
Simunix, a large scale platform simulator

David Glessner

Adrien Faure

2015-11-19

Yet another Slurm simulator

- We want to:
 - Test new developments
 - Test new clusters
 - Test new configurations
 - Develop machine learning algorithms

Yet another slurm simulator

- Previous works to test slurm:
 - Production cluster
 - Small dev cluster
 - Virtual machines
 - Multiple-slurmd (multiple slurm nodes per real node)
 - Slurm simulator

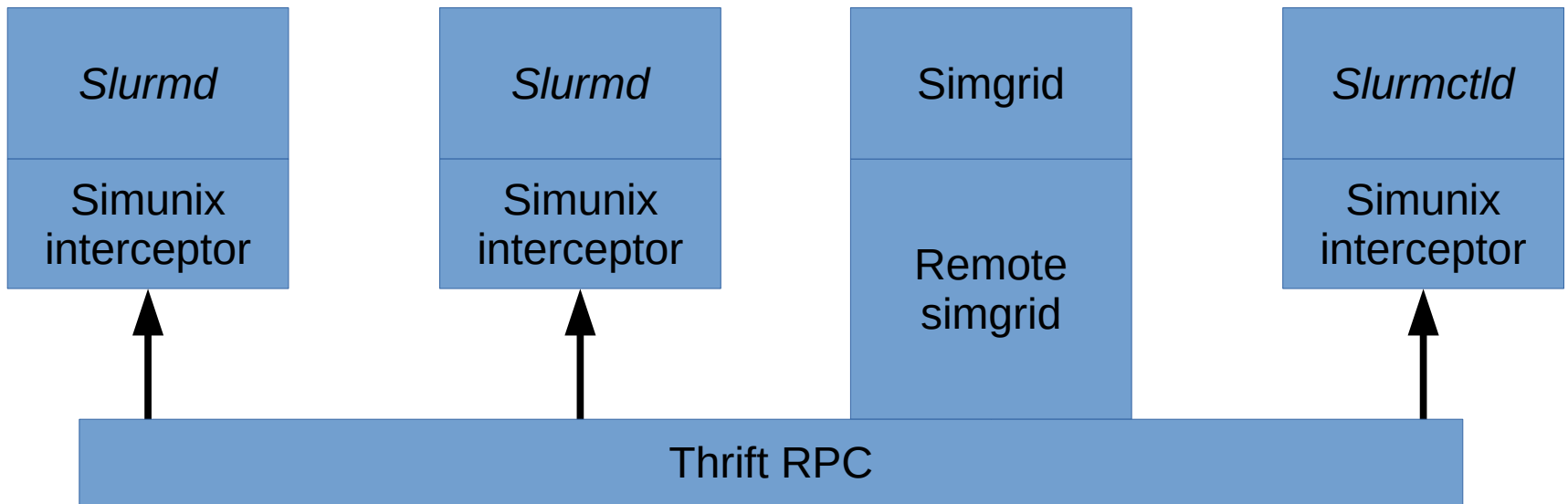
Yet another slurm simulator

- Previous works to test slurm:
 - Production cluster **impossible!**
 - Small dev cluster **too small!**
 - Virtual machines **too heavy!**
 - Multiple-slurmd **limited!** (eg. network contention)
 - Slurm simulator **limited!** (eg. the code of slurm is modified)

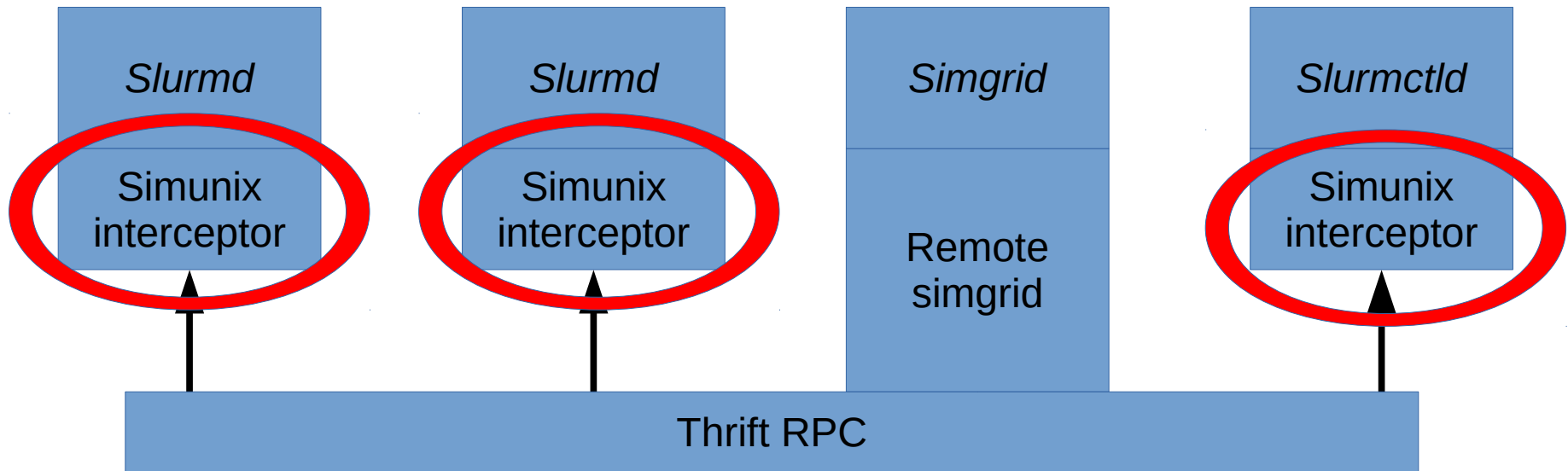
Our solution

- We simulate the underlying platform, not slurm!

The architecture



The architecture



The architecture – interceptors

- They implement the "UNIX" API: pthreads, pthread_mutex, gettimeofday, sleep, send, recv...
- Then, they traduce UNIX calls to calls to the simulaltor

The architecture – interceptors

How to intercept function calls?

- Change how linking is done!
- The Linux linker load from the system and LD_PRELOAD the needed shared libraries
- It fills the GOT (Global Object Table) with the address of each functions of each libraries
- The compiler compile

```
sleep(10);
```

to

```
GOT["sleep@libc"] (10);
```

(Of course, it's not exactly like this, if you have more question RTFM of the ELF format)

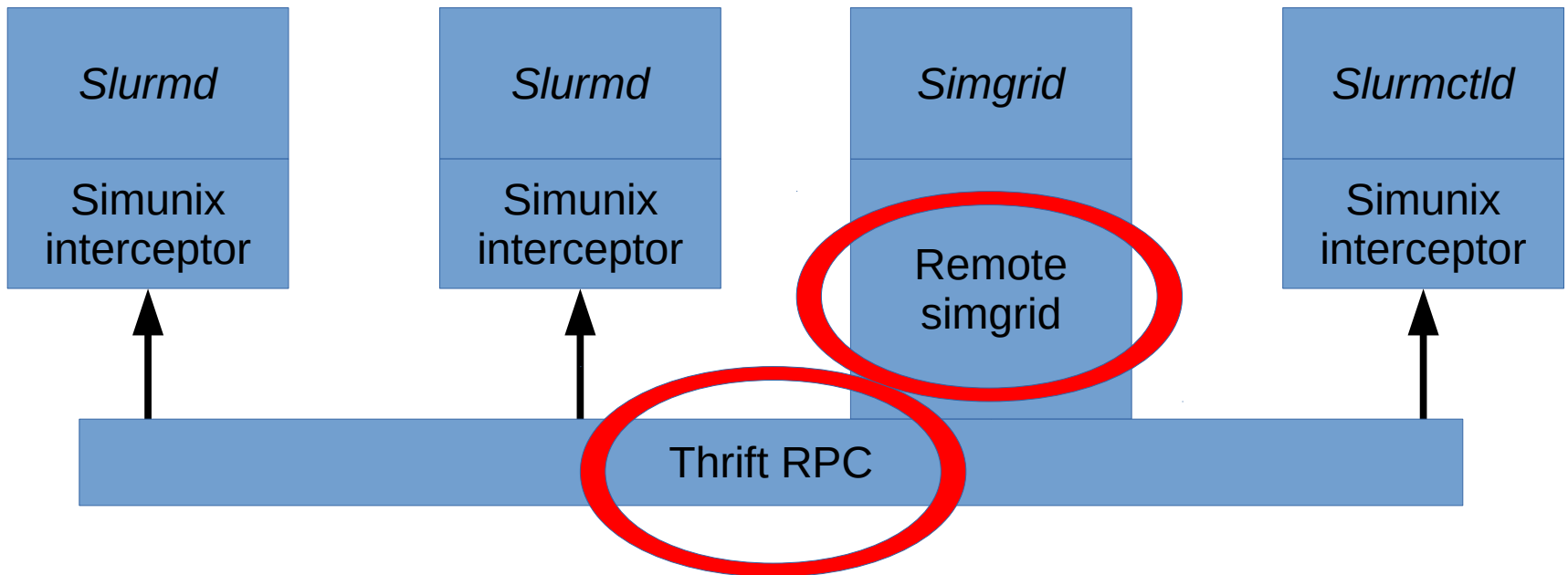
The architecture – interceptors

How to intercept function calls?

- Change how linking is done!
- At runtime, simunix rewrite the GOT
 - Of the selected binary/libraries
 - Not on the simunix library nor the Simgrid library!
 - Addresses in the GOT are replace by our own functions:

```
GOT["sleep@libc"] = &simunix_sleep;  
GOT["time@libc"] = &simunix_time;  
...
```

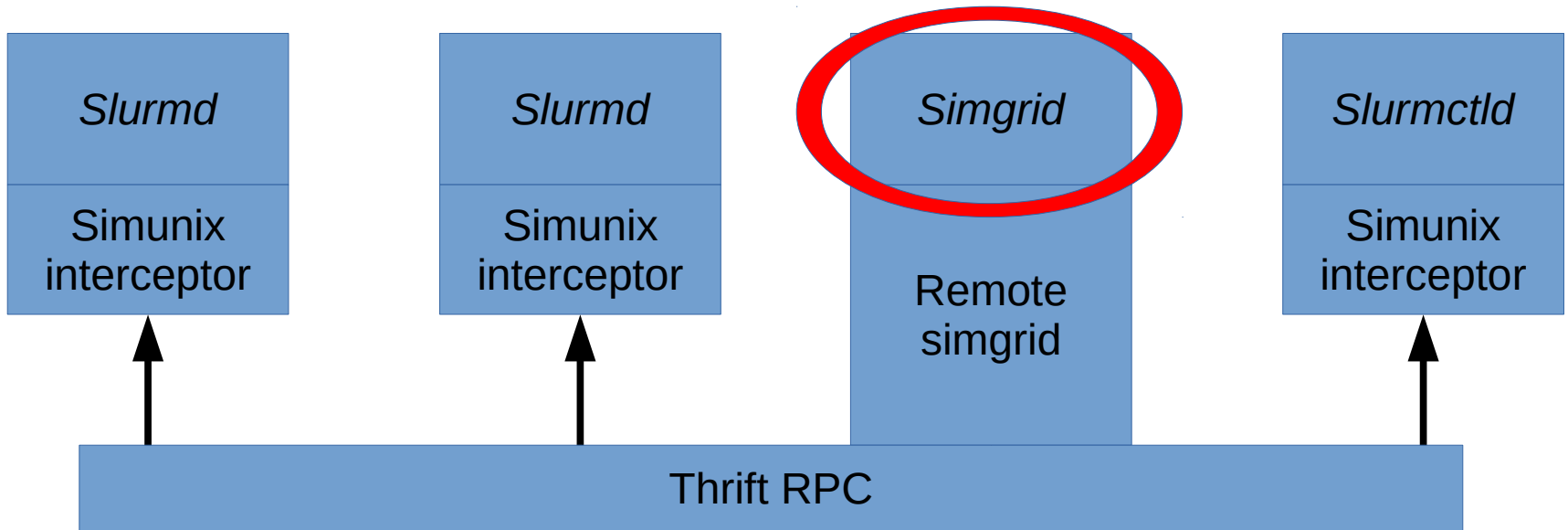
The architecture



The architecture – Thrift RPC

- *Apache Thrift* is an open-source library to build RPC client and server
- From a custom language, it generates files to build the client and the server
- We use it to transfer calls from the interceptors to the simulator
- *Remmote-simgrid* is our code to use thrift with simgrid

The architecture



The architecture – Simgrid

Simgrid

- a framework to design simulators of distributed applications
- Supports:
 - advanced network models
 - energy consumption models
 - I/O models
- Actively developed
- Good practice : they (in)validate their simulator (they explicitly give the strengths and weaknesses of their models by testing them and compared them to real runs)



How to start a simulation?

How to start a simulation?

Specific to slurm:

- A slurm installation
- A slurm configuration

Specific to Simgrid:

- A platform XML file (that describe the platform)
- A deployment XML file (that describe who will run where)

Specific to the experiment:

- A script to launch commands (sinfo, srun...)

Demo!

Some experiments already done

- Scalability: we are limited to ~50 nodes

```
1400000300
 Sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
debug*    up      infinite   50   idle dummy[0-49]
Start a job
Thrift: Sat Sep 24 14:09:23 2010 TServerSocket::listen() BIND 40909
slurmctld: Weighted Age priority is 0.000000 * 0 = 0.00
slurmctld: Weighted Fairshare priority is 0.000000 * 0 = 0.00
slurmctld: Weighted JobSize priority is 0.000000 * 0 = 0.00
slurmctld: Weighted Partition priority is 0.000000 * 0 = 0.00
slurmctld: Weighted QOS priority is 0.000000 * 0 = 0.00
slurmctld: Job 152 priority: 0.00 + 0.00 + 0.00 + 0.00 + 0.00 + 0 - 0 = 1.00
slurmctld: _slurm_rpc_submit_batch_job JobId=152 usec=1820
Submitted batch job 152
Wait 100...
slurmctld: sched: Allocate JobID=152 NodeList=dummy[0-49] #CPUs=50 Partition=debug
^C[host0:slurmd_dummy02_thread374_fork376_fork:(0) 300.288809] [simix_kernel/INFO]
```

(Bug spotted, fixed soon)

Some experiments already done

- Scalability: we are limited to ~ 50 nodes
- Run different slurm version (16.05.3

Some experiments already done

- Scalability: we are limited to ~ 50 nodes
- Run different slurm version (16.05.3, 15.08.9)

```
====LAUNCHER
1400000300
slurm 15.08.9
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
debug*    up      infinite   10   idle dummy[0-9]
Submit a job
Submitted batch job 2
Wait 100...
1400000400
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
debug*    up      infinite   10   alloc dummy[0-9]
$ squeue
          JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
           2      debug    job_1     root  R          1:40     10 dummy[0-9]
Wait 101...
Thrift: Sun Sep 25 19:42:31 2016 TServerSocket::listen() BIND 54700
1400000501
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
debug*    up      infinite   10   idle dummy[0-9]
====LAUNCHER END
Thrift: Sun Sep 25 19:42:37 2016 TServerSocket::listen() BIND 26262
```

Some experiments already done

- Scalability: we are limited to ~50 nodes
- Run different slurm version (16.05.3, 15.08.9 and 2.6.9)

```
=====LAUNCHER
1400000300
slurm 2.6.9
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
debug*    up      infinite   10   idle dummy[0-9]
Submit a job
Submitted batch job 2
Wait 100...
1400000400
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
debug*    up      infinite   1   drain dummy0
debug*    up      infinite   9   idle  dummy[1-9]
$ squeue
          JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
           2     debug    job_1     root PD        0:00     10 (ReqNodeNotAvail)
Wait 101...
1400000501
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
debug*    up      infinite   1   drain dummy0
debug*    up      infinite   9   idle  dummy[1-9]
=====LAUNCHER END
terminate called after throwing an instance of 'apache::thrift::transport::TTransportEx
```

Some experiments already done

- Scalability: we are limited to ~ 50 nodes
- Run different slurm version (16.05.3, 15.08.9 and 2.6.9)
- Speed: 10 nodes doing nothing for 4 days run in 1m30

Some experiments already done

- Scalability: we are limited to ~ 50 nodes
- Run different slurm version (16.05.3, 15.08.9 and 2.6.9)
- Speed: 10 nodes doing nothing for 4 days run in 1m30
- Interceptor size: about 20Mo added to each binary

Future work

Short term future works

- Remove bugs (scalability, srun)
- Ease of installation (provide a docker)
- Ease of use (4 files to start an experiment?!)
- Support databases
- Publish it (GPL)

Long term future works

- How close are we to the reality?
 - Optimize
 - Intercept /proc and related calls
 - Intercept IPMI or RAPL calls (to support energy)
 - Use it on other batch schedulers
-

Thanks

2015-05-07

