The Journal of Open Source Software

Container Tree: Software to Model Container Filesystems, Packages, and Inheritance

Vanessa Sochat[1]

1 Stanford University Research Computing

DOI: 10.21105/joss.01418

Software

- Review
- Repository
- Archive

Submitted: 21 March 2019
Published: 19 May 2019

Summary

Container technology is the unit of operation to ensure everything from scientific reproducibility to reliable enterprise deployment. However, despite this growing popularity, the actual study of containers, including inheritance structures, packaging, and filesystems, is a new problem that has not been addressed. Arguably, a reason for this lack of study comes down to missing software that can represent such structures. While container registries exist like Docker (Merkel, 2014) Hub, Quay.io, and Singularity Hub (Sochat, Prybol, & Kurtzer, 2017), their search functionality is limited in the detail they are able to provide, and their purpose is not to directly serve scientific pursuit. Here I present software, Container Tree that provides data structures well suited to directly study the inheritance, packaging, and filesystems of containers. Complete documentation and code is available ("Container-tree," 2018).

Container File Trees

The Container File Tree will create a tree that describes one or more containers. Specifically, each node in the tree is a folder or file on the path, with metadata stored such as the number of containers represented at the node, and tags that can be used to store container identifiers or other metadata of interest. An actual container filesystem, appropriately, is structured in the same way. The Container Tree data structure adds easy methods to search, trace, and find particular files or folder, and also feeds directly into useful interactive visualizations to explore the filesystem. For a tree with more than one container or tag, the software also provides methods to calculate similarity. A tutorial is available.

Container Package Trees

Container Package Trees allow for the study of packages installed from popular managers Pypi and Apt, including names and versions across one or more containers. By way of the Container Diff ("Container-diff," 2017) open source tool, the user can quickly extract package information for one or more containers to create a package tree. The package trees are ideal data structures to store a representation of packages across containers. From the container Apt or Pip tree the user can easily compare containers based on packages within, or export a feature matrix with containers (rows) by packages (columns). A tutorial is available for the interested user.

Collection Trees

The container collection tree aims to help solve one of the more significant problems related to understanding containers: the problem that it's incredibly challenging to create an inheritance structure to understand how containers build upon one another. A container Collection Tree represents nodes as container namespaces, and each node contains a dictionary of the possible tags for the general namespace. The Container Collection Tree exposes functions to trace, search, and find containers based on a query of interest, and also export the tree structure as an actual filesystem. By way of mapping the inheritance tree to an actual filesystem, we can take advantage of linux command line tools (e.g., find, grep) to explore the space. Akin to the container file tree, the interested researcher can use the tree structure to calculate distances (similarity) between nodes (container collections) in the tree. A detailed example that creates, queries, and then exports a Collection Tree to the filesystem is also available.

The Container Tree software aims to directly help researchers to study different aspects of containers, an essential task if we are to better optimize their development, distribution, and function. More information on containertree, including tutorials, and classes is provided at the Container Tree official documentation. This is an Open Source project, and so feedback and contributions are welcome.

References

Container-diff. (2017). Github; https://www.github.com/GoogleContainerTools/container-diff.

Container-tree. (2018). Github; https://github.com/singularityhub/container-tree. doi:10.5281/zenodo.2602127

Merkel, D. (2014, March). Docker: Lightweight linux containers for consistent development and deployment. *Linux J.* Houston, TX: Belltown Media.

Sochat, V. V., Prybol, C. J., & Kurtzer, G. M. (2017). Enhancing reproducibility in scientific computing: Metrics and registry for singularity containers. *PLoS One*, *12*(11), e0188511. doi:https://doi.org/10.1371/journal.pone.0188511