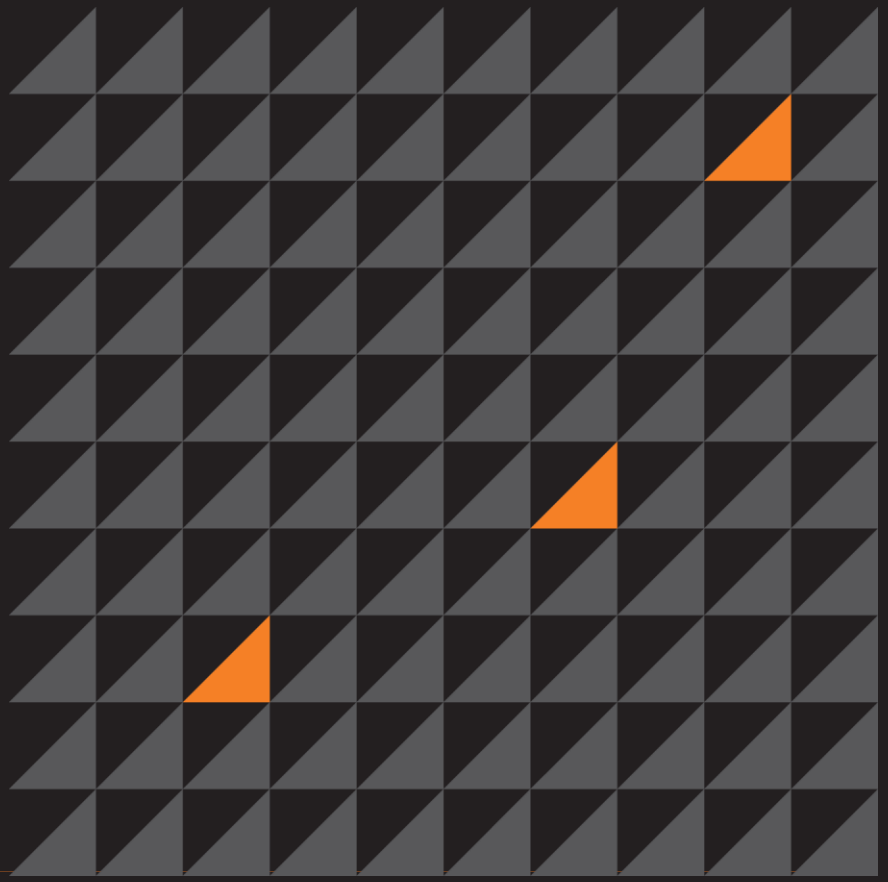
Hack the Gibson

Exploiting Supercomputers

44Con Edition - September 2013

John Fitzpatrick
Luke Jennings



Outline

- Introduction
- Important Security Considerations
- Job Schedulers
  - MOAB
  - Torque
- OpenMPI
- Distributed Filesystems
  - GPFS
  - Lustre
- Cluster Management
- Local Privilege Escalation

Outline

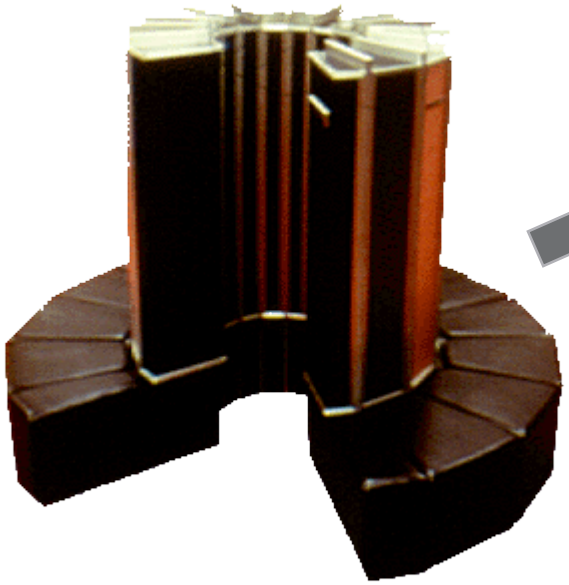
- **Introduction**
- Important Security Considerations
- Job Schedulers
  - MOAB
  - Torque
- OpenMPI
- Distributed Filesystems
  - GPFS
  - Lustre
- Cluster Management
- Local Privilege Escalation

What is High Performance Computing?

- Computer hardware and software aimed at solving very heavy computational tasks

- Common Uses
  - Weather forecasting
  - Data Mining
  - Cryptanalysis
  - Nuclear weapons simulation
  - Molecular dynamics

History



Some Numbers

Tianhe-2

Top500 rank: #1 (June 2013)

- China's National University of Defense Technology
- 33.86 petaflops/s (Linpack)
- 16,000 nodes (2x Intel Xeon IvyBridge + 3x Xeon Phi processors)
- Total 3,120,000 cores
- Memory: 1,375 TiB ~1,500 TB
- Storage: 12.4 PB
- Cost: ~£256 Million

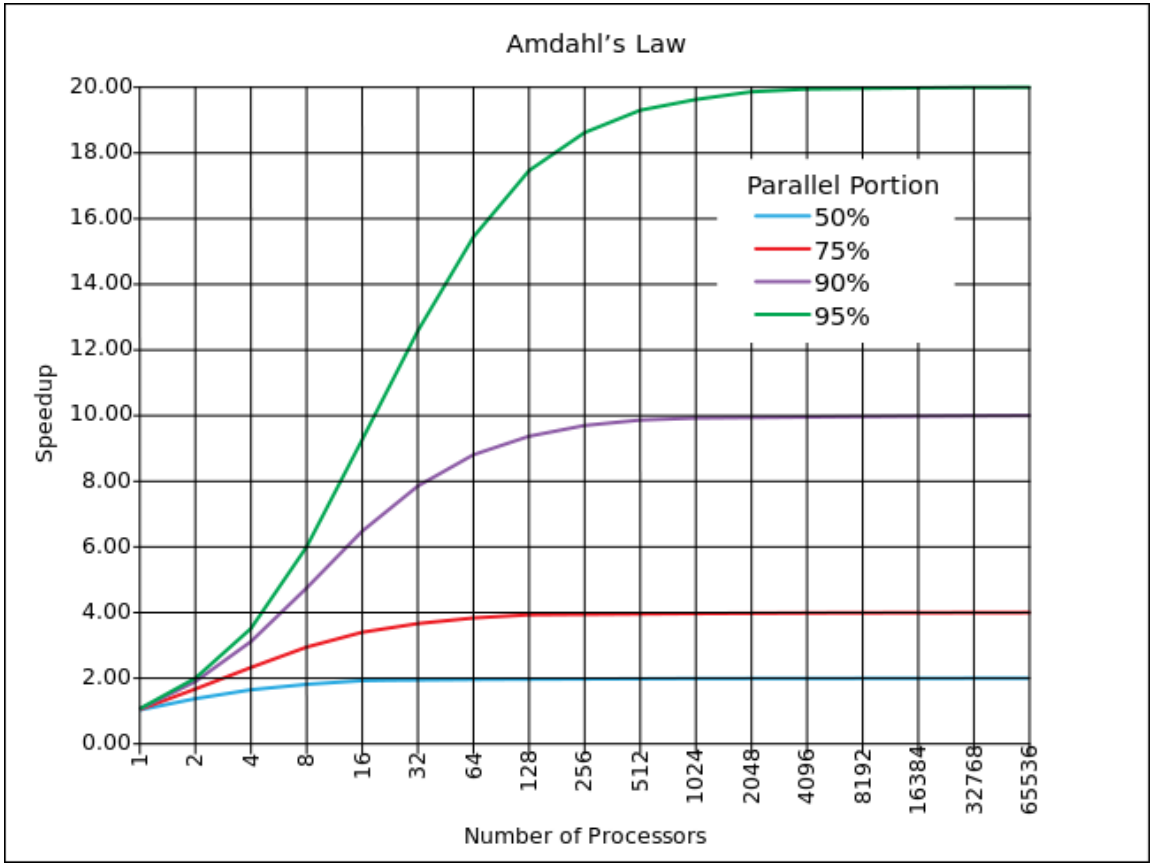
Blue Joule (UK Fastest)

Top500 rank: #18 (June 2013)

- Daresbury Laboratory
- IBM Blue Gene/Q
- 1.25 petaflops/s (Linpack)
- Total 114,688 cores
- Storage ~5 PB

Parallel Computing - Limits

- Not all tasks can be parallelised e.g. Task B requires the result of Task A



Message Passing Interface

- Achieves Cluster Computing Goal
- Implements the concept of “Distributed Shared Memory”

```
main(int argc, char **argv)
{
    int node;

    MPI_Init(&argc,&argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &node);

    printf("I am %d\n",node);

    MPI_Finalize();
}
```

```
I am 2
I am 0
I am 3
I am 1
```

Supercomputer vs Cluster

- What really makes a true “supercomputer” is the use of extremely high performance networking kit to link the nodes
- “Distributed Shared Memory” with normal Ethernet network connections will be orders of magnitude slower than RAM
- Infiniband and Myrinet
  - Upto 300 Gbit/s with link aggregation!
  - 1 µs latency (that’s micro not milli)

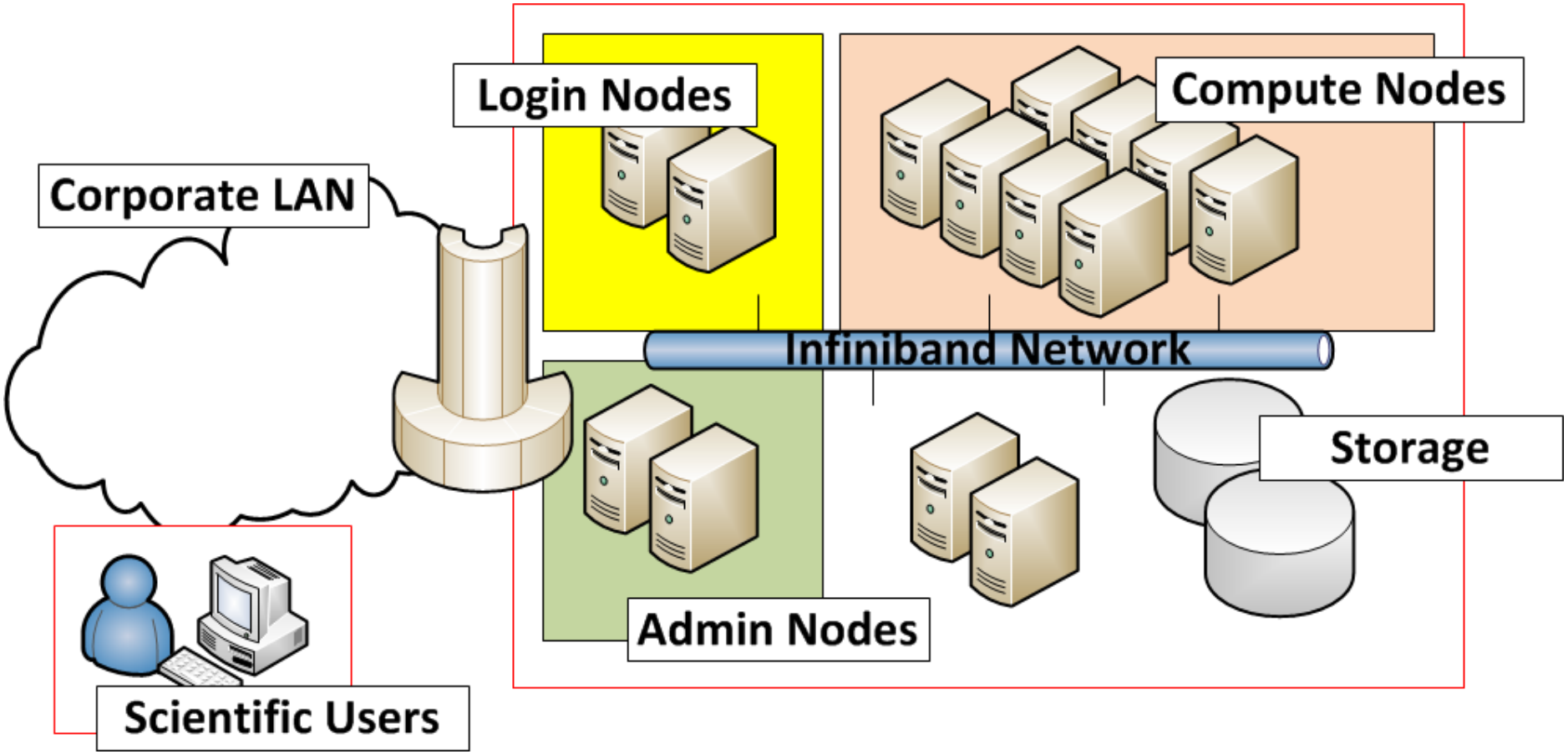
Supercomputer – what about file systems?

- Distributed file systems are used
  - Much higher performance
- Each node will generally have access to identical data
  - Though its boot drive may be unique
- Examples
  - GPFS
  - Lustre

Supercomputer – Job Scheduling

- Not all nodes should be useable by anyone all the time…. Wait your turn!
- Generally a user may write a parallel program using MPI and then submit it to a job scheduler
- This then manages all the submissions and decides what runs on what nodes and what time etc.

Summary Architecture



Outline

- Introduction
- **Important Security Considerations**
- Job Schedulers
  - MOAB
  - Torque
- OpenMPI
- Distributed Filesystems
  - GPFS
  - Lustre
- Cluster Management
- Local Privilege Escalation

Important Security Considerations

Considerations

- Users can typically connect to login nodes
- Users run code on compute nodes
- Trust relationships exist (passwordless key based SSH)
- Job scheduling happens across many nodes
- Nodes may need regular rebuilding

Implications

- Local priv-esc is a very big deal
- Users have access to inner components
- Remote imaging is pretty much essential
- Several critical OS mechanisms are now network accessible

Outline

- Introduction
- Important Security Considerations
- **Job Schedulers**
  - MOAB
  - Torque
- OpenMPI
- Distributed Filesystems
  - GPFS
  - Lustre
- Cluster Management
- Local Privilege Escalation

Job Schedulers

- **Resource Managers e.g. Torque, SLURM**
  - Handle actual running of jobs on a single cluster
  - Basic logic for scheduling e.g. FIFO, round robin etc

- **Workload Managers (Schedulers) e.g. MOAB**
  - Sit on top of resource managers
  - Can handle multiple clusters potentially with different resource managers
  - Intelligent, flexible, scalable scheduling

Outline

- Introduction
- Important Security Considerations
- **Job Schedulers**
  - **MOAB**
  - Torque
- OpenMPI
- Distributed Filesystems
  - GPFS
  - Lustre
- Cluster Management
- Local Privilege Escalation

Job Schedulers - MOAB

- Submit jobs using the “msub” command
- This talks to a web server using XML

```
<job>
      <user>jbloggs</user>
      <group>jbloggs</group>
      <command user="jbloggs" group="jbloggs">helloworld-proj</command>
      <nodelist>node1,node2,node3</nodelist>
</job>
<hmac>2465ebc8cd6e412cdc1ab9fef40bcae6</hmac>
```

- HMAC verifies authenticity of user information submitted
- Default behaviour is insecure – key to generate hmac is hardcoded in binary

Job Schedulers - MOAB

- More secure option is to use “mauth” authentication
- Key is configured in a protected file

```
-r-------- 1 root root 15 2007-04-05 03:47 /opt/moab/.moab.key
```

- When “msub” is called, it makes use of a SUID root binary called “mauth” to obtain the HMAC for the XML
- “mauth” can read the key (SUID root)
- If the user information submitted does not match the caller, it rejects the request
- Should be secure right?

Job Schedulers - MOAB

```
<job>
     <user>jbloggs</user>
     <group>jbloggs</group>
     <command user="jbloggs" group="jbloggs">helloworld-proj</command>
     <nodelist>node1,node2,node3</nodelist>
</job>
<hmac>2465ebc8cd6e412cdc1ab9fef40bcae6</hmac>
```

```
<job>
     <user>root</user>
     <group>root</group>
     <command user="jbloggs" group="jbloggs">helloworld-proj</command>
     <nodelist>node1,node2,node3</nodelist>
</job>
<hmac>2465ebc8cd6e412cdc1ab9fef40bcae6</hmac>
```

```
<job>
     <user>jbloggs</user>
     <group>jbloggs</group>
     <command user="root" group="root">helloworld-proj</command>
     <nodelist>node1,node2,node3</nodelist>
</job>
<hmac>2465ebc8cd6e412cdc1ab9fef40bcae6</hmac>
```

Job Schedulers - MOAB

- Web service and “mauth” check different user IDs ☺
- By default, “root” jobs are not allowed
- However, web service allows “dynamic reconfiguration”
- Pretend to be “root” to enable root jobs, then submit after ☺

Outline

- Introduction
- Important Security Considerations
- **Job Schedulers**
  - MOAB
  - **Torque**
- OpenMPI
- Distributed Filesystems
  - GPFS
  - Lustre
- Cluster Management
- Local Privilege Escalation

Job Schedulers - Torque

- Resource manager
- Terascale Open-Source Resource and Queue Manager
- Can integrate with MAIU/MOAB
- Based on the original PBS project

Job Schedulers - Torque

***2.4.x** – support ended August 2012*

2.5.x – Widely used, recommended, considered reliable
2.5.13 = Latest

***3.0.x** - has everything 2.5.x has plus it supports NUMA architectures*

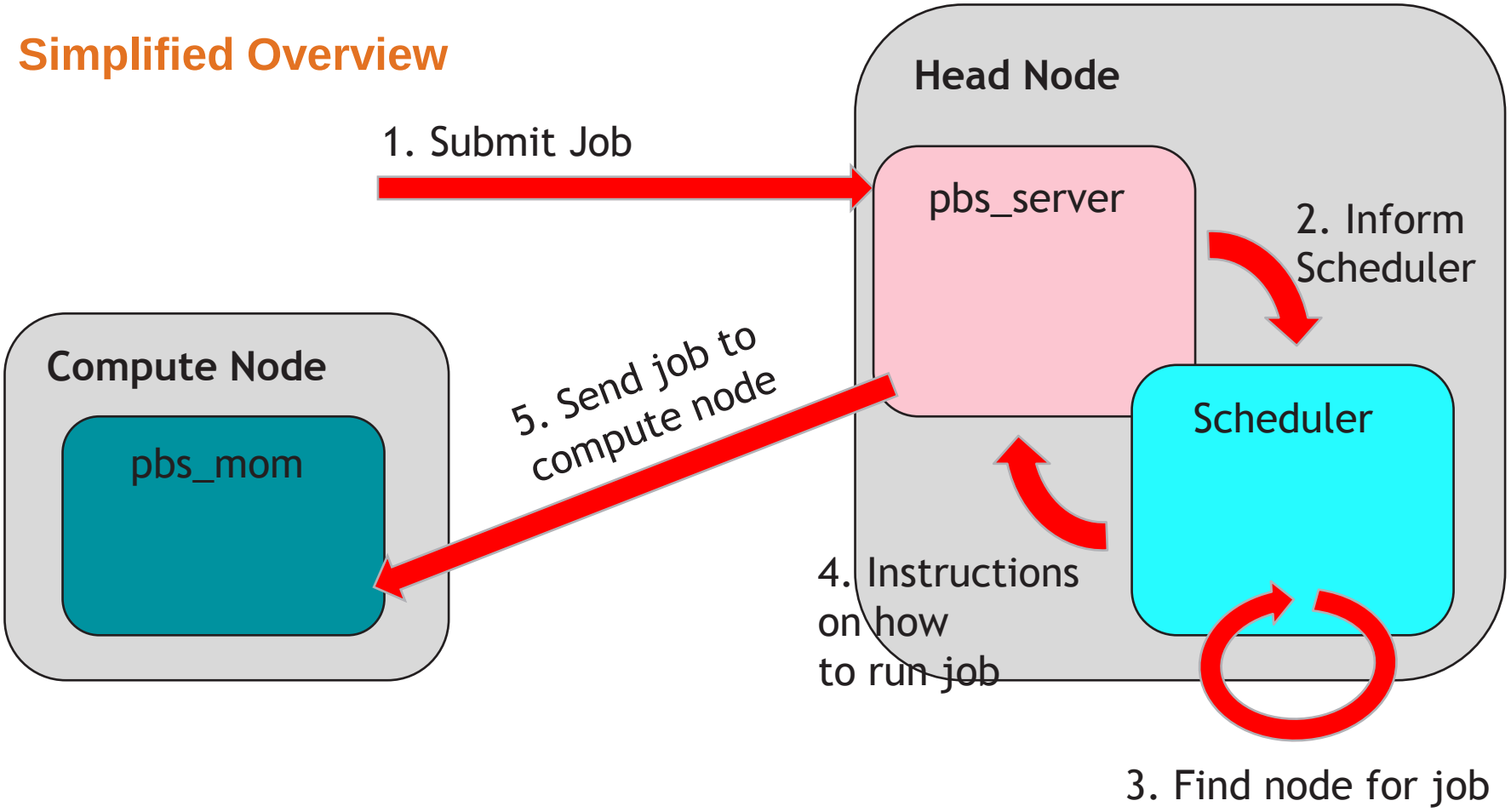
***4.0.x** - has significant enhancements to scalability for petaflop and large environments*

***4.1.x** - has added significant enhancements for Cray systems*

4.2.x – Includes support for Intel Xeon Phi
4.2.3.1 = Latest

Job Schedulers - Torque

Simplified Overview

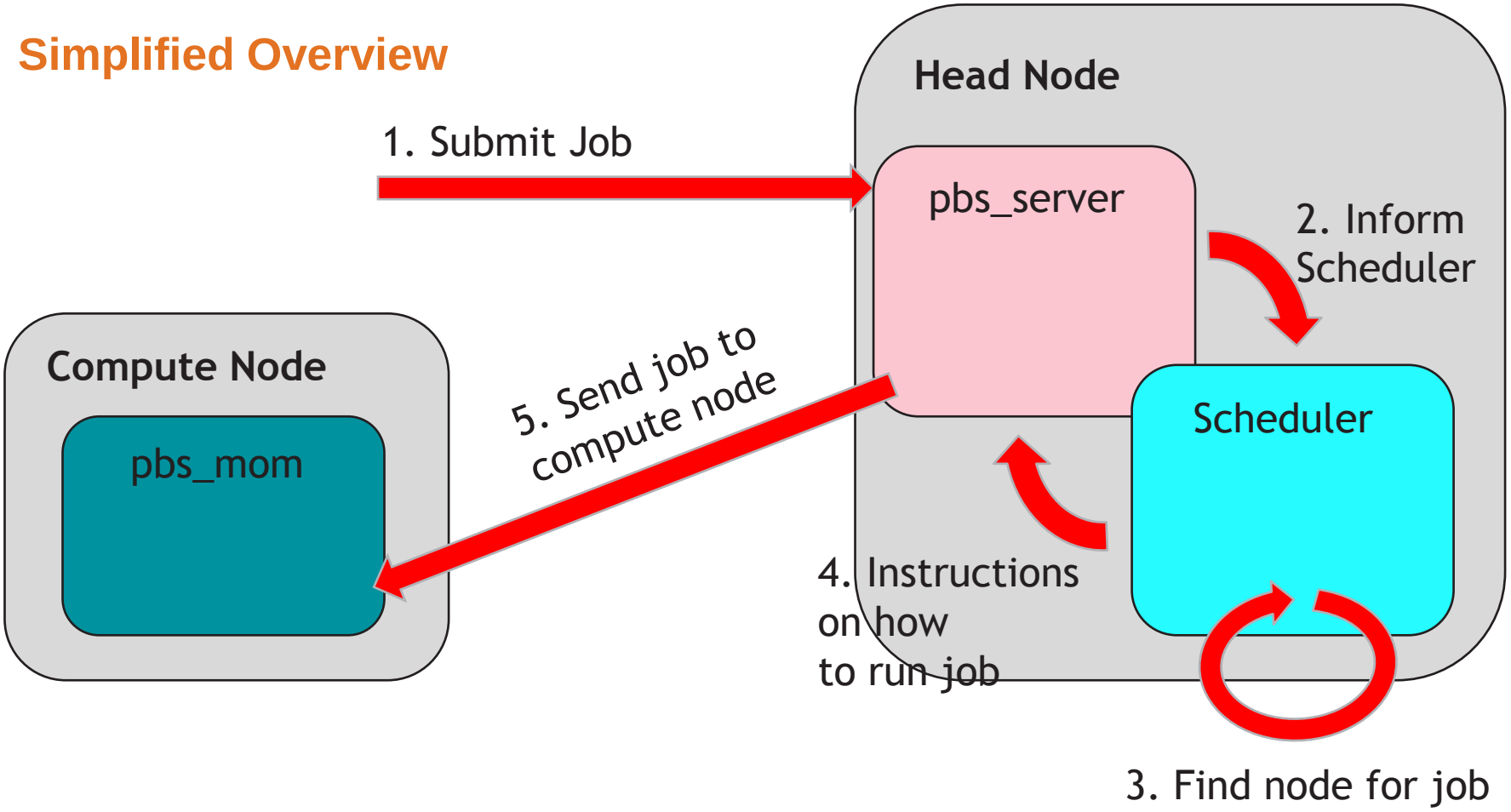


Job Schedulers - Torque

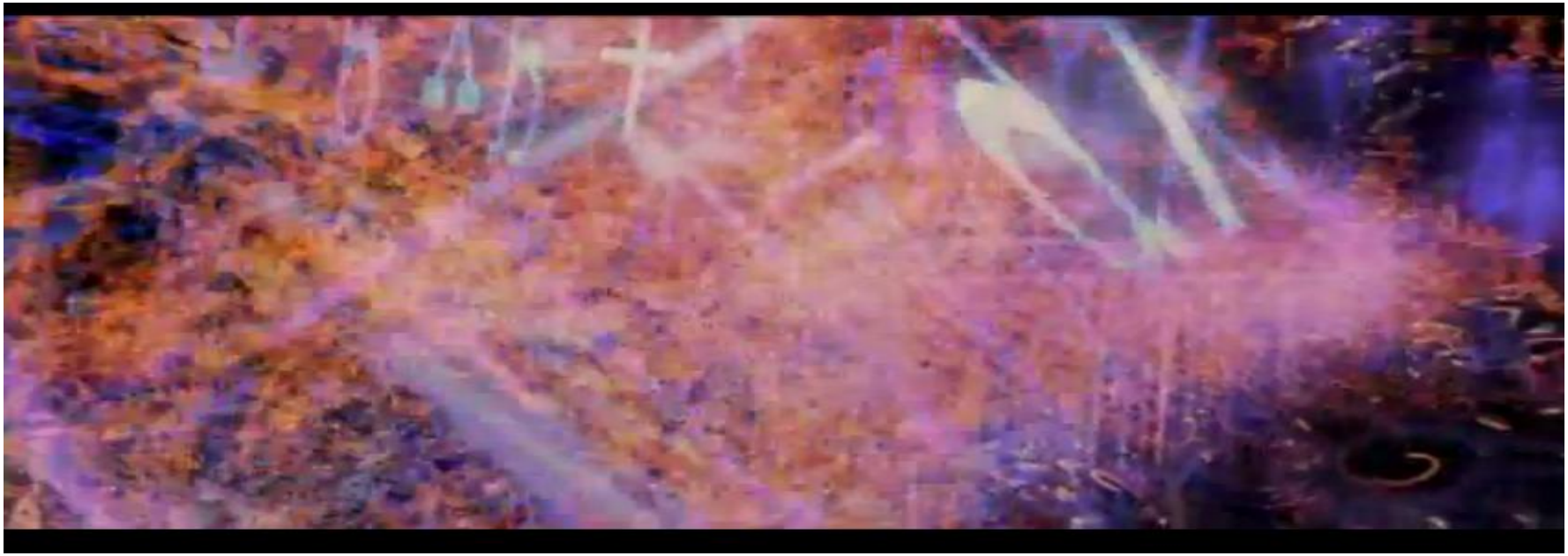
```
$ qstat
Job id                    Name             User            Time Use S Queue
------------------------- ---------------- --------------- -------- - -----
85.server1                STDIN            hpc1                   0 Q batch
87.host1                  STDIN            rd002                  0 Q batch
88.host2                  STDIN            user1                  0 Q batch
89.host2                  STDIN            user1                  0 Q batch
90.server1                STDIN            testuser               0 Q batch
91.host1                  STDIN            rd002                  0 Q batch
```

Job Schedulers - Torque

Simplified Overview



Job Schedulers - Torque



So let's hack the Gibson…

- unauthenticated remote root ;)

Outline

- Introduction
- Important Security Considerations
- Job Schedulers
  - MOAB
  - Torque
- **OpenMPI**
- Distributed Filesystems
  - GPFS
  - Lustre
- Cluster Management
- Local Privilege Escalation

Open MPI

Running standard program e.g. uname

```
root@bt5:~/mpi# mpirun -H node,localhost uname -a
Linux bt5 3.2.6 #1 SMP Fri Feb 17 10:34:20 EST 2012 x86_64 GNU/Linux
Linux bt 2.6.39.4 #1 SMP Thu Aug 18 13:38:02 NZST 2011 i686
GNU/Linux
```

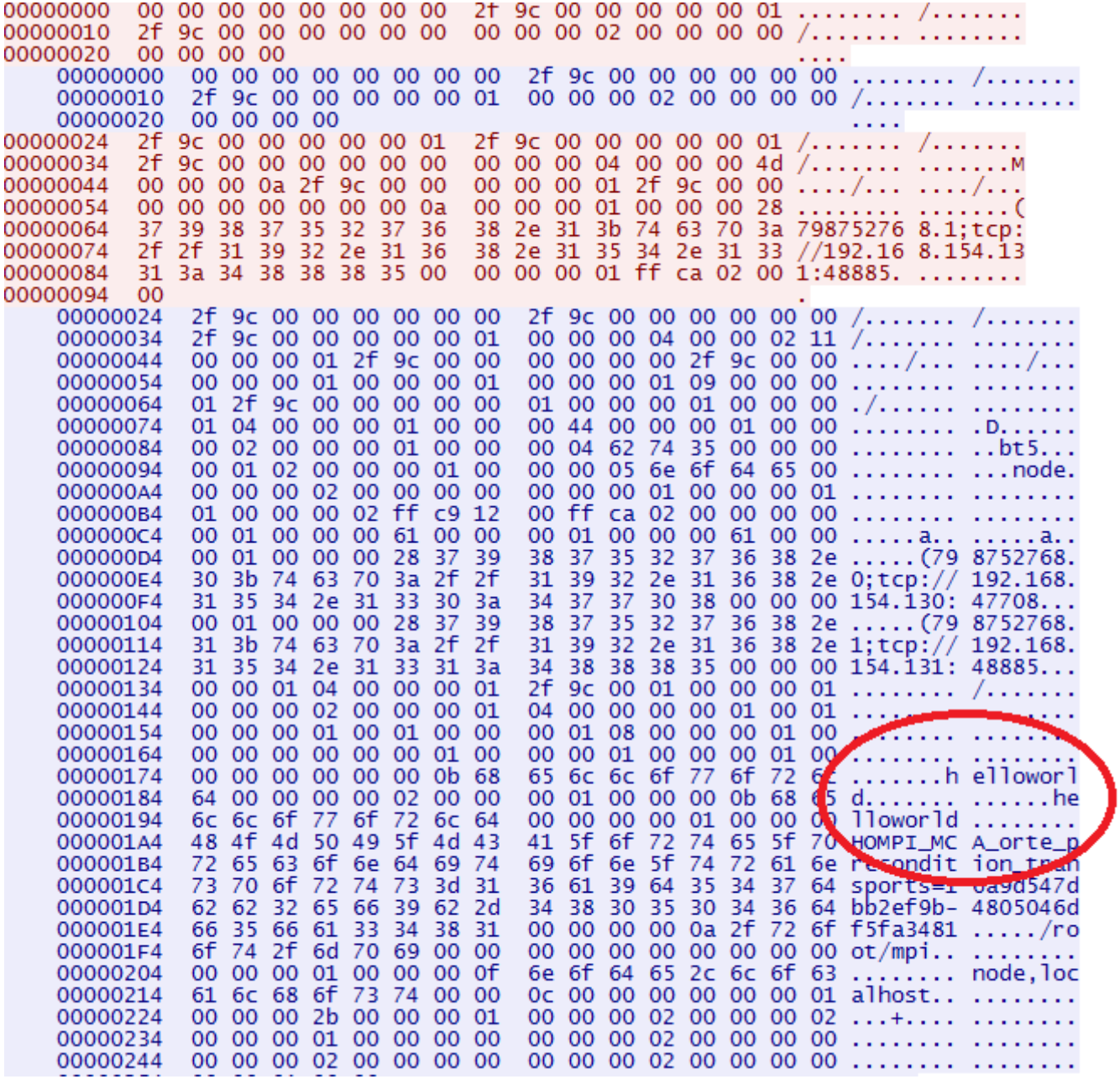
Running custom MPI program

```
root@bt5:~/mpi# mpirun -H node,localhost helloworld
Hello from process 0 of 2 running on bt (0)
Processor 0 sending buf data
Hello from process 1 of 2 running on bt5 (0)
Processor 1 received buf data: Hello :)
Process 0 received reply: Wagwan!
```

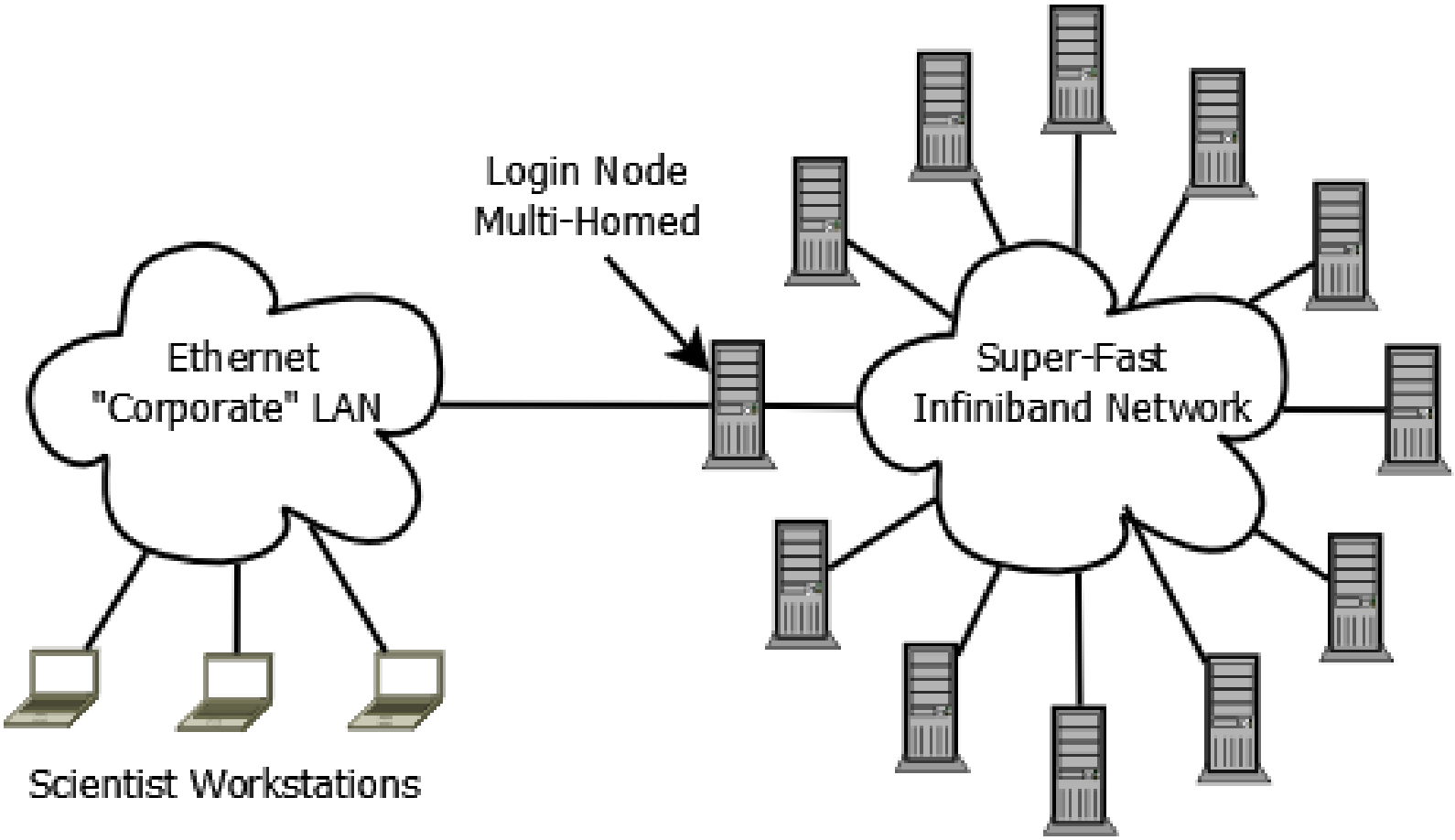
Open MPI

```
/usr/bin/ssh -x node  orted
        --daemonize
        -mca ess env
        -mca orte_ess_jobid 76873728
        -mca orte_ess_vpid 1
        -mca orte_ess_num_procs 2
        --hnp-uri "76873728.0;tcp://192.168.154.130:43622"
```

Open MPI


00000000 00 00 00 00 00 00 00 00 2f 9c 00 00 00 00 00 01 .... /.......
00000010 2f 9c 00 00 00 00 00 00 00 00 00 02 00 00 00 00 /....... ....
00000020 00 00 00 00 ....
00000000 00 00 00 00 00 00 00 00 2f 9c 00 00 00 00 00 00 .... /.......
00000010 2f 9c 00 00 00 00 00 01 00 00 00 02 00 00 00 00 /....... ....
00000020 00 00 00 00 ....
00000024 2f 9c 00 00 00 00 00 01 2f 9c 00 00 00 00 00 01 /....... /.......
00000034 2f 9c 00 00 00 00 00 00 00 00 00 04 00 00 00 4d /....... ....M
00000044 00 00 00 0a 2f 9c 00 00 00 00 00 01 2f 9c 00 00 ..../... ..../...
00000054 00 00 00 00 00 00 00 0a 00 00 00 01 00 00 00 28 ....(
00000064 37 39 38 37 35 32 37 36 38 2e 31 3b 74 63 70 3a 79875276 8.1;tcp:
00000074 2f 2f 31 39 32 2e 31 36 38 2e 31 35 34 2e 31 33 //192.16 8.154.13
00000084 31 3a 34 38 38 38 35 00 00 00 00 01 ff ca 02 00 1:48885. ....
00000094 00 .
00000024 2f 9c 00 00 00 00 00 00 2f 9c 00 00 00 00 00 00 /....... /.......
00000034 2f 9c 00 00 00 00 00 01 00 00 00 04 00 00 02 11 /....... ....
00000044 00 00 00 01 2f 9c 00 00 00 00 00 00 2f 9c 00 00 ..../... ..../...
00000054 00 00 00 01 00 00 00 01 00 00 00 01 09 00 00 00 ....
00000064 01 2f 9c 00 00 00 00 00 01 00 00 00 01 00 00 00 ./...... ....
00000074 01 04 00 00 00 01 00 00 00 44 00 00 00 01 00 00 .... .D......
00000084 00 02 00 00 00 01 00 00 00 04 62 74 35 00 00 00 ....bt5...
00000094 00 01 02 00 00 00 01 00 00 00 05 6e 6f 64 65 00 ....node.
000000A4 00 00 00 02 00 00 00 00 00 00 00 01 00 00 00 01 ....
000000B4 01 00 00 00 02 ff c9 12 00 ff ca 02 00 00 00 00 ....
000000C4 00 01 00 00 00 61 00 00 00 01 00 00 00 61 00 00 .....a.. .....a..
000000D4 00 01 00 00 00 28 37 39 38 37 35 32 37 36 38 2e .....(79 8752768.
000000E4 30 3b 74 63 70 3a 2f 2f 31 39 32 2e 31 36 38 2e 0;tcp:// 192.168.
000000F4 31 35 34 2e 31 33 30 3a 34 37 37 30 38 00 00 00 154.130: 47708...
00000104 00 01 00 00 00 28 37 39 38 37 35 32 37 36 38 2e .....(79 8752768.
00000114 31 3b 74 63 70 3a 2f 2f 31 39 32 2e 31 36 38 2e 1;tcp:// 192.168.
00000124 31 35 34 2e 31 33 31 3a 34 38 38 38 35 00 00 00 154.131: 48885...
00000134 00 00 01 04 00 00 00 01 2f 9c 00 01 00 00 00 01 .... /.......
00000144 00 00 00 02 00 00 00 01 04 00 00 00 00 01 00 01 ....
00000154 00 00 00 01 00 01 00 00 00 01 08 00 00 00 01 00 ....
00000164 00 00 00 00 00 00 01 00 00 00 01 00 00 00 01 00 ....
00000174 00 00 00 00 00 00 0b 68 65 6c 6c 6f 77 6f 72 6c ....h ellowor l
00000184 64 00 00 00 00 02 00 00 00 01 00 00 00 0b 68 65 d....... ......he
00000194 6c 6c 6f 77 6f 72 6c 64 00 00 00 00 01 00 00 00 lloworld ....
000001A4 48 4f 4d 50 49 5f 4d 43 41 5f 6f 72 74 65 5f 70 HOMPI_MC A_orte_p
000001B4 72 65 63 6f 6e 64 69 74 69 6f 6e 5f 74 72 61 6e recondit ion_tran
000001C4 73 70 6f 72 74 73 3d 31 36 61 39 64 35 34 37 64 sports=1 6a9d547d
000001D4 62 62 32 65 66 39 62 2d 34 38 30 35 30 34 36 64 bb2ef9b- 4805046d
000001E4 66 35 66 61 33 34 38 31 00 00 00 00 0a 2f 72 6f f5fa3481 ...../ro
000001F4 6f 74 2f 6d 70 69 00 00 00 00 00 00 00 00 00 00 ot/mpi.. ....
00000204 00 00 00 01 00 00 00 0f 6e 6f 64 65 2c 6c 6f 63 .... node,loc
00000214 61 6c 68 6f 73 74 00 00 0c 00 00 00 00 00 00 01 alhost.. ....
00000224 00 00 00 2b 00 00 00 01 00 00 00 02 00 00 00 02 ...+.... ....
00000234 00 00 00 01 00 00 00 00 00 00 00 02 00 00 00 00 ....
00000244 00 00 00 02 00 00 00 00 00 00 00 02 00 00 00 00 ....

Open MPI


Login Node
Multi-Homed
Ethernet
"Corporate" LAN
Super-Fast
Infiniband Network
Scientist Workstations

Open MPI

- Vulnerable to a MITM...but most MITM attacks require root privileges e.g. ARP spoofing
- If we have root on a node, we have root on everything anyway
- The “Super-fast” network is only going to be physically accessible from within the data centre
- ...so is this vulnerability actually applicable to the environment?

Open MPI

- Exploitable as a low privileged user?
  - What if I connect to the port first, before the remote node?
  - Will the master node give up listening, so I can hijack the port?
  - Race condition?

- Can I win the race?
  - Legitimate workflow involves an SSH connection then a TCP connect back
  - My code can all run on the master node
  - No network comms = I have the speed advantage

Open MPI

- The port we could brute force, but we need that ID too…

```
--hnp-uri "76873728.0;tcp://192.168.154.130:43622"
```

- …but Linux normally gives away command line arguments via /proc file system e.g. “ps aux” below

```
daemon     833  0.0  0.0  18932   392 ?        Ss   Nov05   0:00 atd
root       834  0.0  0.0  21124  1024 ?        Ss   Nov05   0:15 cron
root       835  0.0  0.0  11332   640 ?        Ss   Nov05   0:12 /usr/sbin/irqbalance
root       844  0.0  0.1 120676  3644 ?        Sl   Nov05   0:00 /usr/sbin/console-kit-daemon --no-daemon
postgres   965  0.0  0.1  49356  4080 ?        S    Nov05   0:00 /opt/metasploit/postgresql/bin/postgres -D
/opt/metasploit/postgresql/data -p 7337
postgres  1278  0.0  0.0  49356  1244 ?        Ss   Nov05   0:20 postgres: writer process
postgres  1279  0.0  0.0  49356   976 ?        Ss   Nov05   0:17 postgres: wal writer process
postgres  1280  0.0  0.0  49492  1180 ?        Ss   Nov05   0:19 postgres: autovacuum launcher process
postgres  1281  0.0  0.0  20764  1008 ?        Ss   Nov05   0:03 postgres: stats collector process
root      1303  0.0  0.0  59480   716 ?        Ssl  Nov05   0:00 /usr/sbin/vmware-vmblock-fuse -o
subtype=vmware-vmblock,default_permissions,allow_other /var/run/vmblock-fuse
root      1359  0.0  0.1  85000  4008 ?        S    Nov05   1:28 /usr/sbin/vmtoolsd
root      1423  0.0  0.0      0     0 ?        S    Nov05   0:05 [flush-8:0]
root      1439  0.0  0.0   6604   664 ?        Ss   Nov05   0:15 dhclient3 -e IF_METRIC=100 -pf
/var/run/dhclient.eth1.pid -lf /var/lib/dhcp3/dhclient.eth1.leases eth1
ntp       1560  0.0  0.0  27912  1580 ?        Ss   Nov05   0:06 /usr/sbin/ntpd -p /var/run/ntpd.pid -g -u
113:121
```

Open MPI

The Exploit

1. Monitor /proc file system
2. When the ssh connection is detected, read the port number and application specific ID
3. Connect to port and “talk” MPI to it along with the correct ID so the master node is happy execution has started
4. Master node will then release the port, so we can listen on it
5. If we are quick enough, we receive the connect back from the remote node and “talk” MPI and issue our own “command”

Open MPI

The Exploit

DEMO

Outline

- Introduction
- Important Security Considerations
- Job Schedulers
  - MOAB
  - Torque
- OpenMPI
- **Distributed Filesystems**
  - GPFS
  - Lustre
- Cluster Management
- Local Privilege Escalation

Distributed Filesystems

- Gbit/sec
- Infiniband
- Petabytes
- Tens of thousands of disks

Distributed Filesystems

- Commonly used:
  - GPFS
  - NFS
  - Lustre

- Performance is Key

Outline

- Introduction
- Important Security Considerations
- Job Schedulers
  - MOAB
  - Torque
- OpenMPI
- **Distributed Filesystems**
  - **GPFS**
  - Lustre
- Cluster Management
- Local Privilege Escalation

Distributed Filesystems - GPFS

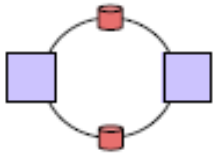
IBM – file system

"It's quite expensive - we bought an instance of GPFS from IBM and they gave us a free supercomputer with it"

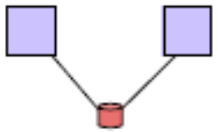
Good tech overview here:
https://computing.llnl.gov/code/sio/ucrl-web-148419-pt-1.pdf

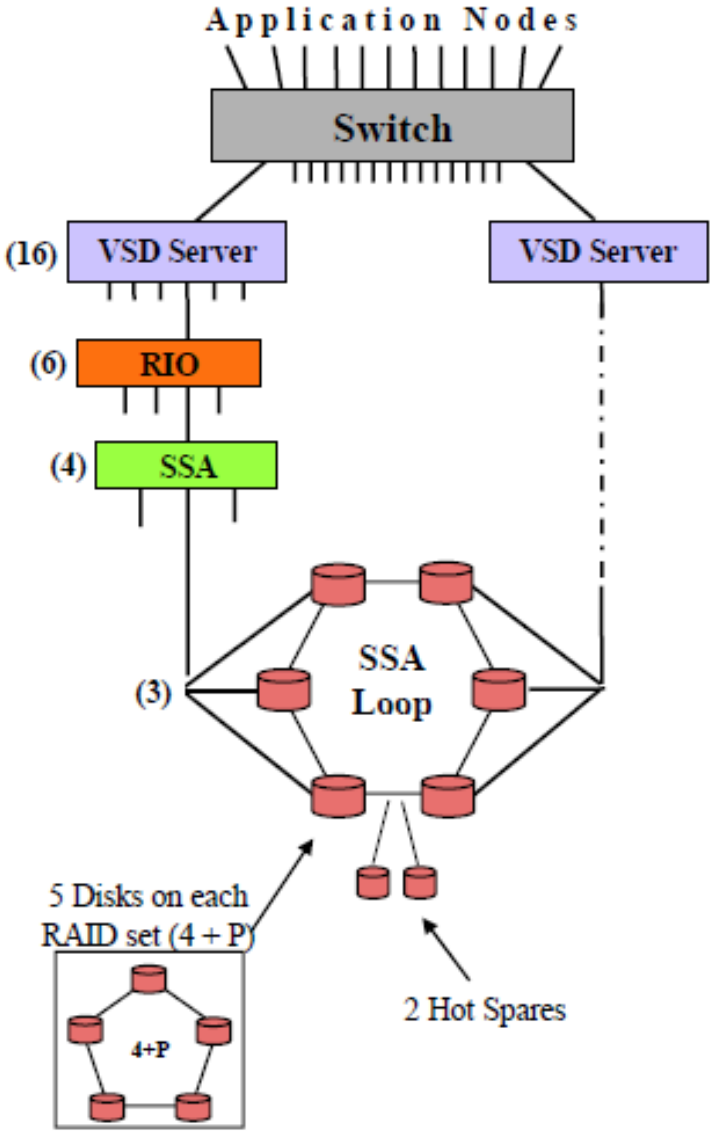
Distributed Filesystems - GPFS

- application node – node running user app that accesses mounted GPFS
- VSD server node – *Virtual Shared Disk* node (or I/O node) with disks attached
- RIO – *Remote I/O* that allows for a connection between the server node and the disks
- SSA loop – *Serial Storage Architecture* disks may be connected in a loop between nodes, allowing for failover:



(This is in contrast to a twin-tailed approach which is used for SCSI [*Small Computer System Interface*]disks to connect two nodes to a single disk.)





Bill Loewe: https://computing.llnl.gov/code/sio/ucrl-web-148419-pt-1.pdf

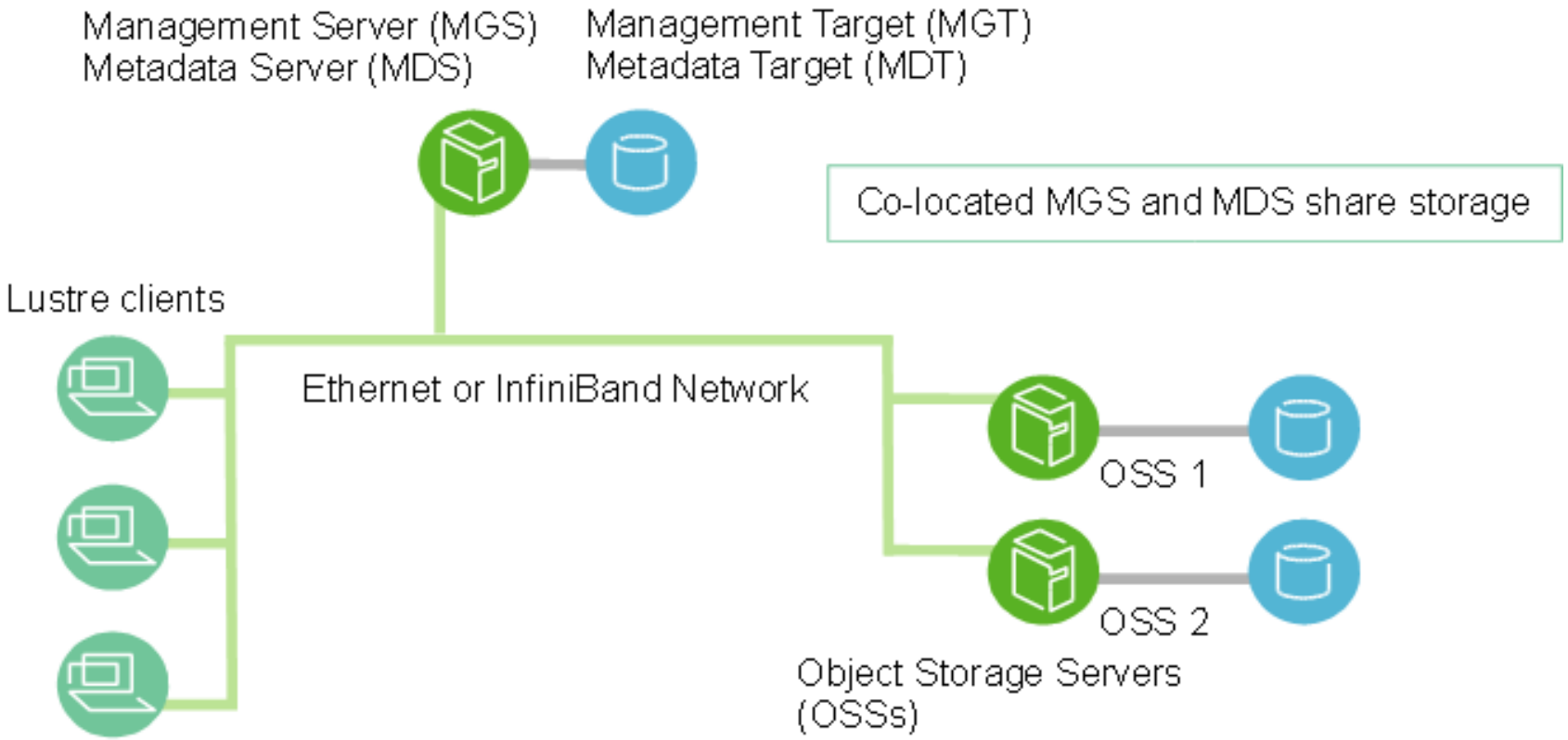
Outline

- Introduction
- Important Security Considerations
- Job Schedulers
  - MOAB
  - Torque
- OpenMPI
- **Distributed Filesystems**
  - GPFS
  - **Lustre**
- Cluster Management
- Local Privilege Escalation

Distributed Filesystems - Lustre

- Widely used
- Over 60 of the top 100 supercomputers use Lustre
- Resold in appliances

Distributed Filesystems - Lustre



Distributed Filesystems - Lustre

The Exploit

(found by reading the documentation!)

DEMO

Outline

- Introduction
- Important Security Considerations
- Job Schedulers
  - MOAB
  - Torque
- OpenMPI
- Distributed Filesystems
  - GPFS
  - Lustre
- **Cluster Management**
- Local Privilege Escalation

Cluster Management

- Tens of thousands of nodes = remote imaging is the norm

- Used to this sort of technology from corporate environments e.g. PXE booting rebuilds etc.

- We've seen a couple of vendor implementations
  - Both used PXE + TFTP (as is common)
  - One used custom protocols for the second stage
  - The other used common protocols like rsync

Cluster Management

- The nature of this type of technology makes it very difficult to secure directly

- But two things are key:
  - Read-only images -> otherwise suffer backdoors
  - No sensitive information in the images -> work to the assumption of no confidentiality

- Sensitive information
  - E.g. root's private SSH key or MOAB key files
  - This information can be added securely as a post imaging step within scripts

Outline

- Introduction
- Important Security Considerations
- Job Schedulers
  - MOAB
  - Torque
- OpenMPI
- Distributed Filesystems
  - GPFS
  - Lustre
- Cluster Management
- **Local Privilege Escalation**

Local Privilege Escalation

- When all your users run code on “servers”, this is important
- Besides patch management and usual UNIX file permissions etc., pay attention to manufacturer added attack surface, e.g. SUID binaries
- On one manufacturer default install we found two non-standard SUID root binaries
- One looked particularly interesting as it had an “extract to directory” option

Local Privilege Escalation

- When run normally it lists something related to the local directory contents

```
0930292390394920testfile1.txt
0923902393902320testfile3.txt
2982983493898498testfile2.txt
```

- After much investigation, it turns out the cryptic numbers encode the ownership info and file permissions

- After further investigation, when supplied to the “extract” option, it modifies the local directory contents to match

Local Privilege Escalation

- …Oh dear! So if we encode something like this on our own system…

```
-rwsr-xr-x   1   root   root   136 Feb 26 2012   /bin/sh
```

- …and then extract the output on the target system

- We get a SUID root shell. Nice ☺

- Neither SUID binary even appeared to be needed, let alone require SUID permissions!
  - Remove unnecessary attack surface

Summary

- Interesting Challenges
  - Performance and security
  - Authenticating users across multiple nodes securely

Questions?

http://labs.mwrinfosecurity.com | @mwrlabs

John.fitzpatrick@mwrinfosecurity.com | @j0hn__f

Luke.jennings@mwrinfosecurity.com