Testbeds Support for Reproducible Research

Lucas Nussbaum
lucas.nussbaum@univ-lorraine.fr

Reproducibility workshop at SIGCOMM’2017
Reproducibility 101

Analysis

Experiments

Measured Data

Analytic Data

Computational Results

Presentation Code

Figures

Tables

Numerical Summaries

Published Article

Text

Protocol
(Design of Experiments)

Scientific Question

Inspired by Roger D. Peng’s lecture on reproducible research, May 2014
Improved by Arnaud Legrand
Reproducibility is not only about data analysis
Experimentation (inc. testbeds and services) has a key role
Testbeds

Many different testbeds – main differences:

- Focus (object of study), kinds of resources
  - From wireless sensors to physical servers

- Level of access and control for experimenters
  - Use of virtualization technologies vs bare-metal reconfiguration

- Guarantees on the overall environment
  - Multi-tenancy on servers and network links, stability over time

This talk:

1. A short, non-exhaustive panorama of testbeds
2. A comparison of support for reproducibility on three similar testbeds: Chameleon, CloudLab, Grid’5000
PlanetLab (2002 → ~2012)¹

- 700-1000 nodes (generally two per physical location)
- Heavily used to study network services, P2P, network connectivity
- Users get slices: sets of containers
- Follow-ups: Planet-Lab Europe, Nornet (+ Mobile Broadband)
- Limitations:
  ♦ Shared nodes (varying & low computation power)
  ♦ Real(?) Internet:
    ★ Unstable experimental conditions → statistics for reproducibility
    ★ Nodes mostly connected to GREN → not really representative

Emulab (2002 → today)$^2$

- Use a cluster of nodes with many network interfaces
- Configure the network on the fly to create custom topologies
  - With link impairment (latency, bandwidth limitation)
- Emulab: a testbed at Univ. Utah, and a software stack
  - Deployed on dozens of testbed world-wide (inc. CloudLab)
  - In Europe: IMEC’s Virtual Wall (Ghent, Belgium)

Internet of Things: FIT IoT-Lab

- 2769 wireless sensors (from WSN430 to Cortex A8)
- 7 sites (Grenoble, Lille, Strasbourg, Saclay, Rennes, IMT Paris, Lyon)
- Also mobile robots
- Typical experiment: IoT communication protocols

https://www.iot-lab.info/

Wireless (WiFi, 4G/LTE, SDR): CorteXlab⁴, R2lab

- Sets of customizable wireless nodes in an anechoic chamber
- For experiments on wireless protocol stacks

http://www.cortexlab.fr
https://r2lab.inria.fr

Software Defined Networking: OFELIA

- Set of sites (*islands*); each site hosts OpenFlow-enabled switches
- Users control their OpenFlow controller, and VM to act as sources/sinks

---

Internet measurements: RIPE ATLAS

- 9700 probes
- For network measurements: ping, traceroute, DNS, SSL/TLS, ...
Clouds, data centers

- Discussed in the second part of this talk
Federations of testbeds

- **Identity-level federation**
  - Enable users to use several testbeds with same credentials

- **API-level federation**
  - Provide the same interface on/for several testbeds

- **Data-plane federation**
  - Combine resources from several testbeds during an experiment
  - Two main use cases:
    - Different testbeds (e.g. Cloud/Edge scenarios, with experiment control at both ends)
    - Similar testbeds → more resources, geographically distributed
The flagship project of testbed federation

A large-scale distributed testbed, or a tightly integrated federation of **aggregates**, providing either compute resources (**racks**) or networking

- InstaGENI racks (32 currently):
  - Descendant from the Emulab software stack
  - Providing VMs (Xen) or raw PCs
  - HP hardware

- ExoGENI racks (12 currently):
  - VMs using OpenStack, or Xen, or OpenVZ
  - Some racks with bare-metal nodes (**xCAT**)
  - IBM hardware

- AL2S, MAX: providing network interconnection between racks

Also the main developer of the GENI API, used by other federations

---

Fed4FIRE

- European federation of about 20 testbeds
- Diverse: wired networking, wireless/5G, IoT, OpenFlow, Cloud
- Follow-up project (Fed4FIRE+) started in 2017

https://www.fed4fire.eu/
Comparing Chameleon, CloudLab and Grid’5000

- Similar scope: *Internet of data centers* (Cloud, Big Data, HPC)
  - Cloud & Big Data: design and evaluation of custom cloud stacks
  - HPC: availability of HPC networks and accelerators

- Similar architecture: sites (racks of servers) interconnected with a dedicated network $\sim$ *in-vitro experimentation*
  - Little or no influence from the outside world
Comparing Chameleon, CloudLab and Grid’5000

- Similar scope: *Internet of data centers* (Cloud, Big Data, HPC)
  - Cloud & Big Data: design and evaluation of custom cloud stacks
  - HPC: availability of HPC networks and accelerators

- Similar architecture: sites (racks of servers) interconnected with a dedicated network \( \sim \text{in-vitro experimentation} \)
  - Little or no influence from the outside world

- Different design choices and history:
    - Software stack: mostly custom developments, since 2003
    - Established testbed (8 sites, 800 machines, 500+ users/y)
  - **CloudLab** – [https://www.cloudlab.us/](https://www.cloudlab.us/), USA, 2014
    - Based on the Emulab codebase
    - Three main sites, 1081 servers, federated with other instances
  - **Chameleon** – [https://www.chameleoncloud.org/](https://www.chameleoncloud.org/), USA, 2014
    - Based on OpenStack + Grid’5000 tools + custom developments
    - Two sites, 424 nodes
Support for reconfiguration

- Goals:
  - Enable experimenters to set up a custom experimental environment
  - Later, recreate the same experimental environment \(\sim\) repeatability

- Nodes: support for installing a custom software environment available
  - Different tools, but providing similar functionality:
    - Frisbee (CloudLab), Ironic (Chameleon), Kadeploy (Grid’5000)

- System images generation:
  - **Grid’5000**: using Kameleon\(^7\)
    - Set of recipes (published in Git)
    - Caching of downloaded artifacts
    - Can serve as a basis for users’ own images
  - **Chameleon**: using diskimage-builder, source code on GitHub\(^8\)
  - **CloudLab**: no documentation of the process

---


Support for reconfiguration: networking

- **CloudLab**: advanced support for networking experiments
  - Custom topologies and network emulation

- **Grid’5000**:
  - Custom topologies can be created using KaVLAN
  - No high-level tool; no integration of network emulation

- **Chameleon**:
  - Limited to what is provided by OpenStack Neutron (VLAN-based)\(^9\)
  - Suitable for network isolation, not really for topologies

Support for collecting provenance

- **Goals:**
  - Understand the experimental environment (hardware, network)
  - Document it \( \leadsto \) repeat, replicate, reproduce

- **Requirement: documentation**
  - **CloudLab:** textual documentation (web pages), and AM API
  - **Chameleon and Grid’5000:** same solution\(^{10}\)
    - Detailed description of all resources as JSON documents (REST API)
    - Automatically verified on a regular basis (hardware inventory tools, regression tests)
    - Archived (stable reference)
    - Web interface to discover resources

\(^{10}\)David Margery et al. “Resources Description, Selection, Reservation and Verification on a Large-scale Testbed”. In: *TRIDENTCOM. 2014.*
Support for long-term data storage

▶ Goals:
  ♦ Store large datasets used during experiments
  ♦ Preserve artifacts generated during the experiment

▶ Various services on all three testbeds:
  ♦ **Chameleon**:
    ★ File-based object store (OpenStack Swift)
  ♦ **CloudLab**:
    ★ File- and block-stores, with versioning and snapshotting (ZFS)
  ♦ **Grid’5000**:
    ★ Files: NFS-based service
    ★ Block and objects: managed Ceph clusters

▶ No way to expose that data on the Web
  ♦ A task for external data repositories?
Support for automation

- Goal: contribute to repeatability and replicability by providing ways to automate experiments

- Low-level: APIs for experimenters on all three testbeds, to discover, reserve and setup resources
  - **CloudLab**: SFA AM API (GENI)
  - **Chameleon**: OpenStack APIs
  - **Grid’5000**: custom REST API (SFA AM API is WIP)

- High-level: experiment orchestration tools
  - **CloudLab**: profiles
  - **Chameleon** appliances
  - **Grid’5000**: various tools available, including integrated solutions for OpenStack and Ceph
Open questions

- Respective responsibilities of testbeds and experimenters
  - Especially for automation and monitoring

- Load generation and faults injection in in-vitro testbeds
  - Lack of generators and traces

- Standardization and federation of efforts
  - Standard APIs, reproducibility check lists
Main takeaways

▶ Many testbeds available
  ♦ Often with a fairly open access policy
  ♦ Using them is a good way to help repeatability and replicability
    ★ They should be developed as *public goods* for our community

▶ Some testbeds have good support for reproductibility
  ♦ But there’s more work needed in that area