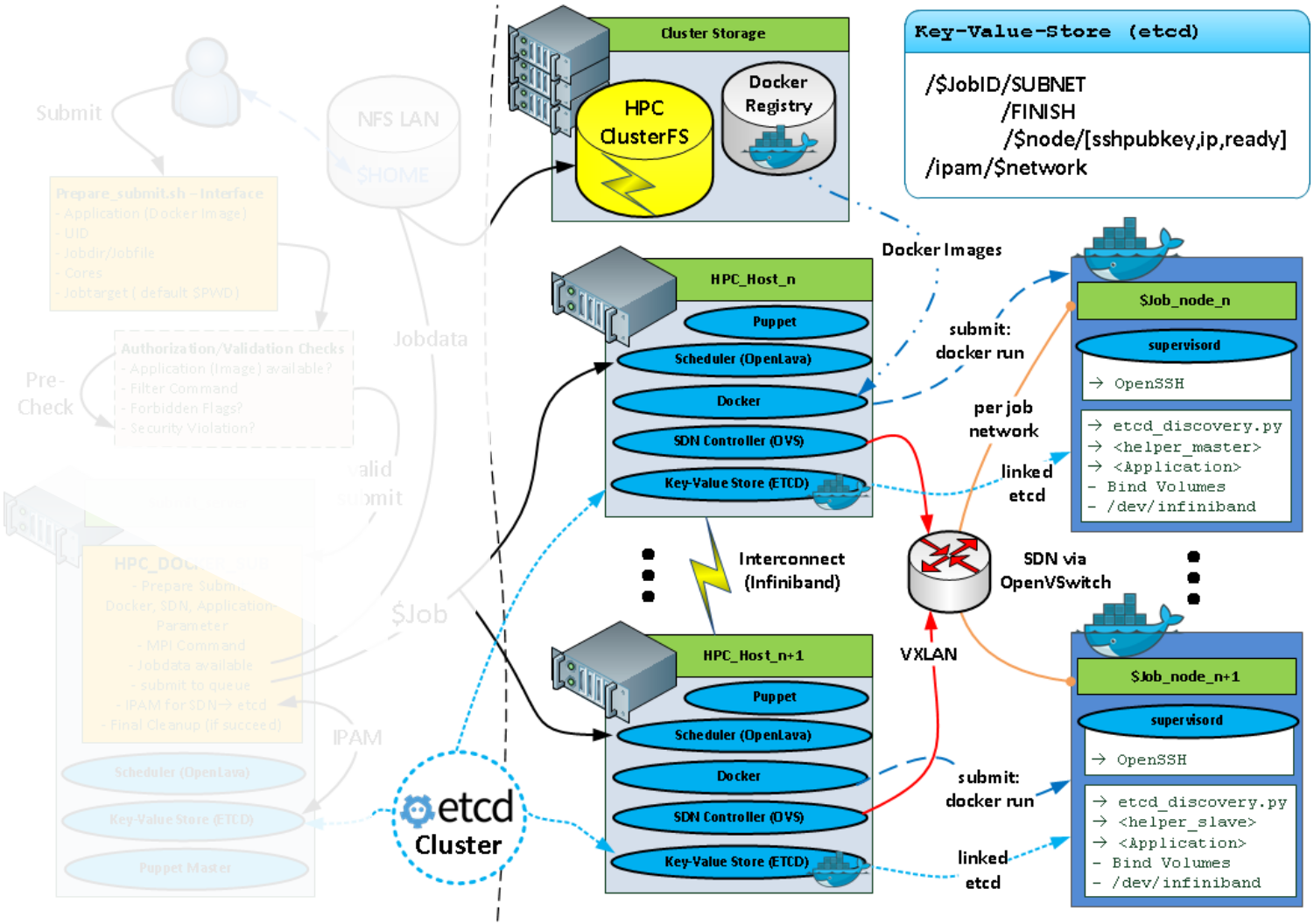
5. Autonome Koordinierung: Ablauf Containerprozess IV

science + computing
an atos company

8. Ist bei allen Teilnehmer READY=True gesetzt ist die „Discovery Phase“ abgeschlossen
9. Rollenentscheidung:
   - <JobID>_node_0 = Master
   - <JobID>_node_n = Slave
10. Master Container führt (MPI) Commando aus
    - Slave wartet auf Anweisung vom Master

6. Container-basierte Parallele Rechnung

science + computing
an atos company



6. Container-basierte Parallele Rechnung

science + computing
an atos company

- Master Container koordiniert die Berechnung via MPI an die Slaves --> Verwenden der Interconnects!
- Zugriff auf die Jobdaten über Cluster Filesystem
  - `docker run -v /lustre/scratch:/lustre/scratch ...`
- Interconnect Zugriff über Kernel Module
  - `docker run -v /dev/infiniband:/dev/infiniband ...`

```
1  [|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||100.0%]     Tasks: 124, 616 thr; 5 running
2  [|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||100.0%]     Load average: 6.09 7.25 5.72
3  [|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||100.0%]     Uptime: 25 days, 15:57:49
4  [|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||100.0%]
Mem[|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||19029/32014MB]
Swp[||                                                                                              241/19999MB]

  PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
12979 root       20   0 2211M 37548 29184 R 96.4  0.1  0:27.01 simpleFoam -parallel
12982 root       20   0 2211M 37584 29192 R 94.1  0.1  0:26.28 simpleFoam -parallel
12981 root       20   0 2211M 37744 29168 R 89.4  0.1  0:26.19 simpleFoam -parallel
12980 root       20   0 2211M 37688 29272 R 87.1  0.1  0:26.19 simpleFoam -parallel
```

6. Container-basierte Parallele Rechnung

science + computing
an atos company

```
2015-10-07 08:26:06.247285  PING TO "10.20.30.1" AS "99_node_0" SUCCESSFUL
2015-10-07 08:26:06.248212  SUBTHREAD REQUEST: "99_node_0" --> "99/99_node_0/IP" VALID VALUE RECEI
VED
2015-10-07 08:26:06.248344  RECEIVED "4" VALUES FOR ALL "2" PARTICIPATING COMPUTE NODES
2015-10-07 08:26:06.248947  CHECK SSH CONNECTION TO "99_node_0"
2015-10-07 08:26:06.249247  CHECK SSH CONNECTION TO "99_node_1"
2015-10-07 08:26:06.250489  WAIT TO RECEIVE AN "READY" FROM EVERYBODY
2015-10-07 08:26:06.298464  WAIT FOR READY STATE ON "0" NODES, BY NAME:
99_node_0,
99_node_1,
2015-10-07 08:26:06.484808  SSHTEST EXCEPTION:  [Errno 111] Connection refused
2015-10-07 08:26:08.486106  CHECK SSH CONNECTION TO "99_node_1"
2015-10-07 08:26:08.487687  SSHTEST EXCEPTION:  [Errno 111] Connection refused
2015-10-07 08:26:10.247916  SSH CONNECTION TEST TO "10.20.30.1" AS "99_node_0" SUCCESSFUL
2015-10-07 08:26:10.489945  CHECK SSH CONNECTION TO "99_node_1"
2015-10-07 08:26:11.318156  WAIT FOR READY STATE ON "0" NODES, BY NAME:
99_node_0,
99_node_1,
2015-10-07 08:26:12.770512  NO VALID READY STATE FOR "99_node_1" --> "False"
2015-10-07 08:26:12.822670  NO VALID READY STATE FOR "99_node_0" --> "False"
2015-10-07 08:26:14.292069  NODE "99_node_0" IS READY ! --> "True"
2015-10-07 08:26:14.292101  SUBTHREAD REQUEST: "99_node_0" --> "99/99_node_0/READY" VALID VALUE RE
CEIVED
2015-10-07 08:26:14.368772  SSH CONNECTION TEST TO "10.20.30.2" AS "99_node_1" SUCCESSFUL
2015-10-07 08:26:14.369025  ETCD_SET -> 99/99_node_1/READY =
True
2015-10-07 08:26:14.380282  VERIFIED "4" VALUES FOR ALL "2" PARTICIPATING COMPUTE NODES
2015-10-07 08:26:14.382556  NODE "99_node_1" IS READY ! --> "True"
2015-10-07 08:26:14.382581  SUBTHREAD REQUEST: "99_node_1" --> "99/99_node_1/READY" VALID VALUE RE
CEIVED
2015-10-07 08:26:16.323437  RECEIVED AN READY STATE FOR ALL NODES !

2015-10-07 08:26:16.323671  ALL THREADS FINISHED!
2015-10-07 08:26:16,336 etcd-deamon done, environment should be ready and all nodes reachable by I
P/DNS/SSH
2015-10-07 08:26:16,340 <<< SLAVE: going into slave mode and wait job to end >>>
```

```
ExecutionTime = 125.44 s  ClockTime = 129 s

Time = 125

DILUPBiCG:  Solving for Ux, Initial residual = 0.00280333, Final residual = 6.12963e-05, No Iterations 2
DILUPBiCG:  Solving for Uy, Initial residual = 0.0107562, Final residual = 0.00107405, No Iterations 1
DICPCG:  Solving for p, Initial residual = 0.0485313, Final residual = 0.000476529, No Iterations 155
time step continuity errors : sum local = 0.00262967, global = -3.67257e-05, cumulative = 0.0208422
DILUPBiCG:  Solving for epsilon, Initial residual = 0.00413815, Final residual = 0.00035738, No Iterations 1
DILUPBiCG:  Solving for k, Initial residual = 0.00819351, Final residual = 0.000620499, No Iterations 1
ExecutionTime = 126.44 s  ClockTime = 130 s

Time = 126

DILUPBiCG:  Solving for Ux, Initial residual = 0.00278323, Final residual = 6.08281e-05, No Iterations 2
DILUPBiCG:  Solving for Uy, Initial residual = 0.0106768, Final residual = 0.00106639, No Iterations 1
DICPCG:  Solving for p, Initial residual = 0.04867, Final residual = 0.000481033, No Iterations 153
time step continuity errors : sum local = 0.00265665, global = -3.18223e-05, cumulative = 0.0208103
DILUPBiCG:  Solving for epsilon, Initial residual = 0.00409317, Final residual = 0.000351521, No Iterations 1
DILUPBiCG:  Solving for k, Initial residual = 0.00810753, Final residual = 0.000610143, No Iterations 1
ExecutionTime = 127.62 s  ClockTime = 131 s

Time = 127

DILUPBiCG:  Solving for Ux, Initial residual = 0.00276508, Final residual = 6.03878e-05, No Iterations 2
DILUPBiCG:  Solving for Uy, Initial residual = 0.0105884, Final residual = 0.00105276, No Iterations 1
DICPCG:  Solving for p, Initial residual = 0.0488238, Final residual = 0.000471606, No Iterations 153
time step continuity errors : sum local = 0.00260673, global = -3.98541e-05, cumulative = 0.0207705
DILUPBiCG:  Solving for epsilon, Initial residual = 0.00404836, Final residual = 0.000344494, No Iterations 1
DILUPBiCG:  Solving for k, Initial residual = 0.00802045, Final residual = 0.000596552, No Iterations 1
ExecutionTime = 128.43 s  ClockTime = 132 s

Time = 128

DILUPBiCG:  Solving for Ux, Initial residual = 0.00274749, Final residual = 5.99422e-05, No Iterations 2
DILUPBiCG:  Solving for Uy, Initial residual = 0.0105001, Final residual = 0.00103428, No Iterations 1
```

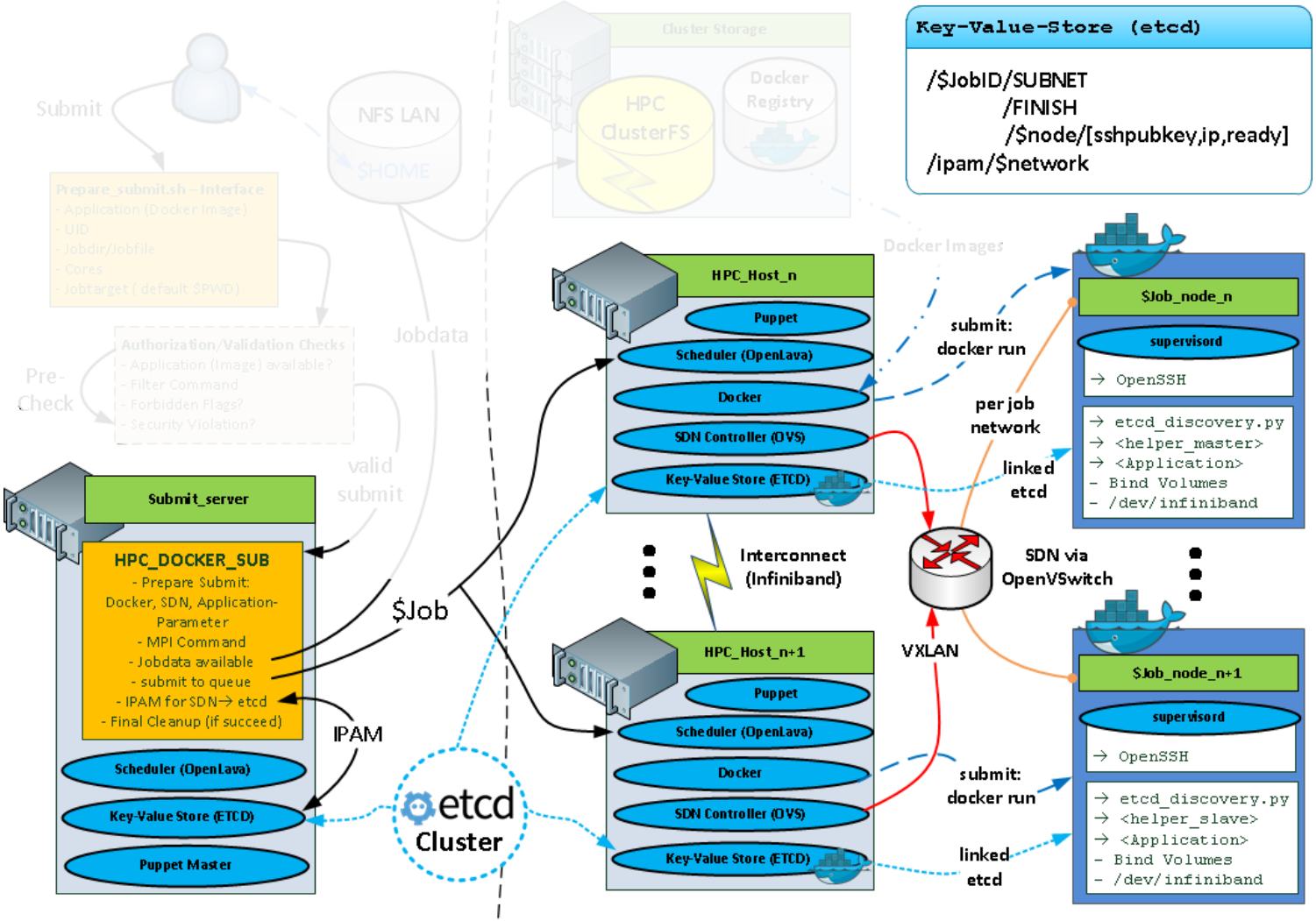
```
JOB=99
NODES=2
IFACE=eth1
FOAM_RUN=/mnt/nfs/OF_pitzDailyExptInlet
COMMAND=mpirun -v -hostfile system/machines -x PATH=$PATH -x LD_LIBRARY_PATH=$LD_LIBRARY_PATH -x WM_PROJECT_DIR=$WM_PRO
JECT_DIR  -x MPI_BUFFER_SIZE=2000000000 -np 4 -report-bindings -bind-to-core -mca btl_tcp_if_include eth1 simpleFoam -p
arallel
> Inintialize ETCD JobTree by pushing onto localhost:2379 --> /99/FINISHED = 'FALSE'
{"action":"set","node":{"key":"/99/FINISH","value":"FALSE","modifiedIndex":465,"createdIndex":465}}
> Starting '99_node_0' (MASTER) on root@10.1.25.138
Connecting Docker container 9be0cf7b1764a078932812fdf6849dd1addacd1c73a3cdc095f5636f5f85c9b7 to pod
RTNETLINK answers: File exists
------------------------------------------------------------
> Starting '99_node_1' (SLAVE) on root@10.1.25.131
Connecting Docker container 3e5994ed4d61bdcfa1ecc4094f192245e0d2d23fffcdda6636b8276dfc9c960c to pod
RTNETLINK answers: File exists
------------------------------------------------------------
------------------------------------------------------------
DONE!
bash: __git_ps1: command not found...
[root@testfedora-local cluster_openfoam]$
```

```
vkYgn1wPUNzheTVvKVtqQQwpb0Jwmfqhy0ci/UfdMxjUWn7WjK2cImDilAIY4Sy6whWEiZu/+C18GbbabDnQepd7gf4
Tsvg5R6CrWnale1R7+UZ6DDol1I5H/zMLozph0hz/oWsgVSdEPpz5dasUz95Yk6fFEBWTpOQS14ln2sJXKGXlaxGfPY
pXQJBAG1mtLtAvqD root@99_node_1
[root@f45f506254c6 /]# etcdctl get /99/99_node_0/SSHPUBKEY
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDtranPgFoPTQBwTKL885GcgP2gxf0nfSQvOHbTpBv/A7sHKFtDUUw
loHodAJ+o+GmioxdiHUUrkSuETR5rjSs/gWs9Oc/kbunhf3d3OQlszvuzg5tzlGIloPc+xws4knzJ09/J9UM1rn+0dk
TyUxc3pnS5727nLRZJnyUG62dCycVrqdKZNJW/NiFJ2WDRFxXTNIQZsCXKEmIKjhizNeymrKPGvRCeyvflMttDLgd8z
jjE6Ved1ADM0iGpCupze+OpxXFl4diItMeMsxAZWQMOznr2r8dO2k51ahcIzxOltRo8m2x9kNRvIYQq6TBdePfK3dRL
XuYhLhhVqGTzesEN root@99_node_0
[root@f45f506254c6 /]# etcdctl ls --recursive
/99
/99/FINISH
/99/99_node_0
/99/99_node_0/IP
/99/99_node_0/READY
/99/99_node_0/SSHPUBKEY
/99/99_node_1
/99/99_node_1/SSHPUBKEY
/99/99_node_1/IP
/99/99_node_1/READY
[root@f45f506254c6 /]#
```

```
<<<skling@subra>>>  1:subra  2:skling@testfedora-local:/mnt/nfs/cluster_openfoam* 3:root@centos7_1:~- 4:skling@testfedora-local:/mnt/nfs/cluster_openfoam  5:NFS-HQ  6:skling@testfedora> workspace Mi 07 Okt 10:28
```

7. Jobende

science + computing
an atos company



7. Jobende

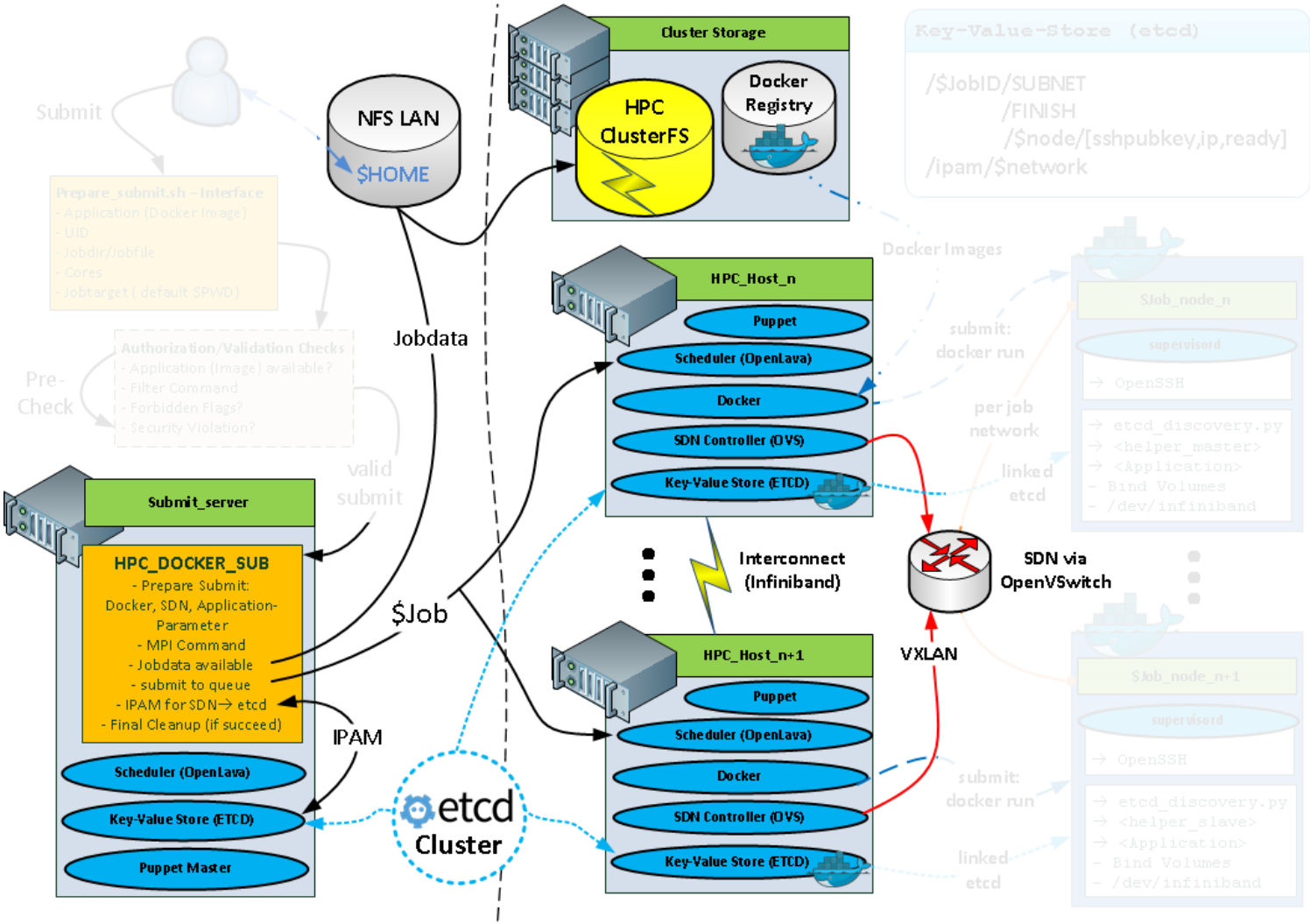
science + computing
an atos company

- Master signalisiert den Jobstatus im KVS Job Verzeichnis:
  - /$JobID/FINISH
  - Slaves beenden sich selbstständig
  - Middleware erhält den Exitcode
  - Queueingsystem sieht Exitcode der Middleware

- Bei Problemen des Rechenjobs?
  - TTL bei Schlüssel /$JobID/FINISH
  - Kontrolliertes Beenden der Jobumgebung

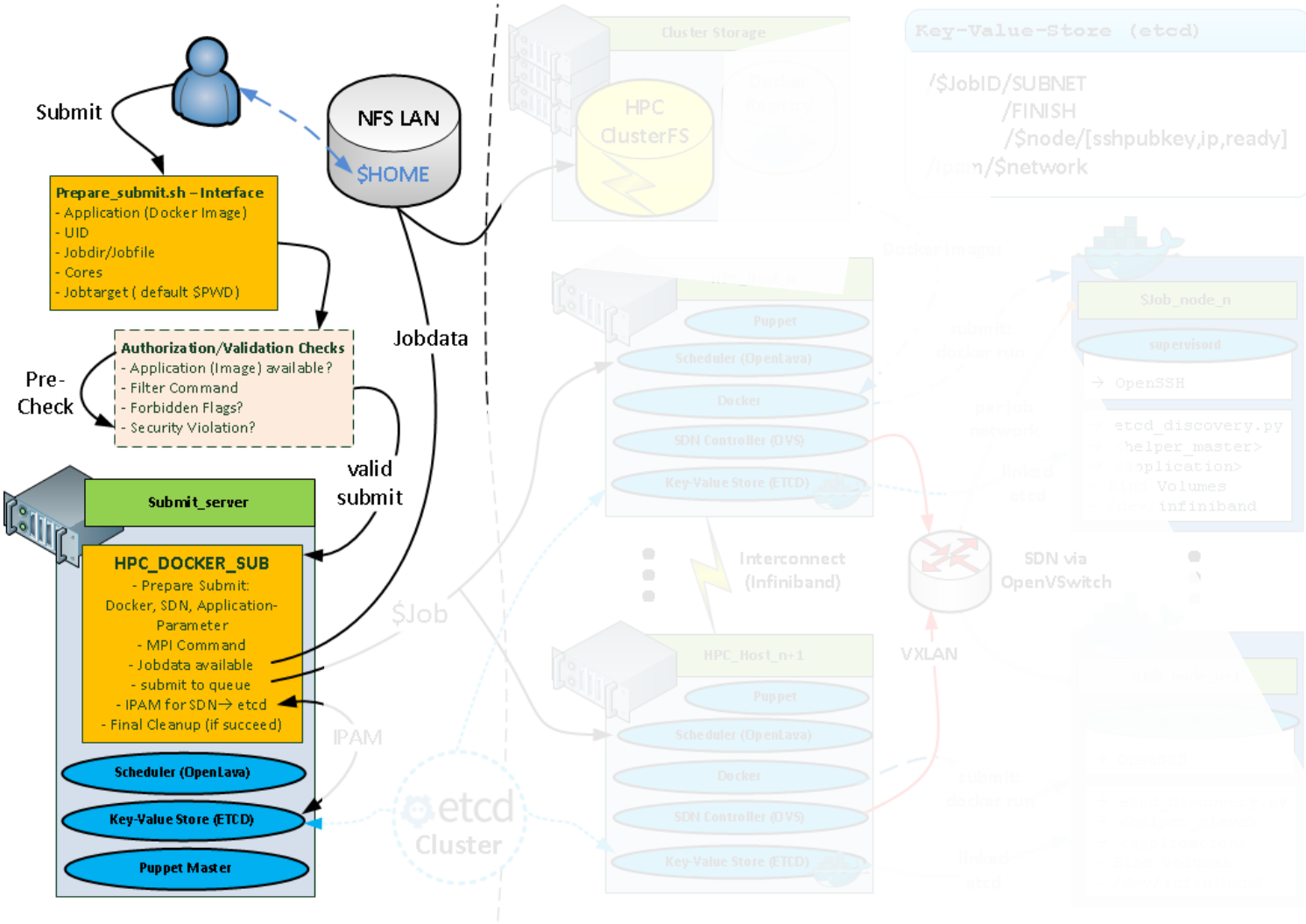
8. Bereitstellen der Ergebnisse & Abbau der Container Job Umgebung

science + computing
an atos company



9. Jobinfo an den Anwender

science + computing
an atos company



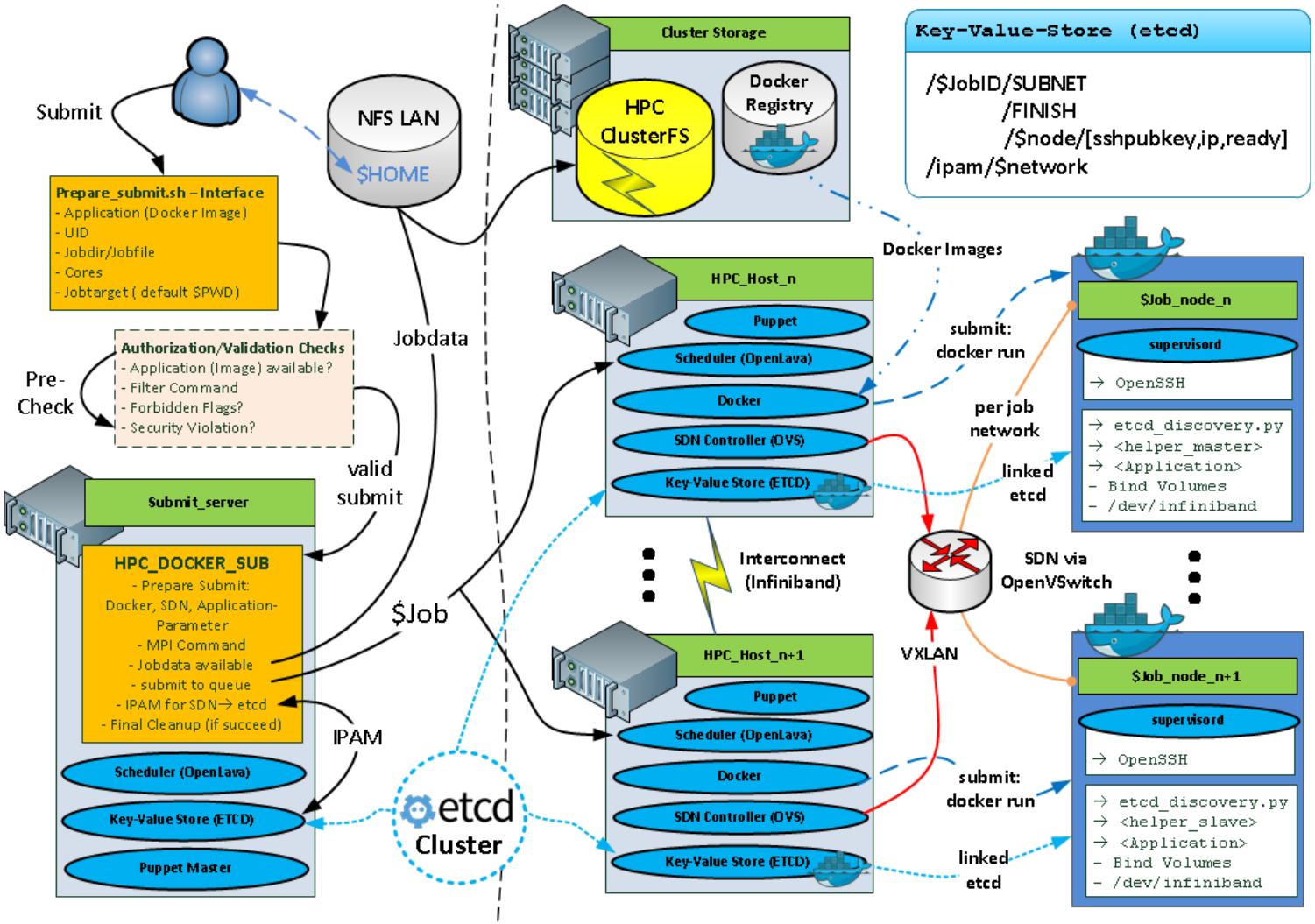
Was wird da überhaupt so gerechnet?

science + computing
an atos company

- z.B. Strömungssimulationen in einer Turbine

Fertiges Konzept

science + computing
an atos company



Fazit

science + computing
an atos company

- (Docker) Container enthalten großes Potential im Computing Bereich
- Verteiltes Rechnen recht kompliziert in klassischen HPC Umgebungen
- Kubernetes und Co. sind bei HPC eher ungeeignet!
- Weitere Entwicklungen und Konzepte verfolgen
  - Keine wirkliche „Komplettlösung“ vorhanden
  - Interessante Ansätze existieren
    - z.b. „Shifter“ (https://github.com/NERSC/shifter)
  - Immer noch nicht ganz ausgereift

... alles nicht so einfach

science + computing
an atos company

Quelle: http://devquixote.com/assets/images/mordor_meme.jpg

Vielen Dank für ihre/eure Aufmerksamkeit!

science + computing
an atos company

FRAGEN, DISKUSSION, ENDE

... es folgen noch einige Folien als Anhang, da diese leider aus zeitlichen Gründen nicht besprochen werden konnten, aber ich deren Inhalt doch nicht vorenthalten möchte.

Einige Überlegungen vor einer HPC Docker Integration

science + computing
an atos company

- Realisierung verteilte Jobs
- Interaktive Jobs
- Middleware zur Statusmeldung an Queueing System
- Aktualisierung Lokaler Docker Images
- Socket Problematik im Docker
- Sicherheitsprobleme:
  - Wie starten Anwender Docker
  - Sichere Private Docker Registry
  - Richtlinie Docker Imageformat
- ...

Wieso nochmal Docker im HPC? (1/2)

science + computing
| an atos company

- (Kaum) Performanceverlust da direkt unter dem Kernel
  - Direkter Hardwarezugriff (Interconnects, Grafik, …)
- Applikationen in Docker Images vorbereiten & verteilen
- Umgebung inkl. aller Libs/Bins/Co. im Image
- Linux namespaces, cgroups, selinux per default
- Versionierung aktiver Applikationen
- Sparsames UnionFS

Wieso nochmal Docker im HPC? (2/2)

science + computing
an atos company

- Versionskontrolle über Commit/Tag Möglichkeiten
- Nachvollziehbar/Reproduzierbar —> Dockerfiles
- Archivierung/Backups über Image-Tarballs möglich
- (Docker-)Host wird unabhängig von der Anwendung
  —> Bedingung: Kernel-Kompatibilität
- Docker-Registry: Verteilung, Verwaltung, Auth.
- …
- In ferner Zukunft: Hersteller liefert Software als fertiges Image aus

Meinung zur Docker Performance

science + computing
an atos company

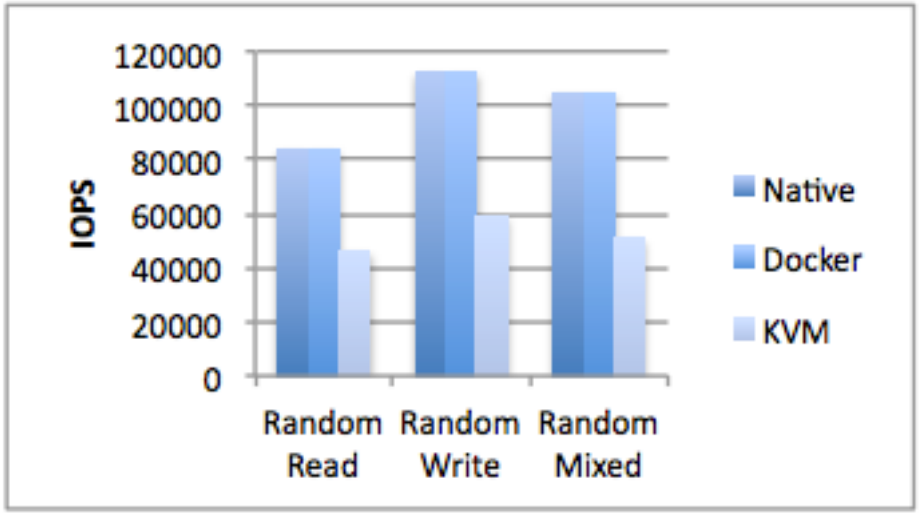
KEIN Verlust bei Rechenleistung, Netz-I/O wird beeinträchtigt beim Einsatz von NAT

Workload		Native	Docker	KVM-untuned	KVM-tuned
PXZ (MB/s)		76.2 [±0.93]	73.5 (-4%) [±0.64]	59.2 (-22%) [±1.88]	62.2 (-18%) [±1.33]
Linpack (GFLOPS)		290.8 [±1.13]	290.9 (-0%) [±0.98]	241.3 (-17%) [±1.18]	284.2 (-2%) [±1.45]
RandomAccess (GUPS)		0.0126 [±0.00029]	0.0124 (-2%) [±0.00044]	0.0125 (-1%) [±0.00032]	Tuned run not warranted
Stream (GB/s)	Add	45.8 [±0.21]	45.6 (-0%) [±0.55]	45.0 (-2%) [±0.19]	
	Copy	41.3 [±0.06]	41.2 (-0%) [±0.08]	40.1 (-3%) [±0.21]	
	Scale	41.2 [±0.08]	41.2 (-0%) [±0.06]	40.0 (-3%) [±0.15]	
	Triad	45.6 [±0.12]	45.6 (-0%) [±0.49]	45.0 (-1%) [±0.20]	

- W.Felter: „ An updated performance comparison of virtual machines and Linux containers“ [link]
- M.Xavier: „Performance Evaluation of Container-based Virtualization for High Performance Computing Environments“ [link]
- …

Quelle Benchmark: IBM Research Report, s.o.

Docker Performance: IBM Research Report [link]



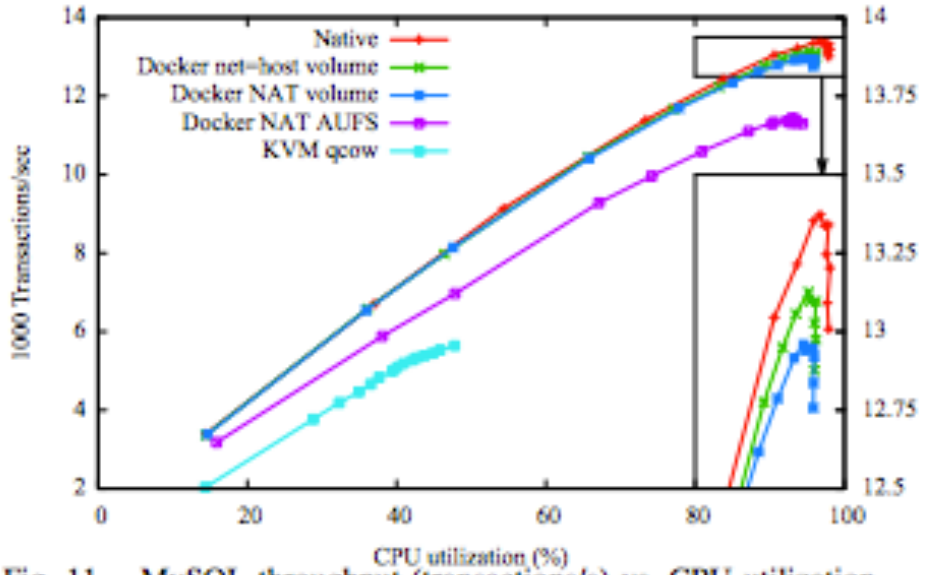
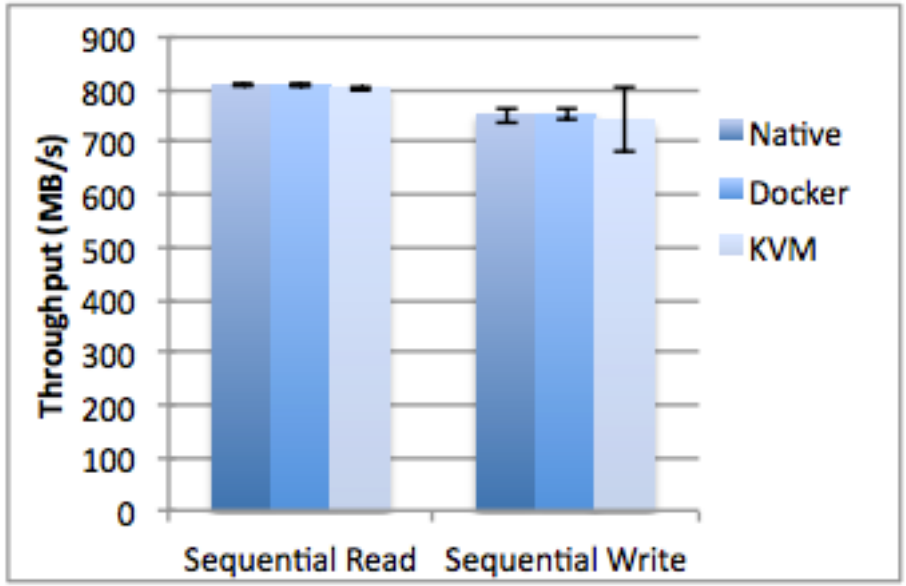
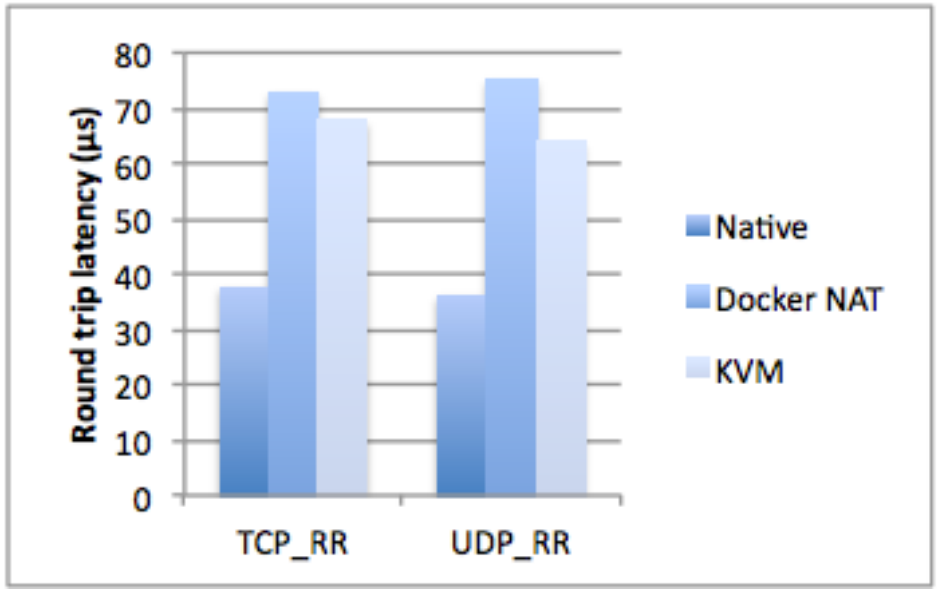


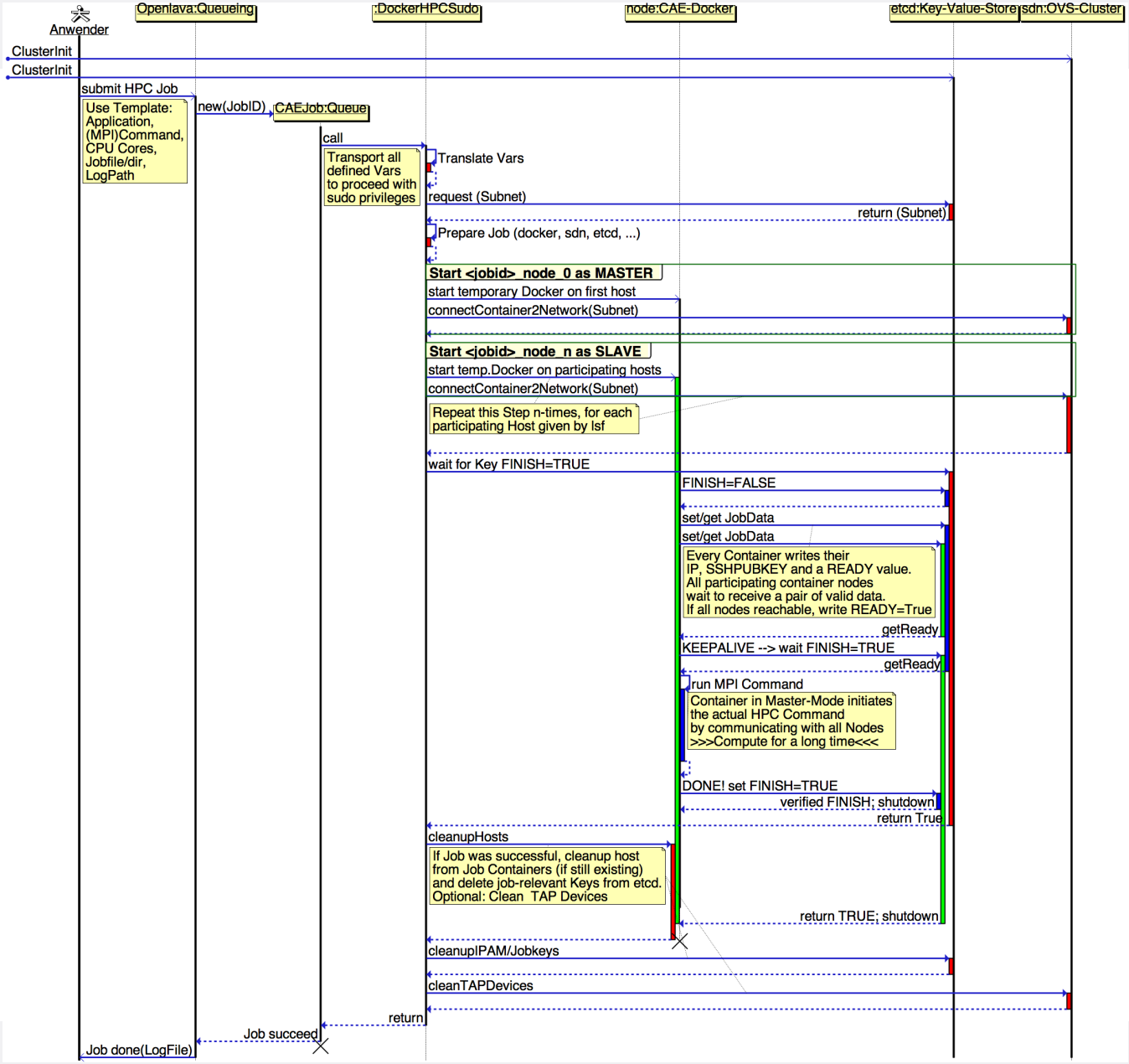
Fig. 11. MySQL throughput (transactions/s) vs. CPU utilization.





Quelle: W.Felter: „ An updated performance comparison of virtual machines and Linux containers“ [link]

Sequenzdiagramm zum PoC



Private Registry: Docker Images beziehen und verwalten

science + computing
an atos company

- Jeder Dockerhost hat ein lokales Repo (/var/lib/docker)
- Login bei privater Docker Registry auf dem Headnode
- Aktualisieren: manuell, Cronjob (oder „Watchtower“)
- Registry:v2 kann UserAuth
  - Nachvollziehbare Administration der Images
  - ro/rw
- Docker Registry Cluster möglich
- Abkoppeln vom öffentlichen DockerHub! --> RedHat

```
docker daemon —add-registry=hpcregistry:5000 —block-registry=docker.io
```

Interaktive Jobs

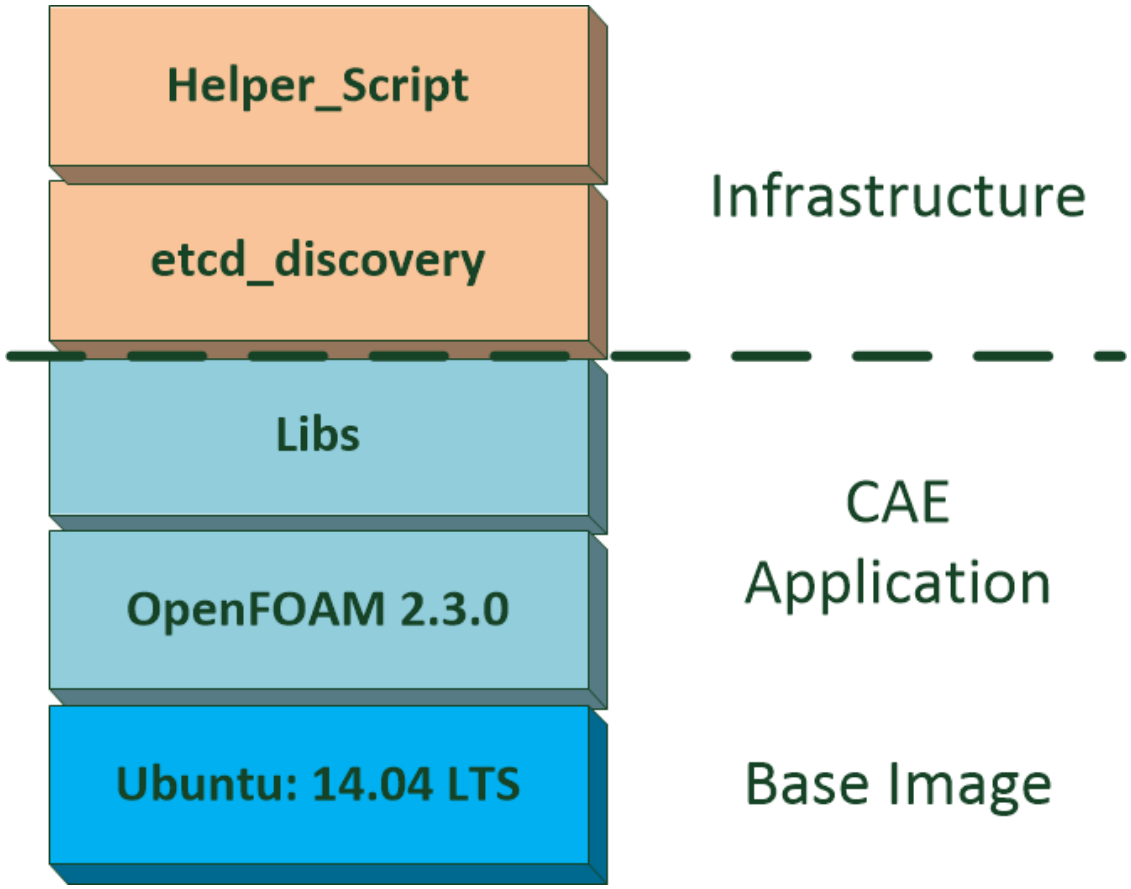
science + computing
an atos company

- Unschön:
  - SSH Verbindung zum HPC Container aufbauen und Output abgreifen
- ...schon besser:
  - „docker logs“ und „docker exec“ auf dem Docker Host arbeiten
- ...noch besser:
  - Dockers Logging Driver verwenden für den Output!
  - Input über Service in Docker regeln

Aufbau der HPC Images

science + computing
an atos company

- Richtlinie einführen!
  -> Logische Trennung nach Funktion
- Base + Applikation:
  - Libs / Bins / …
- Infrastruktur:
  1. Umgebung
  2. Services (optional)
  3. Discovery
  4. Startpunkt
     --> Discovery Skript



Problem: Netzwerk Infrastruktur

science + computing
an atos company

- Stichwort: Service Discovery
  - Container finden sich selbst über gemeinsamen Kommunikationspunkt —> Key-Value-Store
  - IPAM, DNS —> übernimmt hier Middleware
    - Einfach: /etc/hosts
    - Alternativ: Consul, etcd +skydns, ...

Hostnamespace nutzen

science + computing
an atos company

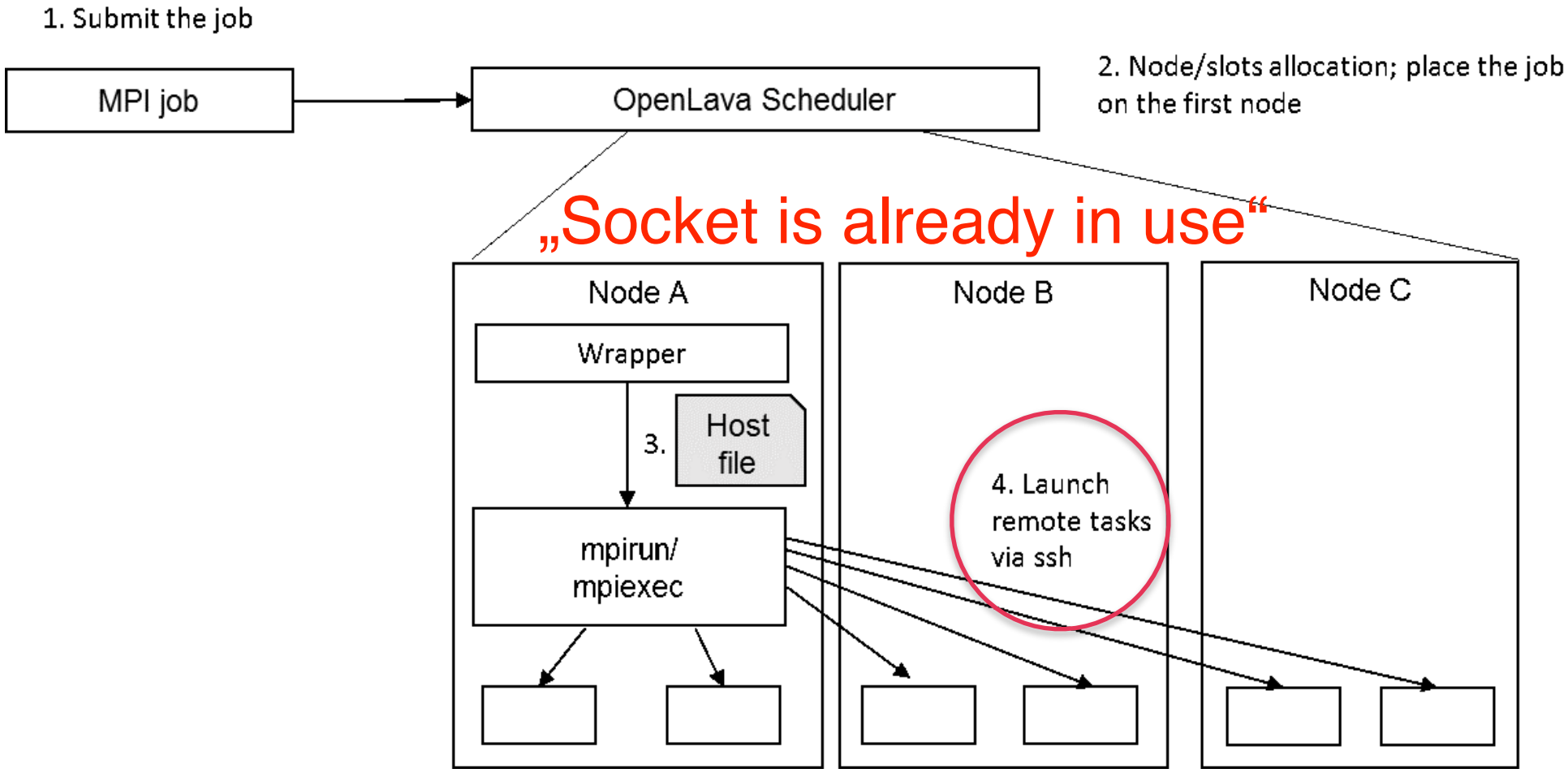
- Wieso überhaupt NET namespaces und nicht:

```
docker run -v /dev/infiniband:/dev/infiniband –net=host …
```

- --net nutzt den Namespace des Host:
  - selber Hostname
  - selbe Interfaces/IPs/MACs
  - IPoIB —> bei NET Namespace: Bridge Interface
  - …
  - Service Sockets im Container ändern
    - KVS Kommunikation

OpenMPI: Socket Problematik Container

science + computing
an atos company



Quelle: http://www.teraproc.com/teraproc-blog/mpi/

OpenMPI: Socket Problematik Container

science + computing
an atos company

- NAT
  - Pro: Freigabe einzelner Services
  - Contra: Kann Komplex werden, schlechte Performance, löst Socket Problematik nicht
- Network Stack des Host (NET-Namespace)
  - Pro: direkt, Beste Performance, keine weiteren Komponenten notwendig
  - Contra: SocketProblem bei >=1 Container auf dem Host
- Linux Bridges & virtuelle Switches
  - Pro: Flexibel, Gute Performance, NET-Namespaces
  - Contra: Zusätzliche Verwaltung, IPoIB Bridge

STATUS QUO:
DOCKER IN QUEUEING SYSTEMEN

Docker in LSF

science + computing
an atos company

- Erste Plugin Beta seit Anfang 2014

IBM Platform LSF Integration with Docker

Docker is an emerging container technology on the Linux Platform that allows users to develop, ship, and run applications almost everywhere. This open beta integrates Platform LSF with Docker to automatically start the Docker container on the execution host of an HPC cluster after users have submitted jobs with the Docker image. The integration honours LSF resource allocation (CPU, memory and affinity) and supports full job control, including job suspension, resumption and termination, and it can also collect run time container resource usage.

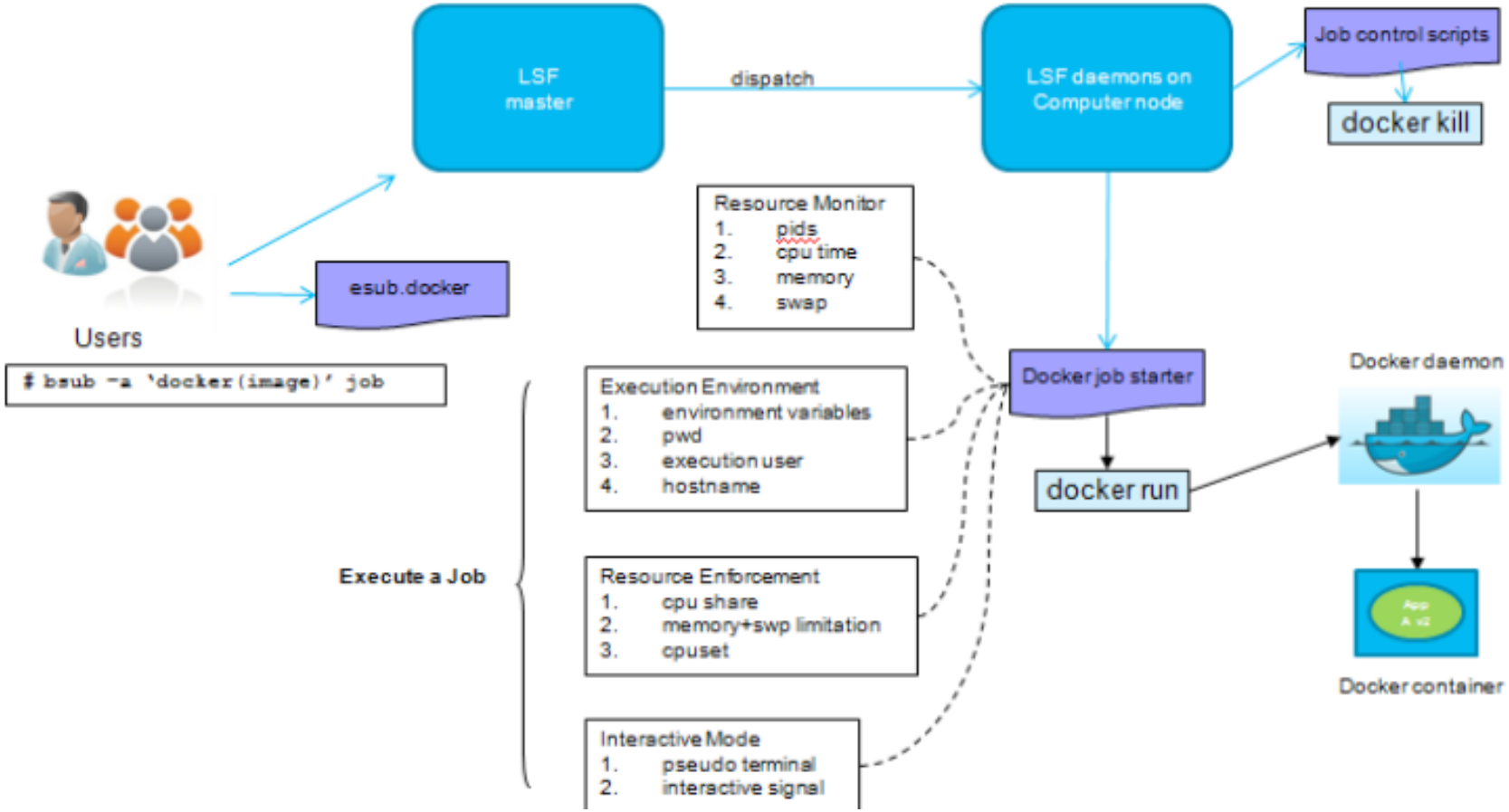
Try the beta build below. Want to know more? Contact ✉ Zhi Qiang Zhai to ask questions or to give your feedback.

Date	Type	Description	Download
1 Aug 2014	Beta	IBM Platform LSF Integration with Docker	Beta 2

Quelle: http://www.ibm.com/developerworks/servicemanagement/tc/plsf/downloads.html

How it works ?

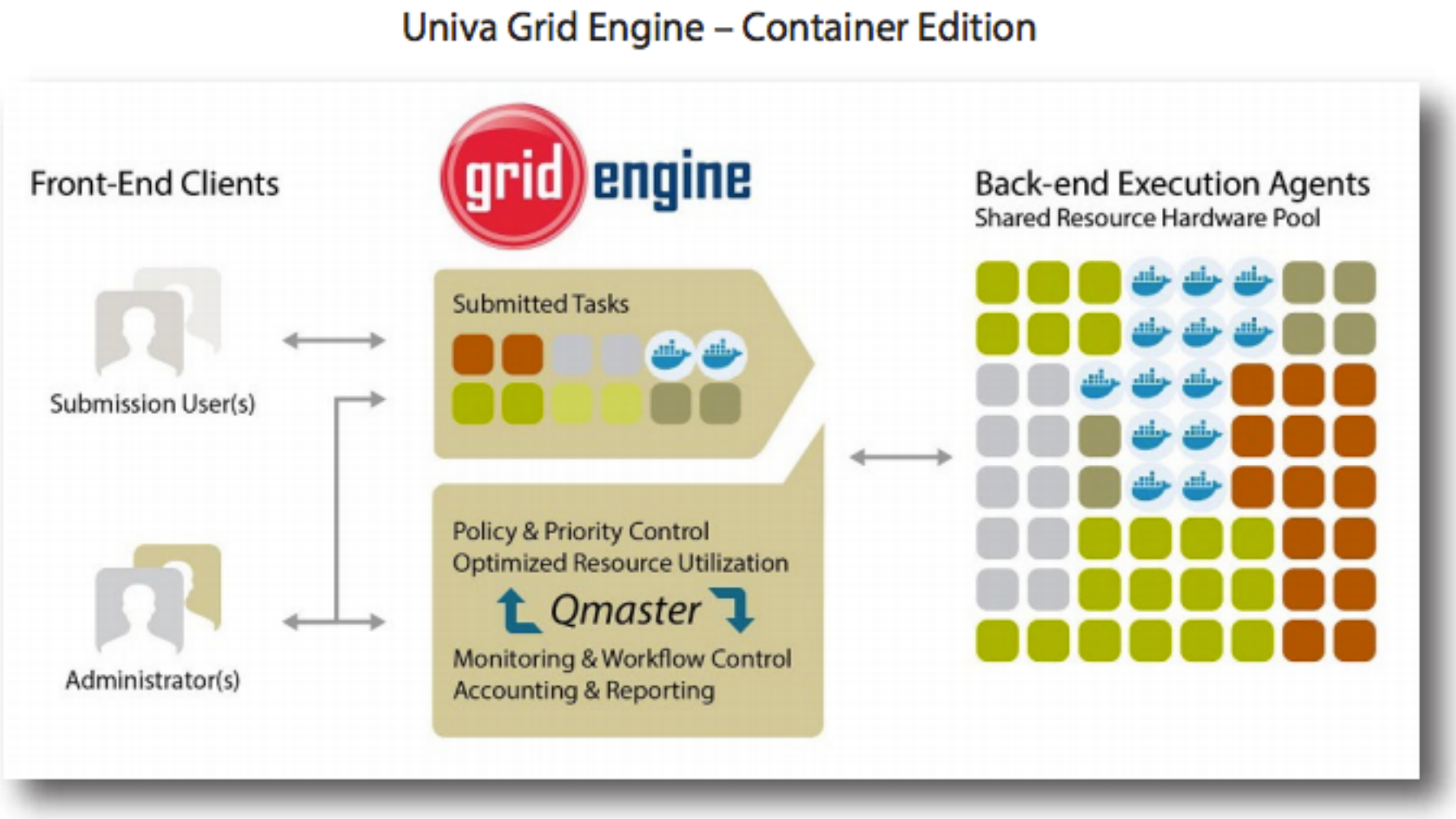
- Leverage existing IBM Platform LSF plugins mechanism (esub, job starter, job control etc)



Quelle: http://openpowerfoundation.org/wp-content/uploads/2015/03/Sanjabi-Sam_OPFS2015_IBM_v2-2.pdf

Docker in Univa GridEngine

science + computing
an atos company



Quelle: https://www.univa.com/resources/files/gridengine_container_edition.pdf

Docker in Univa GridEngine

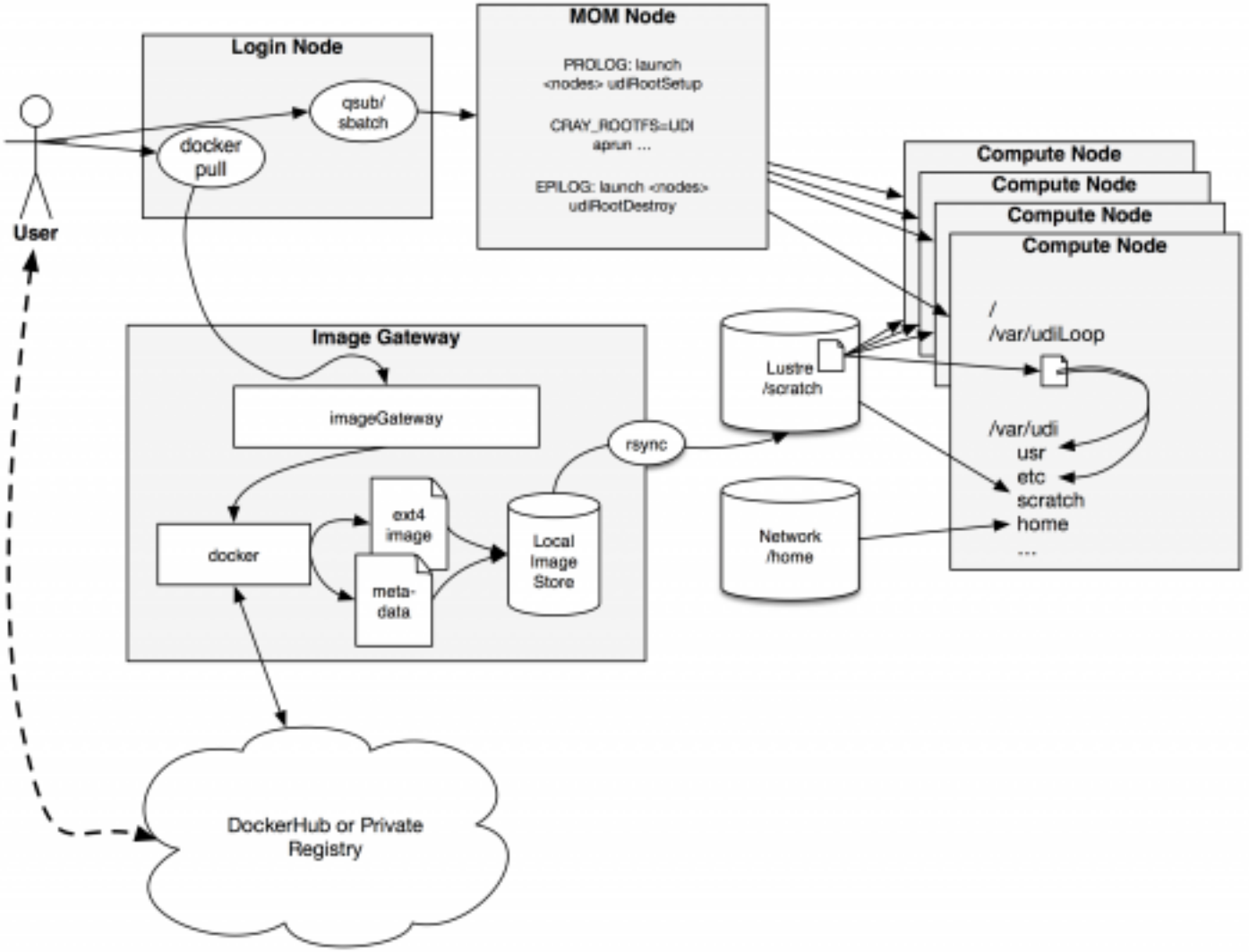
science + computing
an atos company

Additional Docker support includes:

- Docker Directory Mapping - maps user directories outside of a Docker container into a Docker container during a job run, making it easier to transfer files
- Full Job Control of Docker Containers - provides Grid Engine with the ability to suspend, resume or kill existing jobs running in containers
- Full Job Accounting for Docker Containers - provides accounting information for the job running in the Docker container
- Run Parallel Jobs in Docker Containers - Users can submit parallel jobs to Grid Engine that will run on multiple machines in Docker containers
- Automatic Handling - input, output and error files for Docker jobs
- Run Interactive applications in Docker containers - Grid Engine Interactive sessions can run inside a container

Quelle: https://www.univa.com/about/news/press_2016/05312016.php

(NESRC) Shifter: Docker in Slurm/Torque/Cray



Quelle: http://slurm.schedmd.com/SLUG15/shifter.pdf

(NESRC) Shifter: Docker in Slurm/Torque/Cray

science + computing
an atos company

„In a nutshell, Shifter allows an HPC system to efficiently and safely allow end-users to run a docker image. […]“

-GitHub https://github.com/NERSC/shifter/ -

- User-spezifisches Docker Image (UDI)
- Image Gateway
- Sicherheit:
  - udiRoot —> Rechte/Privilegien im Container
  - nur MNT Namespaces —> Loopback VFS
  - static-linked ssh im user-space*
  - kein Socket Problem da sshd und Host Networkstack
  - Filtern des Submitbefehls
- Prototyp anpassbar an verschiedene Queueing Systeme

* http://docs.cray.com/PDF/Shifter_User_Guide.pdf

Bewertung:
Integration von Docker in Queueing Systemen

science + computing
an atos company

- Container starten/stoppen/anhalten reicht nicht aus!
- Keinen direkten Docker Zugriff für Benutzer
  - Sicherheitsrisiko! —> Root-Problematik
  - Beispiel: Fehlende NFS Authentifizierung
  - Möglichkeiten: „docker-sudo-wrapper“
- Probleme bei verteilten, parallelen Rechenjobs auf MPI Basis
  - File Locking (?)
  - Verschiedene Socket Problematik
  - Isolierte Prozesse —> Kein Namespace bei /proc , /sys
  - Automounts unschön im Container

science + computing

an atos company

Vielen Dank für Ihre Aufmerksamkeit.

Sebastian Klingberg
science + computing ag
www.science-computing.de