DockerPedia: a Knowledge Graph of Docker Images

Maximiliano Osorio, Carlos Buil-Aranda, and Hernán Vargas

Informatics Department, Universidad Técnica Federico Santa María, Chile
{mosorio, cbuil, hvargas}@inf.utfsm.cl

Abstract Docker is the most popular implementation of Operating System virtualization, currently its online registry service (Docker Hub) stores more than 4.5 millions of software images. Using that registry it is possible to download and deploy Docker images as software containers. However, these images only show information of the main software, hiding the dependencies needed to run it. To allow users to track what they deploy into their machines, we developed DockerPedia, a resource that publishes information of the packages within the Docker images as Linked Data. Currently our resource includes 28% of the most downloaded images from Docker Hub providing information about the software dependencies and its vulnerabilities allowing to easily reproduce the environment in which each image was deployed as well as to check the security of the image without the need to download it.

1 Introduction

One of the most common ways for easily distributing and deploying software packages is through operating-system-level virtualization. This technology, also known as containerization, refers to an Operating System (OS) feature in which its kernel allows the existence of multiple isolated user-space instances called containers. The most popular virtualization technology is Docker (https://www.docker.com/), which implements software virtualization by creating minimal versions of a base operating system (a container). Docker containers can be seen as lightweight virtual machines that allow the assembling of a computational environment, including all necessary dependencies (libraries, configuration, data needed, etc). Docker distributes these computational environments as software images, allowing the user to easily deploy a server (e.g., Virtuoso), generate reproducible experiments (like in [2]) or even share movies and music. Public and private Docker images can be stored in Docker Hub (https://hub.docker.com/), using this registry it is possible to download and locally deploy Docker images executing thus the software within it. However, Docker Hub does not provide information about what packages are in the images, whether it will deploy correctly or if there are any security issues. Thus, Docker images work as a black box: users know the main software package that runs within the container but they do not have information about other packages included as dependencies to run it.

With DockerPedia we aim to put some light into the black box of running Docker containers. We analyzed 28% of the most downloaded images from Docker Hub, semantically described them and published these description as Linked Data. The descriptions include information about the packages (name, version, date of installation) and a security analysis of them. We linked all this data to the Debian public RDF package[1] database and to several public software vulnerabilities databases. The information provided from DockerPedia allow users to know the exact packages used in a Docker image, its vulnerabilities and, if necessary, reproduce the execution environment without the image itself.

2 Docker Overview

Docker builds software images by reading a set of instructions from a Docker File (a text file containing a set of commands), these files normally have multiple lines each of which are translated into a image layer when Docker builds the image. In this process commands are executed sequentially creating one layer after the another. This way, when an image is updated or rebuilt, only modified layers (i.e. modified lines) are updated.

The Docker Hub Docker Hub is an online registry for Docker image repositories. Currently stores more than 4.5 million images in two types of repositories: official and community. Official repositories contain verified images such as *Nginx, Red Hat* and *Docker*. Community repositories are not verified but can be created by any user or organization. Users can upload images to Docker Hub from other services such as GitHub or pushing their local images using the command line interface that Docker provides. Multiple images can be uploaded to the same repository specifying a different `tag`, in general this attribute is used to specify the image version, labeling the most recent one as *latest*. Uploaded images are available to use for members of the same organization or the whole community.

3 Docker Image Analysis

To extract the information from the Docker images hosted in Docker Hub we performed a search over its free text box to obtain all the Docker images. In February 2018, this search returned 1,363,510 Docker repositories and 4,608,443 images composed of 4,593,602 community images and 14,841 official images. The total size of these images is 53.47 PB. Due to the large amount of computational resources needed to examine such amount of data we only analyzed the 28% most downloaded Docker images from Docker Hub.

For each image we obtain the following information: image name, user name, repository name, image description, last update date, number of pulls (downloads) and number of stars (users rating). Note that we get all the versions of each image (i.e. all `tag` in each analyzed repository) and the information that

[1] https://packages.qa.debian.org/common/RDF.html

these includes such as the tag name, last updated date and image size. For instance, the Docker repository "google/cadvisor" has 59 different images of the same software, and each of these has different packages. We deploy the images to extract the information of the packages installed in it and, using the Clair software[2], we detect the vulnerabilities of each of them. To store the data from the security analysis Clair uses a relational database. However, this database is not available on the Web, thus it does not comply with any of the 5 stars for data publishing [1]. DockerPedia will publish this data and the information gathered from Docker Hub in the Web as Linked Data.

4 DockerPedia Data Publishing

4.1 The Docker Ontology

To publish all the data described in the previous section in the form of a knowledge graph and link it to the security vulnerabilities databases and the Debian package RDF store we used Morph [5], a Relational Database to RDF [3] engine. Since the RDB2RDF process needs an ontology, we developed a lightweight ontology to shape the relations between the different concepts we want to publish. The ontology imports classes and relations from the Docker Ontology [4] for some of the Docker concepts and the WICUS ontology [6] for the software experiment reproducibility concepts. The main classes in the ontology are: SoftwareImage (image from which a container is deployed), ImageLayer (a line within the Docker file that install software within the container), SoftwarePackage (packages installed at the ImageLayer), PackageVersion (the version of each package), SoftwareVulnerability (the vulnerabilities of each package) and the DockerFile class. DockerPedia resources are identified by the URI `http://dockerpedia.inf.utfsm.cl/resource` while the vocabulary is identified by the URI `http://dockerpedia.inf.utfsm.cl/ontology`. The resulting ontology is shown in the figure 1.

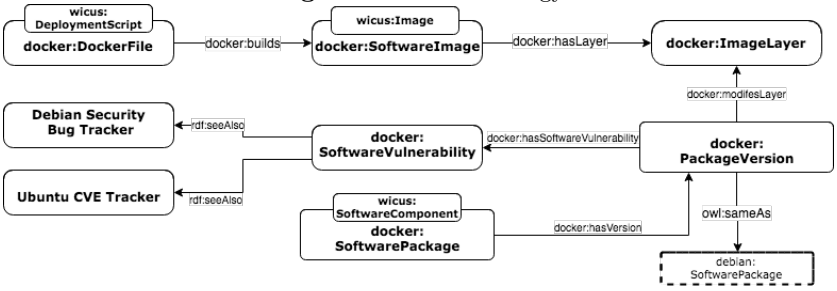

Figure 1. Docker ontology.

[2] Clair is an open-source tool from CoreOS designed to identify known vulnerabilities in Docker images. More information in https://github.com/coreos/clair

4.2 The DockerPedia Resource

The DockerPedia dataset contains information about 4,500,000 Docker Images stored in the Docker Hub portal. These Docker images contain several layers that install and remove software packages from the filesystem until the image is finally deployed as a Docker container. We analyzed all the packages of all the layers within the 28% most downloaded Docker images obtaining a total of 102.993.552 triples and 13.136 links to the Debian RDF package dataset.

Use cases: The goal of this resource is to allow users to understand what is inside the Docker images they run as a container on their hosts (allowing to easily reproduce the environment in which a software package was deployed). Examples of these use cases can be found at https://dockerpedia.inf.utfsm.cl/examples.

5 Conclusions

Throughout this paper, we have presented a resource that allows users to analyze Docker images before they run them into their hosts. We have gathered all Docker images stored at DockerHub and analyzed the 28% most downloaded of them. The analysis resulted in a dataset with more than 100 million triples, storing data about Docker images, software packages and their vulnerabilities, links to the Debian RDF package dataset and the vulnerabilities information pages. The data gathered in DockerPedia allow users to explore the packages and vulnerabilities of a Docker image without the necessity of deploying the container, making possible to reproduce the software execution environment even if the image itself is not available.

Acknowledgments: The authors are supported by the Fondecyt Project 11170714.

References

1. T. Berners-Lee. Is your linked open data 5 star. *Repéré à https://www. w3. org/DesignIssues/LinkedData. html*, 2010.
2. C. Boettiger. An introduction to docker for reproducible research. *SIGOPS Oper. Syst. Rev.*, 49(1):71–79, Jan. 2015.
3. S. Das, S. Sundara, and R. Cyganiak. R2rml: Rdb to rdf mapping language. w3c rdb2rdf working group, 2012.
4. D. Huo, J. Nabrzyski, and C. Vardeman. Smart container: an ontology towards conceptualizing docker. In *International Semantic Web Conference (Posters & Demos)*, 2015.
5. F. Priyatna, O. Corcho, and J. Sequeda. Formalisation and experiences of r2rml-based sparql to sql query translation using morph. In *Proceedings of the 23rd international conference on World wide web*, pages 479–490. ACM, 2014.
6. I. Santana-Perez, R. F. da Silva, M. Rynge, E. Deelman, M. S. Pérez-Hernández, and O. Corcho. Reproducibility of execution environments in computational science using semantics and clouds. *Future Generation Computer Systems*, 67:354–367, 2017.