DTN tests run in a Dockerised environment:
A Blessing or a Curse?

Iacovos Ioannou
(CYNET)

GÉANT Infoshare - Data Transfer Nodes: How Fast can your Data Travel?
www.geant.org
In this presentation, we will cover

1. What is Docker
2. Docker architecture
3. Containers vs Virtual Machines
4. Data Transfer Nodes (DTNs) meet Docker
5. Docker Installation with DTN Software Containers in GTS
6. Results of GÉANT Testbed Service (GTS) Dockerised DTN Tests
7. Scripts and Manuals provided for Dockerised DTN test
8. Demonstration
What is Docker

- **Docker** is a set of Platforms as a Service (PaaS) products that uses OS-level virtualisation to deliver software in packages called containers.

- Containers are isolated from one another and bundle their own software, libraries and configuration files; they can communicate with each other through well-defined channels.

- All containers are run by a single operating system kernel and therefore use fewer resources than virtual machines.

- The service has both free and premium tiers. The software that hosts the containers is called Docker Engine.
Docker architecture

The Docker architecture is a client-server architecture. The client communicates with the Docker daemon for:

- **Building** of containers
- **Execute/run** the containers
- **Distribute** the containers

The Docker client and daemon communicate using a Rest API, over UNIX sockets or a network interface.

https://www.docker.com/blog/introducing-docker-engine-18-09/
Docker architecture
The Docker Architecture consists of the following components:

- **Docker daemon** (service)
- **Docker client** (**client manager** for the **service**)
- **Docker registry** for storing Docker images
- **Docker objects.** The aforesaid components are handled as objects by Docker. Additionally, the following components reside in Docker as objects:
  - **Image:** is a read-only **template with instructions for creating a Docker container.**
  - **Container:** is a **runnable instance of an image.**

[https://docs.docker.com/get-started/overview/](https://docs.docker.com/get-started/overview/)
Containers vs Virtual Machines

**Containers vs VMs differences** are in terms of *scale* and *portability*:

- **Containers** are *small in disk size*. They don’t package anything bigger than an app and all the libraries and files fundamental to run it. The *lightweight* nature of containers with the use of a a *shared operating system* makes the *move across multiple environments* effortless (*Disaster Recovery*). **Containers** provide a technique to *virtualise an OS* so that *different workloads can run* on a *solitary OS instance*.

- **VMs** are relatively *huge in disk size*. They *include their own OS*, permitting them to perform *numerous resource-intensive functions* at once. The *expanded resources offered to VMs permit them to abstract, split, duplicate, and emulate* entire servers, OSs, desktops, databases, and networks. In **VMs**, the *hardware is being virtualised* to run *multiple OS instances*.

Containers’ speed, agility, and portability make containers an important tool for technology deployments.

[https://www.weave.works/assets/images/bltb6200bc085503718/containers-vs-virtual-machines.jpg]
DTN meets Docker in GTS

Why run DTNs on Docker?

Because Docker supports the following:

- **Setup containers with specific services** (DTN Services) that can be accessible though the host IP address
- **Common sharing folder** among host and containers
- **Unrestricted CPU, Memory and bandwidth** (interface) access to containers (of course this is restricted to hardware limitations of the host)
- **Containers** are light in terms of resource consumption.
- We can have multiple services (each service will be a container) on a Docker Host
- **Docker** is written in the Go programming language and takes advantage of several features of the Linux kernel to deliver its functionality.

Docker is constructed with the vision to provide flexibility, dynamicity and easy integration of updates on containers. Therefore, a DevOps engineer can easily update the DTN software to another container, push the container to the live system and delete the old container. This functionality is important for DTN testing and integration, but the overmentioned functionality will reduce the risk when updating the DTN services and software.

GTS is selected because it can provide the necessary resources to setup and maintain a Dockerised environment. This is also shown with the results.
DTN meets Docker at GTS

The following DTN software and Services are installed as a separate container in Docker:

- GridFTP
- FDT
- XrootD

The selection of DTN software and service is done based on the following factors:

1. **Linux support** (Docker on Linux supports Linux Images)
2. **All are Open Source Projects**
3. **Good comments** from users on the web regarding fast data transfer.
4. They provide service and client in order to support huge file transfers.
5. **Isolated software without dependency** on Linux O/S and on Hardware.

The above characteristics make the installation of DTN over Docker on GTS very easy, with the use of scripts.
Docker Installation with DTN Software Containers at GTS

• The group has run several tests in a virtualised environment using containers with Docker. The focus was on the implementation of Dockerised environments to support specific file transfer services (i.e. XrootD, GridFTP and FDT).

• Dockerised DTN test are based on the client/server logic. The client can be a separate machine with only Linux Base Installation, the user can run a script that it will install the required packages. Also, the server that provides the services can be a Linux Base Installation. To setup the services on the server, it is needed to run one script that executes the setup of the Dockerised environment and afterwards another script that will setup the container/services.

• Once this code is ready for testing by NRENs, after the setup of the Dockerised environment and the installation of the containers by the participating NRENs who have servers, the participating NRENs will be able immediately to do the initial testing and evaluation of the DTN software with the use of a client script, which would include criteria like:
  • Performance of data transfers for different data workflow characteristics: large bulk transfers, lots-of-small-file transfers, and streaming data transfers.
  • Ease of use for end users.
  • Trust negotiation between the end hosts and security of the transfer.
Docker Installation with DTN Software Containers on GTS

- **Within containers in docker** you can manage the memory consumption and CPU consumption. However, the group containers have a preset CPU and Memory value, but if they need more, they can directly request it from the host Server.

- Containers configured with the services of gridFTP, XrootD and FDT with other ports than the default ports.
  - "ctop" is used for getting statistics from containers (bandwidth used, CPU used, memory used).

- **Client script is monitored** with the use of “bmon” and “iftop” tools for measuring bandwidth usage of the links.
  - During the process we used iperf for link testing
# Results of GTS Dockerised DTN Tests

<table>
<thead>
<tr>
<th>Hardware testing</th>
<th>Docker</th>
<th>Bare Metal Servers</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R430</td>
<td>R430</td>
</tr>
<tr>
<td>Node/Tools</td>
<td>HAMB-PRA</td>
<td>HAMB-PRA</td>
</tr>
<tr>
<td>iperf</td>
<td>9.2 Gb/s</td>
<td>9.41 Gb/s</td>
</tr>
<tr>
<td>gridFTP</td>
<td>8.53 Gb/s</td>
<td>8.58 Gb/s</td>
</tr>
<tr>
<td>FDT</td>
<td>8.87 Gb/s</td>
<td>9.39 Gb/s</td>
</tr>
<tr>
<td>XrootD</td>
<td>8.00 Gb/s</td>
<td>8.00 Gb/s</td>
</tr>
</tbody>
</table>

- **Minor difference** of results between server and client with larger and smaller distance/RTT
- **Minor difference** of results between Dockerise with non Dockerise environment
- **Minor difference** of results during late morning hours and the rest of the day (GTS)
- **Disk to Disk transfers** where slower than Disk to buffer and Buffer to disk

The Tests repeated with Centos 7 and the results where the similar to the results on Ubuntu 18.04.

Therefore, Dockerised DTN is a Bless!
Scripts and Manuals provided for Dockerised DTN test

• The *scripts* are **fully tested** on GTS and they are **fully automatic** for any O/S (Centos/Ubuntu)

• The following *Scripts* and *Manuals* are prepared:
  • Docker installation scripts and Manuals (Centos/Ubuntu)
  • Docker container installation scripts and Manuals (Centos/Ubuntu)
  • Client script for testing a DTN on another machine (Centos/Ubuntu)

All are shared at GÉANT project’s public DTN Wiki site

https://wiki.geant.org/display/DTN
Demo

Docke
ons, New P
Thank you

Any questions?

Or email us:

gn4-3-wp6-t2-dtn@lists.geant.org

www.geant.org

© GÉANT Association on behalf of the GN4 Phase 3 project (GN4-3).
The research leading to these results has received funding from the European Union’s Horizon 2020 research and innovation programme under Grant Agreement No. 856726 (GN4-3).