A possible solution for HEP processing on network secluded Computing Nodes

International Symposium on Grids & Clouds 2021 (ISGC 2021)

Mirko Mariotti (UniPG and INFN Perugia)
Daniele Spiga (INFN Perugia)
Tommaso Boccali (INFN Pisa)

Outline

- The Problem
  - HPC centers net limitations
  - Known solutions ?
- Implementation
  - The overall idea
  - Tsocks
  - SSH
  - Tunsocks + openconnect
  - CVMFS exec
  - Singularity
- Tests
  - CVMFS mount
  - SRM transfer
  - HTCondor
  - Marconi
  - Galileo
- Conclusions
  - Conclusions

The Problem

The Problem: HPC Centers network limitations

Computing needs

The computing needs of LHC experiments in the next decades are expected to increase substantially.

The Payloads need to be delivered to a node and access **remote resources**:

- Access to conditions
- Access to SW
- Access to remote input data / stageout of output data

HPC Centers

Many Funding Agencies are aiming to a consolidation of the national LHC computing infrastructures, via a **merge** with other large scale computing activities, such as HPC and Cloud centers.

The Problem

Many technical issues has to be addressed to use those HPC centers with HEP payloads (*)

The biggest obstacle with some centers comes from stricter **network** policies with respect to our standard centers, which do not allow an easy merge with the distributed LHC computing infrastructure.

(*) Common challenges for HPC integration into LHC computing

Known solutions ?

- **Negotiate** with the site for at least **minimal connectivity** (as for CINECA: ok for CERN and CNAF)
- **Encapsulate network** on something else (for example BSC(*): via FS, but it needs a service by service approach...)
- Deploy **large containers** to avoid needing CVMFS/Conditions (but condor management traffic? Stageout? Input data?)
- **Create edge services** to transmit / encapsulate / translate accesses
  - A Squid is (also) such a thing
  - ATLAS Harvester (**) is (also) such a thing
  - An xrootd proxy is (also) such a thing
  - How many services do we need to bridge one by one? (xrootd, srm, webdav, condor connections, dashboard reporting, DBS, DAS, Frontier, voms connections …)
- We need a lower level solution, **below the application layer**

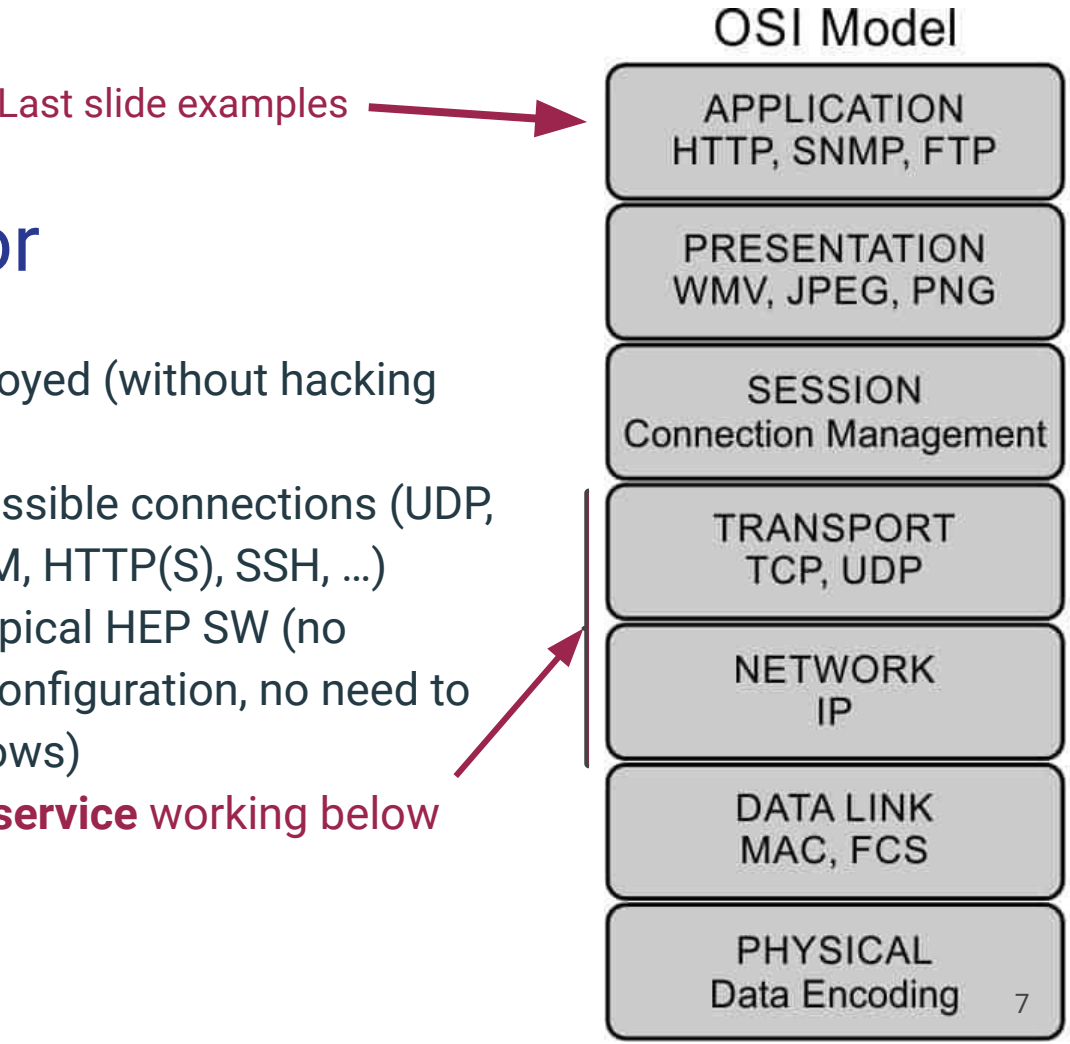
- Having routing by kernel to a NAT is clearly a solution (but it needs the site admins to agree/implement)
- Having a VPN from the site to “somewhere else” is a solution (but it needs the site admins to agree/implement)
- If the sysadmins agree with deploying the previous two, the problem is not existing…
- … but difficult to find the agreement in some (known and future) cases

*: Exploiting network restricted compute resources with HTCondor: a CMS experiment experience
**: Harvester : an edge service harvesting heterogeneous resources for ATLAS

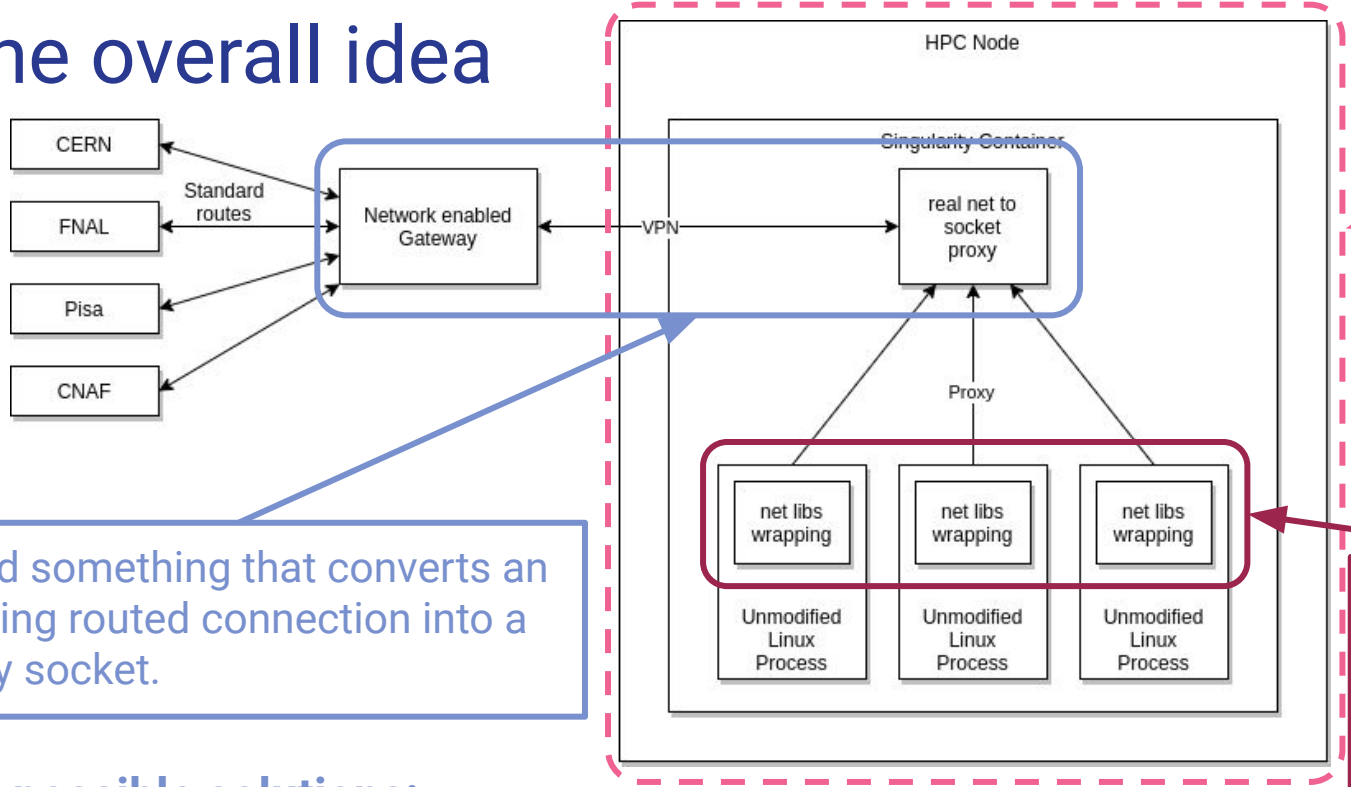
Implementation

The overall idea
We started looking for

- … A solution which can be deployed (without hacking the system) by a standard user
- … A solution covering all the possible connections (UDP, TCP) and services (XrootD, SRM, HTTP(S), SSH, …)
- … A solution not intrusive for typical HEP SW (no recompilation, no changes of configuration, no need to ask for special pilots or workflows)
- If you want, an **universal edge service** working below the application layer.



The overall idea



Isolated node

"You shall not pass"

… and something that converts an existing routed connection into a proxy socket.

Tsocks

After some research, we came up with the idea of wrapping (PRELOADING) the net calls to a local socket...

Two possible solutions:

1. **ssh -D**
2. **tunsocks + openconnect**

Tsocks

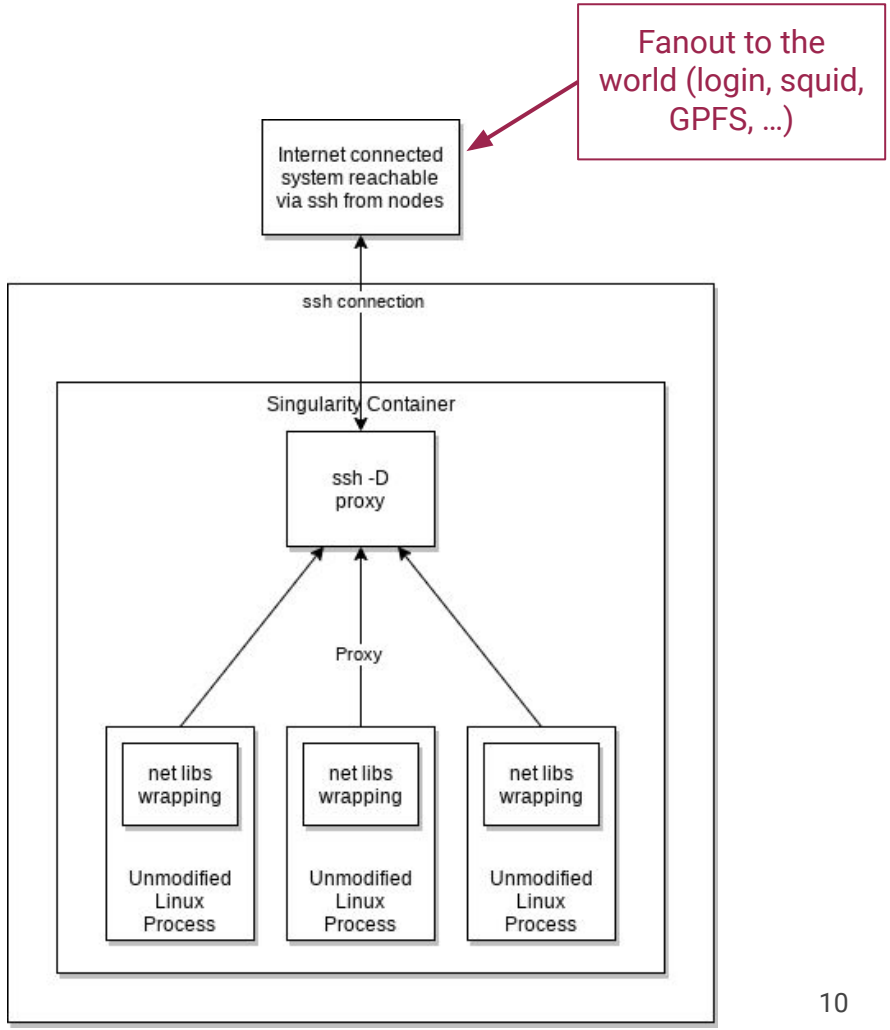
Tsocks is a transparent SOCKS proxying library. It permits the connection to a socket proxy via a preloaded shared library interceptor.

1. **Tsocks statically linked:** no compatibility issues with scientific environments
2. It was not supereasy: standard libtsocks codebase does not intercept all the needed network calls to work with Xrootd/SRM
   a. Great help from CERN (E.Sindrilaru, F.Furano, M.Simon, P.Paparrigopoulos, L.Mascetti) in preparing a suitable one
3. Tsocks encapsulation is smart: in **.tsocks.conf** you can have different configuration per routes (including "do nothing" for local connections)
   a. Even using different servers per different routes

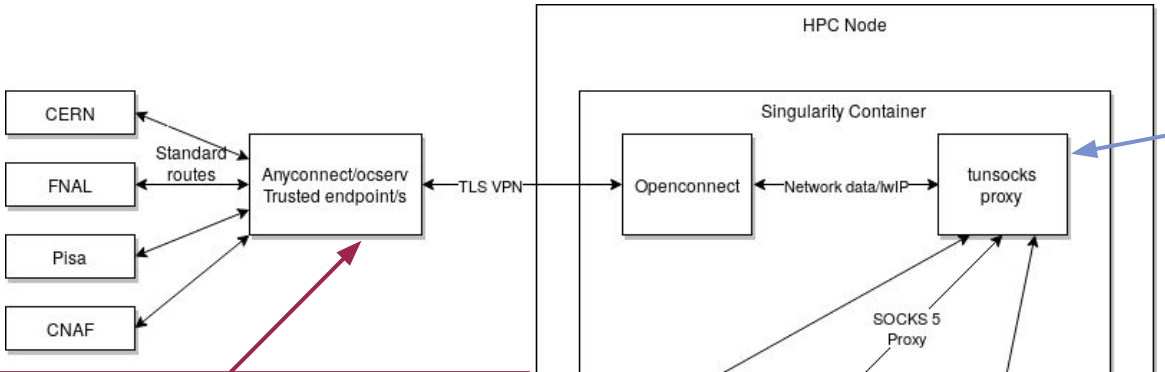
```
local = 192.168.0.0/255.255.255.0
local = 10.0.0.0/255.0.0.0
server = 192.168.0.1
server_port = 1080
path {
        reaches =
        150.0.0.0/255.255.0.0
        reaches =
        150.1.0.0:80/255.255.0.0
        server = 10.1.7.25
        server_type = 5
}
```

First option: ssh -D

- From the compute node, you need to be able to open a socket to an host which sees external connectivity (in most cases, the bastion/login node by definition can since you can go there from the login node)
  - Either ssh -D computenode → host
  - Or ssh l-D localhost on the host, and socket access from computenode
- Once you have that, you need to encapsulate (selected) network connections via the SOCKS
  - Since you cannot use kernel routing commands, use LD_PRELOAD to intercept network calls



Second option: tunsocks + openconnect



- An anyconnect/ocserver has to be installed/configured on the edge node.
- In the case of anyconnect, an array of endpoints able to fan out is deployable (for ex. At CNAF for us!)

- Openconnect client, exactly as ssh, does not require any superuser privilege
- Small complication: you need also tunsocks over tsocks
- In our case, CINECA has routing to CNAF (and CERN), but nothing else

An example of connect script

```
echo "some password" | /usr/sbin/openconnect --passwd-on-stdin --no-dtls -u [user] --script-tun --script "/usr/bin/tunsocks -D 5555"
[edge node ip]  --servercert pin-sha256:kgZNs8k8lbK0cI6IketuiITLcqRxpul1Xa+WD+l7fPY=
```

SSH vs Openconnect

Common traits:

1. The edge node can be physically elsewhere, if it is trusted by the site and reachable from the WN;
2. The edge nodes can be many, and if behind a DNS round robin, load is automatically balanced
3. Tsocks can be chained allowing multiple “jumps”

We are aware that a third approach based on using network namespaces and a user level VPN setup is under development by Ben Tovar at Notre Dame. They use openconnect+ocproxy+tsocks as a fallback when namespaces are not enabled

Openconnect + tunsock	SSH
Installation needed both on the edge node and con the compute one	Probably already installed and available on the edge node
Some configuration needed	Easier to configure
Easier to scale	Not scaling well
More powerful control of routes	Poor routes control
More powerful control of accounts	Managing of accounts complicated
Direct in hardware support (anyconnect devices)	

cvmfsexec

cvmfsexec is for mounting cvmfs as an unprivileged user (github)

The CernVM File System provide a universal way to distribute software among High Energy Physics (HEP) sites.

CernVM-FS is implemented as a POSIX read-only file system in user space (a FUSE module). Files and directories are hosted on standard web servers and mounted in the universal namespace /cvmfs.

Cvmfs require a privileged user to be used. With cvmfsexec the same functionality can be brought to standard users file spaces.

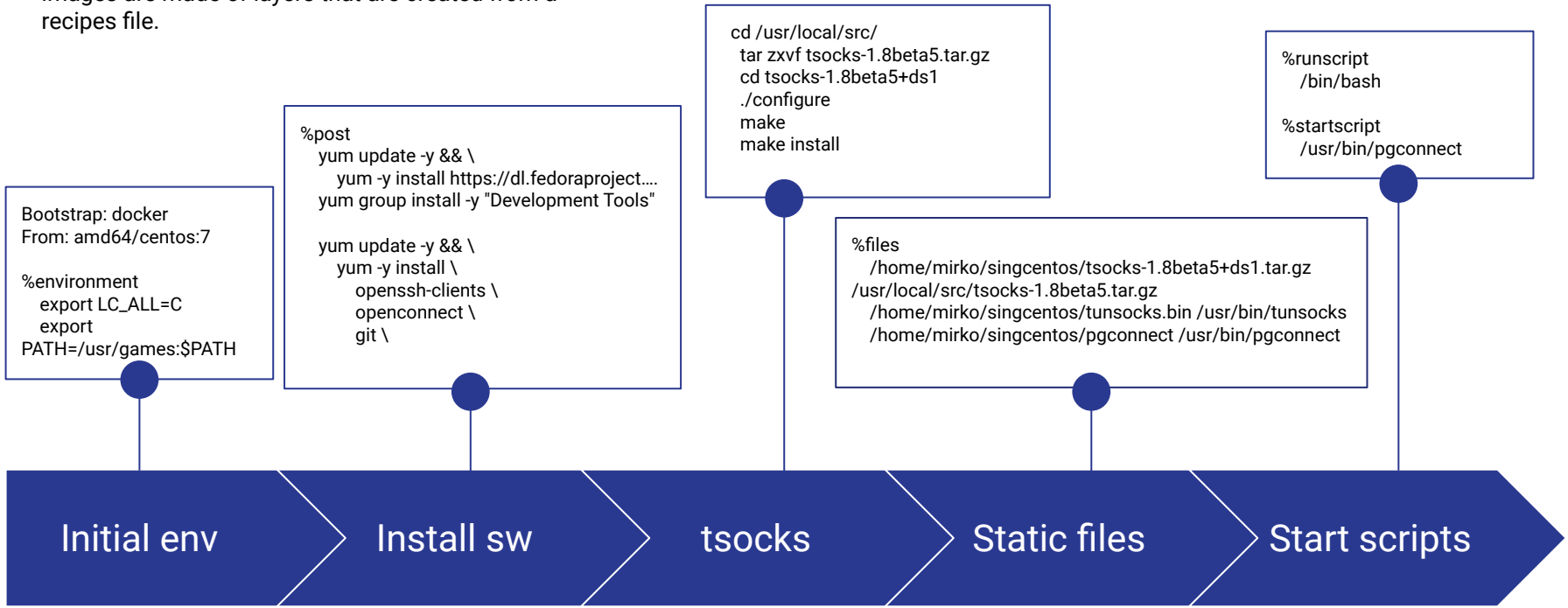
Several possible uses:

1. On systems where only fusermount is available, the mountrepo and umountrepo commands can be used to mount cvmfs repositories in the user's own file space. That path can then be bindmounted at /cvmfs by a container manager such as singularity.
2. On systems where fusermount is available and unprivileged user namespaces are enabled, but unprivileged namespace fuse mounts are not available.
3. On systems where unprivileged namespace fuse mounts are available.
4. On systems that have no fusermount nor unprivileged user namespace fuse mounts, singcvmfs can mount cvmfs repositories inside a container using the singularity --fusemount feature.

Singularity

Singularity is a computer program that performs operating-system-level virtualization also known as containerization. It brings containers and reproducibility to scientific computing and the (HPC) world.

As other containerization software, singularity images are made of layers that are created from a recipes file.



Singularity

Image Build

singularity build imagetest.simg Singularity

Start a container

singularity instance start -B ~/CVMFS:/cmvfs imagetest.simg testcontainer

Open a shell within a container

singularity shell instance://testcontainer

Stop a container

singularity instance stop testcontainer

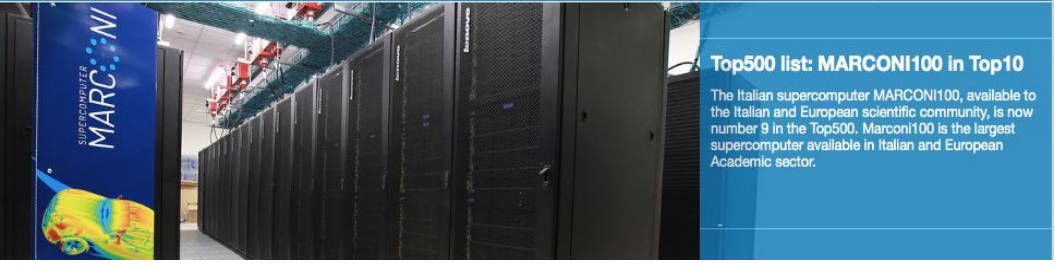
```
Bootstrap: docker
From: amd64/centos:7

%post
   yum update -y && \
      yum -y install https://dl.fedoraproject....
   yum group install -y "Development Tools"

   yum update -y && \
      yum -y install \
         openssh-clients \
         openconnect \
         git \

  cd /usr/local/src/
   tar zxvf tsocks-1.8beta5.tar.gz
   cd tsocks-1.8beta5+ds1
   ./configure
   make
   make install

  echo "server = 127.0.0.1" > /etc/tsocks.conf
  echo "server_port = 5555" >> /etc/tsocks.conf

%files
   /home/mirko/singcentos/tsocks-1.8beta5+ds1.tar.gz /usr/local/src/tsocks-1.8beta5.tar.gz
   /home/mirko/singcentos/tunsocks.bin /usr/bin/tunsocks
   /home/mirko/singcentos/pgconnect /usr/bin/pgconnect

%environment
   export LC_ALL=C
   export PATH=/usr/games:$PATH

%runscript
   /bin/bash

%startscript
   /usr/bin/pgconnect
```

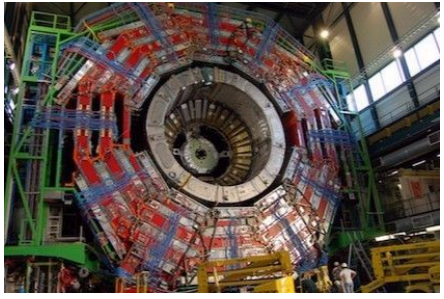
Tests

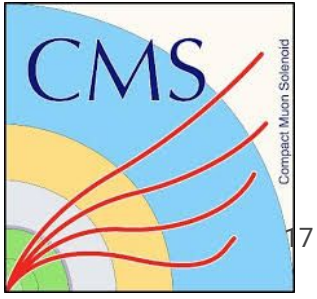
CINECA & CMS

- CINECA is the **HPC/PRACE** node for Italy
- Hosts a number of HPC systems (Marconi, Galileo, Marconi100, Galileo100, DGX, ...)
- Via PRACE grants and partnerships with INFN, **we got access to most of them!**
- Why a good playground?
  - On the **production systems** we got the network access we needed - **thanks!**
  - On other system we **profited** from the missing routing to perform small scale tests (basically just proofs of concept of a few minutes / hours)
- See also "Enabling HPC systems for HEP: the INFN-CINECA experience" in this very session

- CMS is one of the **LHC** experiments
- Its job processing over distributed computing needs in many areas the possibility to connect from the computing node to remote central services
  - Condor Startd to Schedd connections
  - CVMFS for software releases
  - Xrootd/WebDAV/SRM for input output
  - Access to VOMSes
  - Access to CMS central services (Frontier, WMAgent, ...)
  - ...
- Outgoing connections are multiprotocol and in a standard edge service scenario would need multiple proxies / connectors
- **Here we try to encapsulate ALL of them with a single user level tunnel**

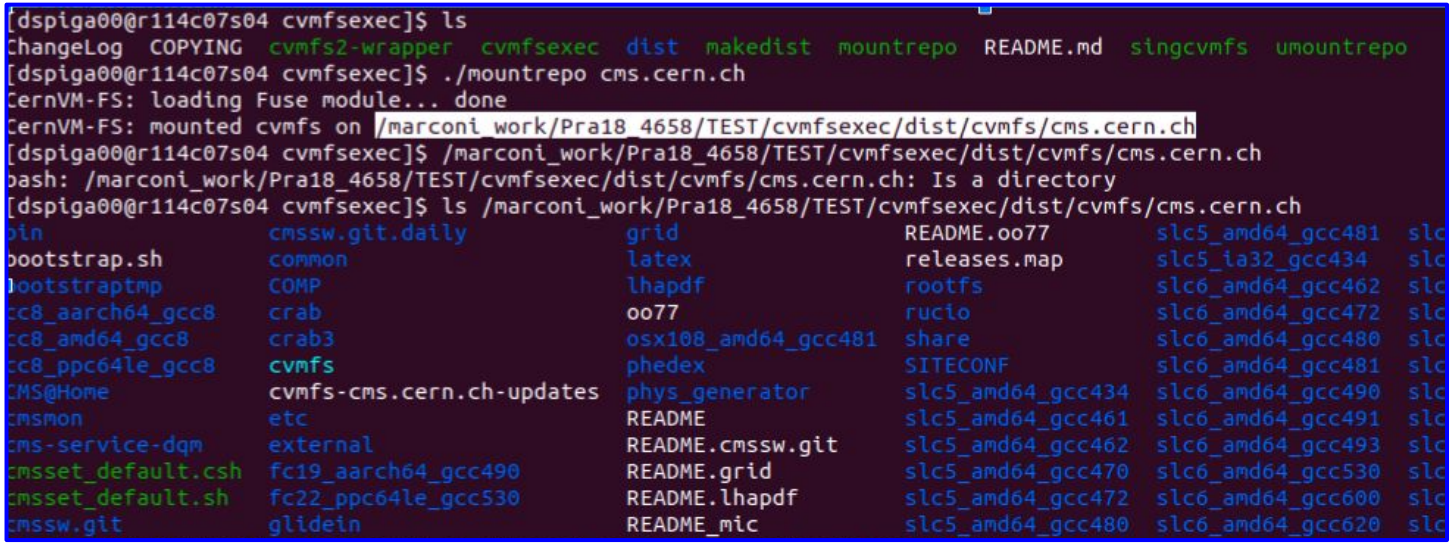






CVMFS mount

CVMFS mounted when no routing to CERN available



Data transfers via SRM

Xrdcp Purdue/Pisa → CINECA directly (via a SOCKS @ Perugia or CERN or CINECA login node)

No routing is normally possible, in the CINECA production environment we needed to deploy an Xrootd proxy @ CNAF

```
Singularity cintest.simg:~> xrdcp -d 1 -f  root://xrootd.rcac.purdue.edu//store/test/rucio/cms/store/mc//RunIIFall17NanoAODv7//EWK_LLJJ_MLL-50_MJJ-
120_TuneCP5_PSweights_13TeV-madgraph-pythia8//NANOAODSIM/PU2017_12Apr2018_Nano02Apr2020_102X_mc2017_realistic_v8-v1//260000/D3D65EB9-4836-C743-8151
-9D9F0178F788.root /dev/null
[33.24MB/33.24MB][100%][==================================================][6.649MB/s]
Singularity cintest.simg:~> xrdcp -f root://stormgf1.pi.infn.it//store/test/xrootd/T2_IT_Pisa/store/mc/SAM/GenericTTbar/AODSIM/CMSSW_9_2_6_91X_mcRu
n1_realistic_v2-v1/00000/A64CCCF2-5C76-E711-B359-0CC47A78A3F8.root /dev/null
[229.3MB/229.3MB][100%][==================================================][45.86MB/s]
```

Srmcp CINECA → external SRM (stageout test)

```
Singularity cintest.simg:~> srmcp  file:////etc/group  srm://t2-srm-02.lnl.infn.it:8443/srm/managerv1?SFN=//pnfs/lnl.infn.it/data/cms/store/user/s
piga/test-cineca-09102020-v2
Singularity cintest.simg:~> srmls srm://t2-srm-02.lnl.infn.it:8443/srm/managerv1?SFN=//pnfs/lnl.infn.it/data/cms/store/user/spiga/
  512 /pnfs/lnl.infn.it/data/cms/store/user/spiga//
      512 /pnfs/lnl.infn.it/data/cms/store/user/spiga/GenericTTbar/
      512 /pnfs/lnl.infn.it/data/cms/store/user/spiga/MinBias/
      515 /pnfs/lnl.infn.it/data/cms/store/user/spiga/test-cineca
      515 /pnfs/lnl.infn.it/data/cms/store/user/spiga/test-cineca-09102020
      515 /pnfs/lnl.infn.it/data/cms/store/user/spiga/test-cineca-09102020-v2
      512 /pnfs/lnl.infn.it/data/cms/store/user/spiga/Zee/
```

Data transfers via SRM: xcheck

```
Singularity cintest.simg:~> export LD_PRELOAD=
Singularity cintest.simg:~> xrdcp -d 1 -f  root://xrootd.rcac.purdue.edu//store/test/rucio/cms/store/mc//RunIIFall17NanoAODv7//EWK_LLJJ_MLL-50_MJJ-
120_TuneCP5_PSweights_13TeV-madgraph-pythia8//NANOAODSIM/PU2017_12Apr2018_Nano02Apr2020_102X_mc2017_realistic_v8-v1//260000/D3D65EB9-4836-C743-8151
-9D9F0178F788.root /dev/null
[2020-10-09 18:11:40.902227 +0200][Error  ][AsyncSock         ] [xrootd.rcac.purdue.edu:1094 #0.0] Unable to connect: No route to host
[2020-10-09 18:11:40.903265 +0200][Error  ][PostMaster        ] [xrootd.rcac.purdue.edu:1094 #0] Elapsed = 0, pConnectionWindow = 120 seconds.
[2020-10-09 18:11:40.903459 +0200][Info   ][PostMaster        ] [xrootd.rcac.purdue.edu:1094 #0] Attempting reconnection in 120 seconds.
^C
Singularity cintest.simg:~> ^C
Singularity cintest.simg:~> srmls -debug  srm://t2-srm-02.lnl.infn.it:8443/srm/managerv1?SFN=//pnfs/lnl.infn.it/data/cms/store/user/spiga/
Storage Resource Manager (SRM) Client version 2.2
Copyright (c) 2002-2009 SRM Working Group http://sdm.lbl.gov/srm-wg

SRM Configuration:
        default_port=8443
        debug=true
        srmcphome=..
        urlcopy=sbin/urlcopy.sh
```

```
        retry_num=20
        delegate=false
        full_delegation=true
        action is ls
        surl[0]=srm://t2-srm-02.lnl.infn.it:8443/srm/managerv1?SFN=//pnfs/lnl.infn.it/data/cms/store/user/spiga/
Fri Oct 09 16:11:50 UTC 2020: In SRMClient ExpectedName: host
Fri Oct 09 16:11:51 UTC 2020: SRMClient(https,srm/managerv2,GSI)
2020-10-09 16:11:55,900 [main] ERROR org.dcache.srm.client.SRMClientV2 - srmLs : try # 0 failed with error No route to host (Host unreachable)
2020-10-09 16:11:55,913 [main] ERROR org.dcache.srm.client.SRMClientV2 -  srmLs : try again
^CFri Oct 09 16:11:57 UTC 2020: stopping
Singularity cintest.simg:~>
```

HTCondor

We used the **manual_glidein_startup** in order to test that the daemon communications are OK

condor startd

```
0/12/20 17:34:53 (pid:10338) slot1: State change: IS_OWNER is false
0/12/20 17:34:53 (pid:10338) slot1: Changing state: Owner -> Unclaimed
0/12/20 17:34:53 (pid:10338) State change: RunBenchmarks is TRUE
0/12/20 17:34:53 (pid:10338) slot1: Changing activity: Idle -> Benchmarking
0/12/20 17:34:53 (pid:10338) BenchMgr:StartBenchmarks()
0/12/20 17:34:56 (pid:10338) Initial update sent to collector(s)
0/12/20 17:34:56 (pid:10338) Sending DC_SET_READY message to master <10.24.120.72:12210?addrs=10.24.120.72-12210>
0/12/20 17:35:14 (pid:10338) State change: benchmarks completed
0/12/20 17:35:14 (pid:10338) slot1: Changing activity: Benchmarking -> Idle
dspiga00@r114c18s04 log]$
```

Global pool (collector)

```
ash-4.2$
ash-4.2$ condor_status  -pool vocms0815 | grep r114c18s04
lot1@glidein_2215_509155035@r114c18s04.marconi.cineca.it              LINUX      X86_64 Unclaimed Idle          0.000  96392  0+00:00:03
```

```
[tboccali@r256n20 ~]$ uname -a
Linux r256n20 4.14.0-115.14.1.el7a.ppc64le #1 SMP Thu Oct 3 05:32:24 EDT 2019 ppc64le ppc64le ppc64le GNU/Linux
cpu             : POWER9, altivec supported
```

CINECA Marconi 100 bare node, no routing, no gateway

Basic setup:

- IBM Power9; no outside networking (no gateway defined)
- But I can do **my_pc → (ssh) → CINECA login node → (SLURM) → compute node**
- From compute node I can ssh back to the login node
- Via a SOCKS tunnel to the login node, **I route**!
- With tsocks + cvmfsexec, **I see CVMFS**!

There is no gateway!

```
[tboccali@r256n20 ~]$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
10.38.0.0       0.0.0.0         255.255.0.0     U     0      0        0 ib0
10.39.0.0       0.0.0.0         255.255.0.0     U     0      0        0 enP5p1s0f0
link-local      0.0.0.0         255.255.0.0     U     1002   0        0 enP5p1s0f0
link-local      0.0.0.0         255.255.0.0     U     1004   0        0 ib0
```

```
[tboccali@r256n20 ~]$ ssh -vvv -D 1085 -o TCPKeepAlive=yes -o ServerAliveInterval=60 tboccali@login01
```

```
[tboccali@r256n20 INF20_test_1]$ export LD_PRELOAD=/m100_work/INF20_test_1/tsocks-1.8beta5+ds1/libtsocks.so
[tboccali@r256n20 INF20_test_1]$  ssh tboccali@lxplus.cern.ch
Warning: Permanently added the ECDSA host key for IP address '188.185.89.71' to the list of known hosts.
tboccali@lxplus.cern.ch's password:
```

```
[tboccali@r256n20 INF20_test_1]$ export LD_PRELOAD=/m100_work/INF20_test_1/tsocks-1.8beta5+ds1/libtsocks.so
[tboccali@r256n20 INF20_test_1]$ /m100_work/INF20_test_1/cvmfsexec/mountrepo cms.cern.ch
CernVM-FS: loading Fuse module... done
CernVM-FS: mounted cvmfs on /m100_work/INF20_test_1/cvmfsexec/dist/cvmfs/cms.cern.ch
```

Marconi100: full dress rehearsal for CMS

- **TSOCKS** to the login node
- **CVMFSEXEC** to mount /cvmfs/cms.cern.ch (via the tsocks)
- Singularity image **docker://cmssw/cc7:ppc64le** with bind mount /cvmfs (all wrapped in tsocks)
- **SQUID** pointing to http://cmsbpfrontier.cern.ch:3128
- **Standard validation test** works flawlessly (SLURM slot with 25 cores, 1 V100)
  - (even tried the GPU workflows, they work with the --nv singularity flag!)

```
%Cpu(s): 29.9 us,  0.6 sy,  2.3 ni, 67.2 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem : 32987308+total, 28578873+free, 39208640 used,  4875712 buff/cache
KiB Swap:  4194240 total,  4194240 free,        0 used. 28867526+avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
45924 tboccali  20   0 6501312   2.2g 378496 R  2492  0.7 470:51.67 cmsRun
```

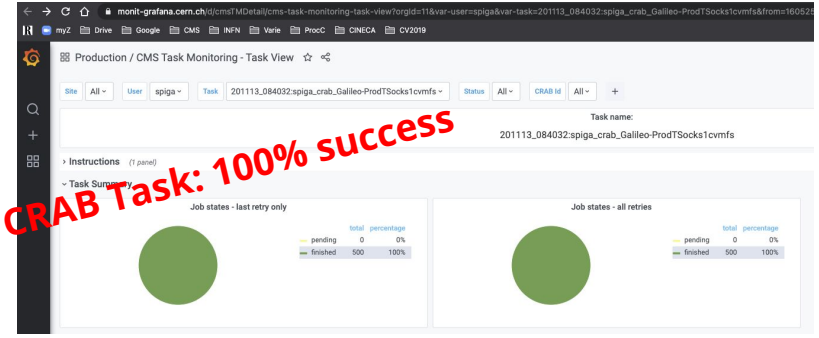
```
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 440.64.00    Driver Version: 440.64.00    CUDA Version: 11.1     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  Tesla V100-SXM2...  On   | 00000035:04:00.0 Off |                    0 |
| N/A   44C    P0    56W / 300W |   2242MiB / 16160MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
|    0     60068      C   cmsRun                                     2231MiB |
+-----------------------------------------------------------------------------+
```

```
Singularity> 11634.502_TTbar_14TeV+2021_Patatrack_PixelOnlyGPU+TTbar_14TeV_TuneCP5
_GenSim+Digi+Reco+HARVEST Step0-PASSED Step1-PASSED Step2-PASSED Step3-PASSED  - t
ime date Wed Oct 14 11:29:31 2020-date Wed Oct 14 11:01:24 2020; exit: 0 0 0 0
1 1 1 1 tests passed, 0 0 0 0 failed
```

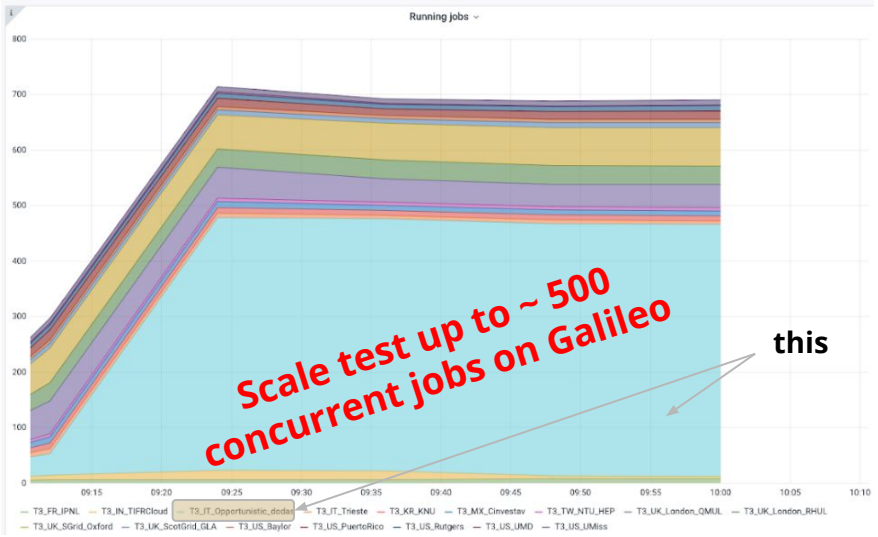
CINECA Galileo

Starting from a system with no CVMFS, no External networking, no suid/sudo rights

- Encapsulate all traffic into a SOCKS5 link
  - CVMFS, access to Frontier, Xrootd input, SRM output
- Start a Condor Glidein and run analysis jobs via CRAB
  - Input @ IFCA, output @ LNL
  - No special configuration needed in CMS SW or Crab config



- Now planning tests @ other HPCs, ideally to help bringing them into production
- Interactive level tests also on M100 (before our requirements were satisfied)



Conclusions

Conclusions

- Going below the application layer, we have tested a solution which potentially can be used in “restricted network” situations (two solutions, indeed…)
- They are protocol and service independent -- a possible “universal edge service” not limited to a single application
  - We positively tested HTCondor, CVMFS, Http, input via Xrootd, stageout via SRM
- Even a single tunnelling host could suffice for a large cluster + data non intensive processing (Monte Carlo generation workflows)
- A wider deployment could scale to allow for data intensive processing
- We do not need these tools @ CINECA, since we handshaked a working network configuration; we are in contact with other HPC centers to test the solution(s)