indium
Make Technology Work

How to load test Micro services-based Applications deployed in Dockers?

White Paper

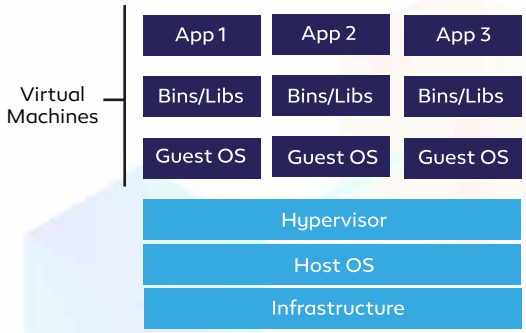
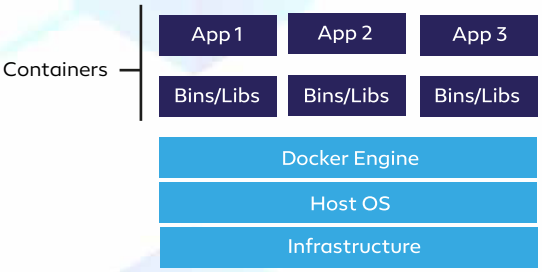
www.indiumsoftware.com

What is a Docker?

Docker is an open-source based platform which provides virtualization of services. It's a container engine which uses Linux kernel features to create containers. The containers are required to package the application which includes the bin and libraries. Developers package the piece of code in a container. So that this container can easily be deployed in a production environment.

Docker is containerization too and is often compared to virtual machines. However, Docker is a container-based technology that lets user to develop distributed applications. Find below the basic comparison table between Docker Vs. Virtual Machine

Benefits of Blockchain

Virtual Machine (VM)	Docker
Virtual machine acts exactly like a computer or desktop	Docker is a tool that uses containers to create, deploy and
Pictorial representation of VM  each virtual machine has its guest operating system above the host operating system, which makes virtual machines heavy.	Pictorial representation of Docker  Docker containers share the host operating system, which is why they are lightweight. Sharing the host OS between the containers make them very light and helps them to boot up in just a few seconds
Virtual machines have separate OS, so porting a virtual machine is di˜cult and it also takes a lot of time to port a virtual machine because of its size	Docker containers are easily portable as they do not have separate operating systems. A container can be ported to a di°er ent OS, and it can start immediately.

What are Microservices?

Microservice is an architectural style which structures an application as a group of services. The services are loosely coupled and hence it's easy to maintain and they can be deployed independently. Microservices are widely accepted new age architecture used by large enterprises for its agility.

Load test microservices-based application hosted in dockers

Identify the business-critical scenarios which need to be monitored in terms of performance (E.g. user creation, add to cart, check out, etc.). Each of the microservice could be written in di°er ent languages like java, python etc. The entire application comprises of a group of microservices which use their own APIs for communicating with each other.

It is recommended doing low-level load testing of each service during the development phase to iron out the performance issues of microservices at the early stages. While carrying out microservices load testing, it's important to have monitors in place to observe the behavior of the components associated with the microservice.

A docker-based application can also be load tested using open source performance testing tool like JMeter.

Key performance metrics to monitor microservices-based application deployed in Docker

Each microservice represents a separate server process and each microservice consumes resources.
Microservice consumes fewer resources than monolithic applications, however, the number of microservices in production will grow rapidly as the architecture scales.

Primary performance issue of using multiple microservices is that you must provide additional resources to run the application. Logging is part of the solution for performance problems and we can see how the microservice is behaving by carrying out log aggregation exercise. Tracing microservice requests play a pivotal role to understand what calls are being made and where the actual overhead is relying upon.

There are reasons for the slow response from a microservice, and they need to be identified. Application Performance Management tools can also be used for deep dive diagnosis to help identify specific services causing bottlenecks within the application pertaining to the microservices.

Few tools to monitor the performance of Microservices

- Raygun APM
- Zipkin (open source)
- Apache Kafka
- Grafana
- Prometheus

Important metrics to be monitored in Docker

Host CPU: Understanding the CPU utilization of hosts and containers helps one optimize the resource usage of Docker hosts. The container CPU usage can be throttled to avoid a single busy container slowing down other containers by using up all available CPU resources. Throttling the CPU time is a good way to ensure the minimum of processing power needed by essential services.

If the Docker has plenty of CPU capacity and if there, but you still suspect that it is compute-bound, you may want to check a container-specific metric "CPU throttling". If you do not specify any scheduling priority, then available CPU time will be split evenly between running containers. If some containers don't need all of their allotted CPU time, then it will be made proportionally available to other containers.

Name	Description
user CPU	Per cent of the time that CPU is under the direct control of processes
system CPU	Per cent of the time that CPU is executing system calls on behalf of processes
throttling (count)	Number of CPU throttling enforcements for a container
throttling (time)	Total time that a container's CPU usage was throttled

Host Memory: The total memory used in each Docker host is important to know for the current operations and for capacity planning. Dynamic cluster managers use the total memory available on the host and the requested memory for containers to decide on which host a new container should ideally be launched. Deployments might fail if a cluster manager is unable to find a host with su˜cien t resources for the container. Hence, it is important to know the host memory usage and the memory limits of containers.

Name	Description
Memory	Memory usage of a container
RSS	Non-cache memory for a process (stacks, heaps, etc.)
Cache Memory	Data from disk cached in memory
Swap	Amount of swap space in use

Host Disk Space: Docker images and containers consume additional disk space. For example, an application image might include a Linux operating system and might have a size of 400 - 500 MB depending on the size of the base image and installed tools in the container. Persistent Docker volumes consume disk space on the host as well. In our experience watching the disk space and using clean-up tools are essential for continuous operations of Docker hosts.

Name	Description
I/O serviced	Count of I/O operations performed, regardless of size
I/O service bytes	Bytes read or written by the cgroup

INDIA

Chennai | Bengaluru | Mumbai
Toll-free: 1800 123 1191

USA

Cupertino | Princeton
Toll-free: 1 888 207 5969

UK

London

SINGAPORE

+65 9630 7959

Sales Inquiries
sales@indiumsoftware.com

General Inquiries
info@indiumsoftware.com