

Design and Evaluation of a Virtual Experimental Environment for Distributed Systems

L. Sarzyniec, T. Buchert, E. Jeanvoine, L. Nussbaum



27/02/2013
PDP 2013, Belfast

Many objects of study:

- Response time
- Scalability
- Fault-tolerance
- Throughput
- Robustness
- Fairness
- Performance
- Complexity
- etc.

Many experimental methodologies:

- *In-situ*: real applications on real platforms
Grid'5000, FutureGrid, PlanetLab, etc.
- Simulation: modeled applications on modeled systems
SimGrid, ns-2, OMNET++, etc.

In-situ methodology:

- 😊 more realistic
- 😊 uses real implementation
- 😞 limited to available environmental conditions
- 😞 usually unreproducible

Simulation:

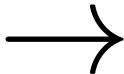
- 😊 enables unprecedented experiments
- 😊 perfectly reproducible
- 😞 simplified assumptions
- 😞 lower realism

Is there a **middle ground** methodology?

Technique used to efficiently simulate the behavior of a computer on another one, usually more powerful

Idea:

- 1 Use an existing platform
- 2 Model your desired platform
- 3 Efficiently emulate the desired platform using the real platform



Advantages:

- Combines advantages of simulation and *in-situ* approaches:
 - allows to use real applications and infrastructure
 - enables complicated experiments
- Paves the way to reproducibility

Answers following type of questions:

- How can I reproduce an experiment published in 2001 even if 1.5GHz processors do not exist anymore?
- How can I evaluate my new P2P software designed for DSL networks?
- How does this runtime with advanced load-balancing capabilities perform on highly hierarchical networks?

During the rest of the talk I will:

- 1 present our emulation-based solution
- 2 describe its architecture
- 3 show and discuss evaluation results
- 4 conclude and outline future work



Distem is a (freely available) software to build virtual distributed experimental environments.



Heterogeneous nodes,
Long distance networks,
Grid, Cloud, P2P,

...

What can Distem do for you?

Features of Distem include:

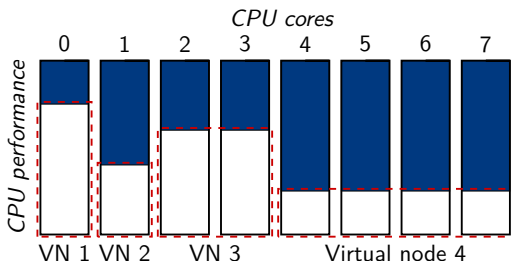
- **Introducing heterogeneity** in otherwise homogeneous cluster:
 - **CPU heterogeneity**
How does your solution perform when some nodes are slower?
 - **Network heterogeneity**
Does your solution work in Internet-like infrastructure?
- **Emulating complex network topologies**
How does your solution perform on a Grid?
- **Enlarging the scale of the experiment**
How does your solution perform on several thousands of nodes?
- **User-friendliness**

Distem can host **multiple virtual nodes** on one **physical node**:

- with a different number of cores
- with different CPU performance

There are 2 strategies for degrading performance:

- CPU-Gov – based on hardware CPU throttling
- CPU-Hogs – advanced CPU burning



Distem can emulate properties of network links between nodes.

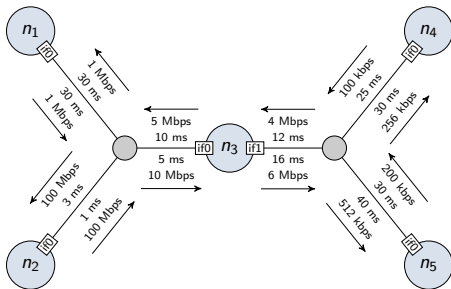
Each link can have a different:

- **maximum bandwidth**
- **latency**

They can be set for incoming and outgoing traffic independently.

Complex network configuration

- Define **properties of network links** using network heterogeneity
- Use them to emulate **several local networks** linked together



Distem uses a lightweight virtualization to:

- **share resources** between nodes:
 - CPU
 - network
 - filesystem
- host many instances of **virtual nodes** on a single node

This powerful feature:

- enables challenging experiments of unprecedented scale
- saves resources and energy

Distem strives to be user-friendly:

- complex and tedious tasks are automated:
 - configuring network interfaces
 - populating routing tables
 - distributing system images
 - etc.
- 3 interfaces with increasing complexity and feature-set are offered:
 - command-line
 - Ruby library
 - REST interface (with JSON to represent data)

- **Command-line interface** – distem command

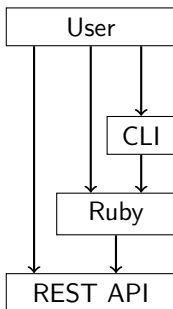
- 😊 Easy to use
- 😞 Hard to automate
- 😞 No access to more advanced features

- **Ruby library**

- 😊 Easy to automate
- 😊 Access to all features
- 😊 Easy to use (if you know Ruby)
- 😞 Requires Ruby

- **REST API**

- 😊 Language agnostic
- 😞 Requires REST knowledge



CLI example:

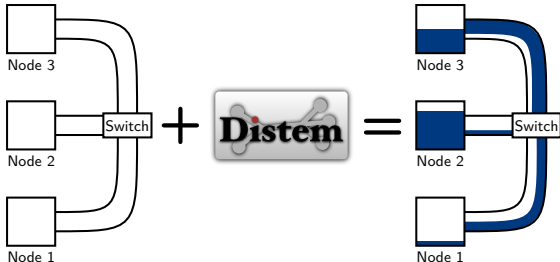
```
distem --create-vnetwork vnetwork=net,address=10.144.0.0/22
distem --create-vnode vnode=node-1,rootfs=file:///image.tgz
distem --create-viface vnode=node-1,iface=if0,vnetwork=net
distem --start-vnode node-1
distem --execute vnode=node-1,command="hostname"
```

Distem uses **modern Linux features**:

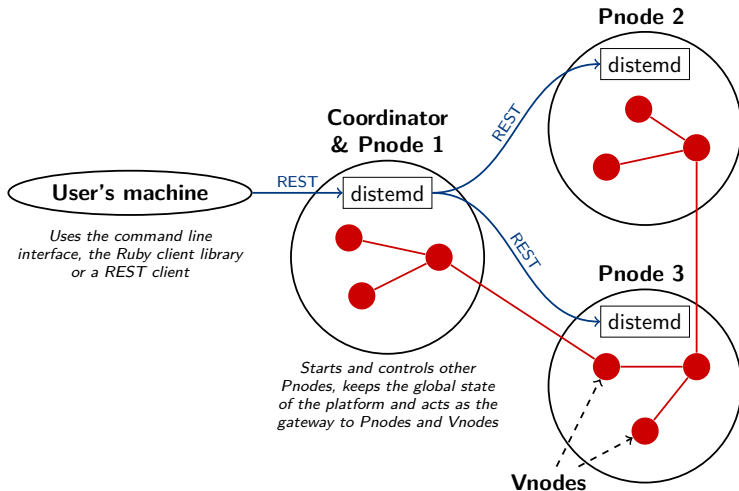
- Control Groups and Linux containers (LXC)
- CPU frequency scaling
- advanced networking (network bridging)
- network traffic control (packet schedulers and shapers)



Note that it limits the scope of experiments to Linux/Unix.



Communication architecture



To evaluate Distem we designed a few experiments concerned with:

- network emulation:
 - precision of latency emulation
 - latency emulation over time
 - precision of bandwidth emulation
 - emulation of a simple topology
 - basic performance analysis of scp and rsync tools
- CPU emulation (Linpack, DGEMM and FFT benchmarks)
- scalability (by performing a large deployment with Distem)

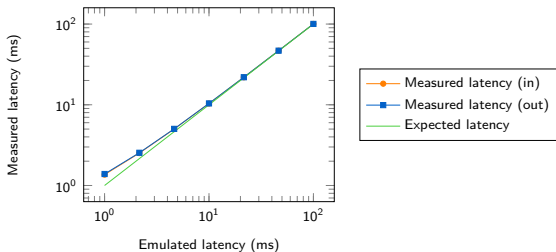
Each measurement was repeated many times and results are averaged.

We used the Grid'5000 testbed.



Precision of latency emulation

- Purpose: test if latency is properly emulated
- Setup:
 - 2 physical nodes, 1 virtual node on each
 - Emulated latencies from 1 ms to 100 ms
- Data point: RTT (custom ping tool) between two virtual nodes

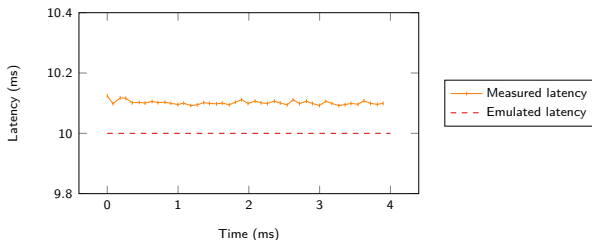


Conclusion

Emulation is accurate, especially for values above real network latency (0.3 ms).

Latency emulation over time

- Purpose: test if latency is constant over time
- Setup:
 - 2 physical nodes, 1 virtual node on each
- Data point: result of high-frequency RTT probing

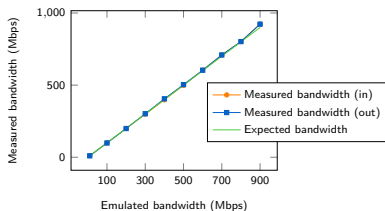
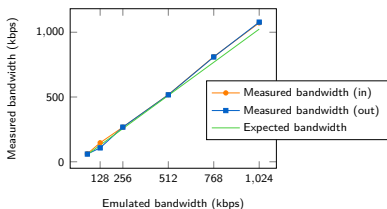


Conclusion

Emulation is stable and correct during the measurement.

Precision of bandwidth emulation

- Purpose: test if bandwidth emulation is correct
- Setup:
 - 2 physical nodes, 1 virtual node on each
 - Two bandwidth ranges: [56 Kbps, 1Mbps] and [50 Mbps, 1Gbps]
- Data point: bandwidth between two nodes (using iperf)

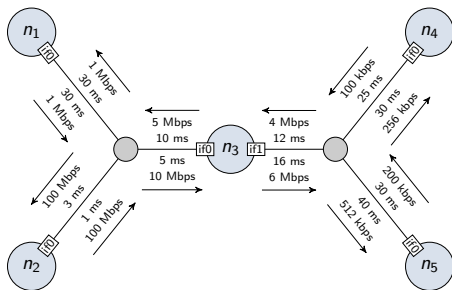


Conclusion

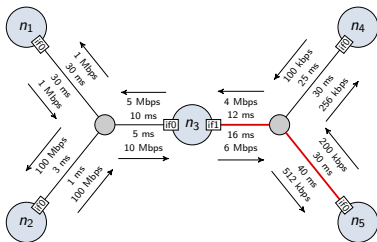
Bandwidth emulation is accurate in both situations.

Emulation of a simple topology

- Purpose: test if Distem properly emulates non-trivial network
- Setup:
 - simple topology created with Distem
 - heterogeneous network links
- Data point: bandwidth and RTT between each pair of nodes



Emulation of a simple topology (cont.)



From \ To	n_1	n_2	n_3	n_4	n_5
n_1	-	0.06 s / 1.07 Mbps	0.08 s / 1.07 Mbps	0.16 s / 0.29 Mbps	0.17 s / 0.55 Mbps
n_2	0.06 s / 1.04 Mbps	-	0.02 s / 9.57 Mbps	0.10 s / 0.29 Mbps	0.12 s / 0.55 Mbps
n_3	0.08 s / 1.05 Mbps	0.02 s / 4.92 Mbps	-	0.08 s / 0.30 Mbps	0.10 s / 0.57 Mbps
n_4	0.16 s / 0.17 Mbps	0.10 s / 0.16 Mbps	0.08 s / 0.15 Mbps	-	0.13 s / 0.15 Mbps
n_5	0.17 s / 0.26 Mbps	0.12 s / 0.27 Mbps	0.10 s / 0.27 Mbps	0.13 s / 0.26 Mbps	-

$$RTT_{n_3 \rightarrow n_5} = RTT_{n_5 \rightarrow n_3} = (16 \text{ ms} + 40 \text{ ms}) + (30 \text{ ms} + 12 \text{ ms}) = 98 \text{ ms} \approx 0.1 \text{ s}$$

$$BW_{n_3 \rightarrow n_5} = \min \{6 \text{ Mbps}, 512 \text{ kbps}\} \approx 0.57 \text{ Mbps}$$

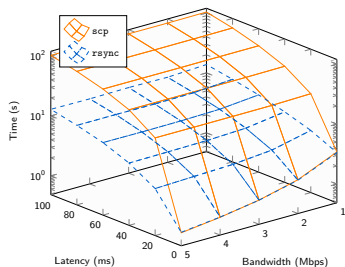
$$BW_{n_5 \rightarrow n_3} = \min \{200 \text{ kbps}, 4 \text{ Mbps}\} \approx 0.27 \text{ Mbps}$$

Conclusion

Emulation is correct: bandwidth between each pair of nodes is a minimum of link bandwidths, and RTT is a sum of all latencies on the path.

Performance analysis of scp and rsync tools

- Purpose: compare scp and rsync when transferring Distem sources
- Setup:
 - 2 physical nodes, 1 virtual node on each
 - Emulated latency (from 0 to 100 ms) and bandwidth (from 1 to 5 Mbps)
 - scp and rsync are *unchanged*
- Data point: time needed to transfer 700 KBs (small files)

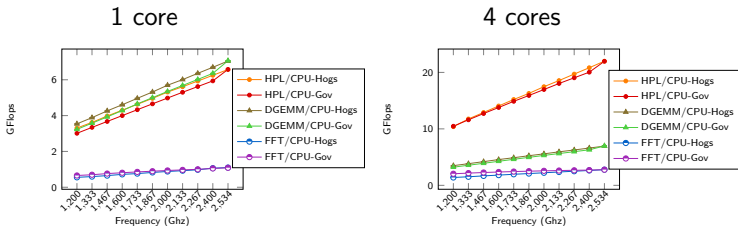


Conclusion

rsync outperforms scp thanks to a more efficient transfer of many files.

CPU emulation

- Purpose: compare CPU emulation strategies available in Distem
- Setup:
 - 1 physical node with 1 virtual node, 1 or 4 cores
 - Emulated frequencies from [1.2 GHz, 2.54 GHz] range
- Data point: each benchmark result

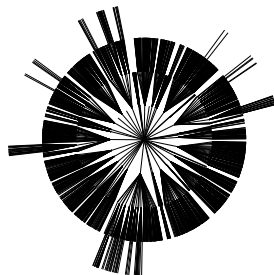


Conclusion

Even with real benchmarks (CPU & memory usage), the CPU-Hogs strategy (usable without any hardware support) behaves as expected.

- Purpose: test scalability of Distem
- Setup: 25 or 100 physical nodes
- Data point: time required to deploy from 500 to 5000 virtual nodes and to launch a taktuk command (command execution tool using tree topology)

























TakTuk communication topology with 2560 virtual nodes (10 physical nodes)



# Vnodes \ # Pnodes	Distem installation (s)		Virtual nodes deployment (s)		TakTuk command (s)	
	25	100	25	100	25	100
500	56.9	95.4	53.7	51.7	2.4	2.1
1000	56.4	94.7	94.3	91.1	3.4	3.8
2500	59.7	95.8	219.2	207.7	10.3	9
5000	64	97.1	445.5	410.7	21.3	21.3

Conclusion

- Deploying 5000 virtual nodes takes less than 10 minutes
- One physical node can host many virtual nodes efficiently

	Network emulation	CPU emulation	Complexity	Maintained / Available
Emulab				
Wrekavoc				
PlanetLab				
ModelNet				
DieCast				
Distem				

Distem is the only tool offering advanced emulation and low complexity. Moreover it is freely available and actively maintained.

In this talk I presented Distem. It features:

- Emulation of resources:
 - Network parameters and topology
 - CPU performance
- Scalability and efficiency:
 - Sharing of resources using virtualization
 - Thousands of nodes can be deployed in a few minutes
- Easy way to design and run advanced experiments:
 - Supports reproducibility (by saving topology information)
 - Ready to use on Grid'5000

In the near future we plan to:

- Push scalability even further
- Emulate more properties: memory, advanced CPU features

More information on <http://distem.gforge.inria.fr/>