ACM REP24 Tutorial: Reproducible distributed environments with NixOS Compose

Dorian Goepp, Fernando Ayats Llamas, Olivier Richard, Quentin Guilloteau

ACM REP24 Tutorial: Reproducible distributed environments with NixOS Compose

Dorian Goepp[1], Fernando Ayats Llamas[1], Olivier Richard[1], and Quentin Guilloteau[3]

[1]Univ. Grenoble Alpes, Inria, CNRS, LIG, F-38000 Grenoble, France
[3]University of Basel, Basel, Switzerland

1 Abstract

Developing software environments for experiments is an iterative, tedious, and time consuming process, even more when aiming for reproducibility. A widespread approach is to rely on self-contained software environments called images, be it containers, virtual machines, or tarballs (as used on Grid'5000). Practitioners usually build their image multiple times, adding every time a forgotten dependency or fixing a previously added one. Worst, the build process of such image often does not stand the test of times: when we update the apt repositories and install packages in a dockerfile naïvely, the downloaded package version will evolve over time and the image shall eventually fail to build when the package repositories are no longer supported upstream. In the case of Grid'5000, the building time of such images takes around ten minutes for full system tarballs, which does not encourage the experimenters to follow good reproducible practices when setting them up. As a result, those images cannot be rebuilt nor modified by someone else.

In this tutorial, we introduce the users to NixOS Compose [5], a tool based on Nix [3] and NixOS [4] to generate and deploy reproducible environments on distributed platforms. We will first present Nix and the notions required to use NixOS Compose. As NixOS Compose can target several platforms, the users will set up their environment with lightweight containers (docker) on their local machines, allowing them to iterate quickly on their environment description. Once the environment ready with containers, users will be able to quickly test it on the Grid'5000 testbed [1] using `kexec`, before generating a full system tarball.

2 Topic & Relevance

As the scientific community is traversing a reproducibility crisis, and the computer science field does not make an exception, there is a global trend to improve the reproducibility practices of scientists and professionals. This task has proven harduous for distributed software environments due to their complexity and their frequent requirement for complicated software stacks. All it takes is one missing dependency in these stacks to lead to difficult or even non-reproducible results in the future.

Functional package managers such as Nix [3] and Guix [2] address this issue by capturing every application's dependency. Such tools aim to rigorously describe a software environment, while keeping this information lightweight compared to opaque container or virtual machine images. Our tutorial focuses on Nix, the NixOS Linux Distribution [4] and our tool: NixOS Compose [5].

The NixOS Linux distribution, which extends the Nix paradigm to the whole operating system, improves the Quality-of-Life of experimenters who also need to control the entire system (kernel, firmware, services, etc.). Also, building deployment images for any kind of system and environment (such as Kubernetes or Grid'5000) is a time-consuming iterative process. Hence the introduction of NixOS Compose (based on NixOS). It captures the definition of a distributed system that can be realised locally with lightweight tools (containers and Virtual Machines) as well as remotely on physical hardware (e.g., Grid'5000). Practitioners run systems locally in the development phase, allowing a fast iteration cycle and then can later generate images and deploy remotely in the experimental phase. This tool aims to reduce the friction of developing clean, reproducible, distributed environments while allowing virtually anywone to achieve the same system configuration.

We believe this tutorial will benefit AMC REP attendees, assisting them in improving the reproducibility of their experiments and executions in single nodes or distributed setups.

3 Format, Audience & Content

Format: The tutorial will last **two slots of 3 hours (6 hours in total)**, and will be in **English**. The first slot will introduce Nix and its concepts through examples and common usage. In the second slot, the attendees will use NixOS Compose to produce a distributed environment for an experiment. Attendees will be granted a temporary access to the Grid'5000 testbed to deploy their environment on physical machines.

This tutorial would work in an hybrid format.

Audience: Attendees will need a Linux or MacOS laptop with an Internet connection, as well as root privileges on their machine (required to install Nix). No knowledge of Nix nor NixOS is needed, but basic knowledge of the Linux environments and tools as well as basic notions of functional programming would be beneficial.

Content: In this tutorial, attendees will *(i)* learn about Functional package managers and Nix, *(ii)* create reproducible packages with Nix, *(iii)* bundle packages in a reproducible and sharable software environment with Nix, *(iv)* create reproducible docker image from Nix packages, *(v)* create reproducible NixOS image, and *(vi)* use NixOS Compose to create and deploy reproducible and distributed NixOS environments on Grid'5000.

4 Previous Editions

History: There are have been four previous editions for the Nix tutorial: `https://nix-tutorial.gitlabpages.inria.fr/nix-tutorial/index.html`. And two previous editions of the NixOS Compose tutorial: `https://nixos-compose.gitlabpages.inria.fr/tuto-nxc/`.

Novelty: We will present NixOS Compose and take an example of distributed experiment to build a reproducible environments.

5 Organizers

Dorian Goepp has learned the hurdles of reproducibility in distributed systems through five years as a robotics research engineer, following his first M.S. in advanced robotics (EMARo) obtained in 2015. Since 2023, he is pursuing a Ph.D. in Computer Science at the University of Grenoble (France) on the topic of tools for reproducibility for distributed systems.

Fernando Ayats Llamas is a Research Engineer at Inria, working on packaging and continuous integration for supercomputers. He completed his MSc in Research in Computer Systems Engineering at the University of Cádiz (2023). He is also highly interested in functional package managers and reproducible software deployments.

Olivier Richard is an Associate professor at Grenoble Alpes University. His research interests are focused on system architecture for high performance computing and large distributed system. He also works on tools and methods to enhance experiments' reproducibility in these domains. He co-designed the Grid'5000 national testbed (head of Grenoble's site since 2003).

Quentin Guilloteau holds a Ph.D. in Computer Science from University Grenoble Alpes in France (2023). He is currently a Post-doc at the University of Basel in Switzerland working on multi-level scheduling. He is also interested in reproducible research and autonomic computing in HPC.

Acknowledgments

Jonathan Bleuzen, Adrien Faure, and Millian Poquet helped writing this tutorial.

References

[1] D. Balouek, A. Carpen Amarie, G. Charrier, F. Desprez, E. Jeannot, E. Jeanvoine, A. Lèbre, D. Margery, N. Niclausse, L. Nussbaum, O. Richard, C. Pérez, F. Quesnel, C. Rohr, and L. Sarzyniec. Adding virtualization capabilities to the Grid'5000 testbed. In I. I. Ivanov, M. van Sinderen, F. Leymann, and T. Shan, editors, *Cloud Computing and Services Science*, volume 367 of *Communications in Computer and Information Science*, pages 3–20. Springer International Publishing, 2013.

[2] L. Courtès. Functional package management with guix. *arXiv preprint arXiv:1305.4584*, 2013.

[3] E. Dolstra, M. de Jonge, and E. Visser. Nix: A Safe and Policy-Free System for Software Deployment. page 14, 2004.

[4] E. Dolstra and A. Löh. Nixos: A purely functional linux distribution. *SIGPLAN Not.*, 43(9):367–378, sep 2008.

[5] Q. Guilloteau, J. Bleuzen, M. Poquet, and O. Richard. Painless transposition of reproducible distributed environments with nixos compose. In *2022 IEEE International Conference on Cluster Computing (CLUSTER)*. IEEE, 2022.