Filière Informatique de gestion
Studiengang Wirtschaftsinformatik

Hes·so VALAIS WALLIS

Container-based software distribution for high performance computing using Singularity

Student : Jannis Kalbermatten
Professor : Prof. Dr. Henning Müller

Summary

1. Erforschen der Container-Software Singularity
2. Analyse der Implementationsvarianten für die Erstellung eines Singularity-Containers
3. Implementierung der bestehenden Datenanalyse in Singularity

Einleitung

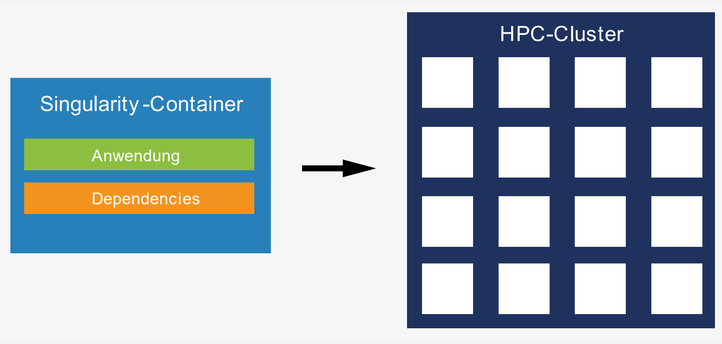
- Analyse von **medizinischen Bildern** benötigt hohe Rechenleistung und Speicherkapazität.
- Für die Analyse ist man auf **externe High-Performance Computing (HPC) Infrastrukturen** angewiesen.
- Die bildbasierte Datenanalyse besteht aus vielen **Komponenten und Programmbibliotheken**, welche zuvor installiert werden müssen. Dies ist jedoch **auf externen Systemen nicht möglich**.
- **Container-Technologien** lösen dieses Dilemma. Einer der wohl bekanntesten Container-Technologien heisst **Docker**. Docker **eignet sich** jedoch **nicht** für HPC, da im Hintergrund ein Docker **Daemon** läuft, welcher **Root-Zugriff** auf dem System benötigt.
- Die Container-Software **Singularity** eignet sich für HPC, da sie unter anderem zur Ausführung des Containers **keinen Root-Zugriff** auf dem System benötigt.

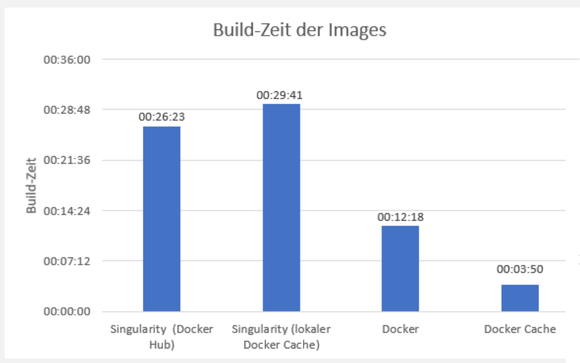
Vorgehen

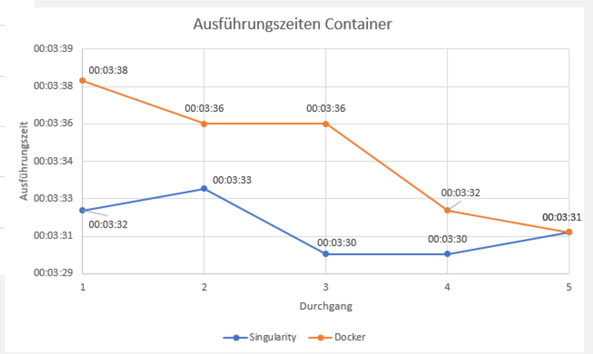
- **Verständnis** für **Container-Technologien** entwickeln. Insbesondere **Singularity** und **Docker**.
- **Stand der Technik** von Singularity erforschen.
- **Singularity-Container-Tools** testen.
- **Rootless Docker** ergründen und testen.
- Analyse der Anwendung zur Datenanalyse und der möglichen **Implementationsvarianten** in Singularity.
- **Implementierung** des Singularity-Containers mithilfe eines **bestehenden Docker-Images** in drei unterschiedlichen Varianten.
- **Performance Tests** des Singularity-Containers und ein Vergleich mit Docker.

Resultate

- Singularity weist **Gemeinsamkeiten und Unterschiede** zu anderen Container-Technologien auf.
- **Interoperabilität** zwischen Singularity und Docker konnte gewährleistet werden.
- Singularity-Container konnte mithilfe eines **bestehenden Docker-Images** implementiert werden.
- Die **Ausführungszeiten** des Singularity-Containers waren konstanter als bei Docker und im Durchschnitt schneller.
- Die **Erstellung** des Singularity-Images beanspruchte mehr Zeit als bei Docker, da das bestehende Docker-Image in Singularity zuerst umgewandelt werden musste.







Schlussfolgerung

- Die Anwendung konnte in Singularity mithilfe des bestehenden Docker-Images implementiert werden. Die Konvertierung des Docker-Images in das Singularity Image Format (SIF) funktionierte mühelos.
- Die Performance Tests zeigen nur einen minimalen bis gar keinen Unterschied zwischen Singularity und Docker.
- Singularity eignet sich bestens für High-Performance Computing.

Contact and more informations:
http://www.hevs.ch/ig