31.8.–3.9.2015
in Nürnberg

Herbstcampus

Wissenstransfer
par excellence

Docker Dir einen

Virtualisierung mit Docker

Simon Sudler

Redheads Ltd.

SIEMENS

Docker Dir einen!

Virtualisierung mit Docker

SIEMENS

Inhalt

- Warum Docker?
- Docker vs. Virtuelle Maschine
- Architektur
- Zombie-Reaping-Problem
- Schnellstart
- Eigenes Basis-Image erstellen
- Netzwerkanbindung
- Beispiel mit Gerrit, Jenkins und Build-Nodes

SIEMENS

Warum Docker?

- Was ist das Problem?
  - Komplexe Anwendungen → viele Abhängigkeiten
  - Abhängigkeiten von unterschiedlicher Versionen gleicher Bibliotheken
  - Bibliotheken nicht identisch in unterschiedlichen Distributionen
  - Anwendungen benötigen spezielle Konfiguration verwendeter Systemdienste
  - Inkompatible Konfiguration der Systemdienste für verschiedene Anwendungen
- Lösung: Eigene virtuelle Umgebung für jede einzelne Anwendung

SIEMENS

Warum Docker?

- Anforderungen an die virtuelle Umgebung:
  - Reproduzierbar: einfach, schnell, vollautomatisch
  - Sicher: isoliert vom Host-System
  - Erweiterbar: neue Systemdienste, neue Programme
  - Aktualisierbar: einfacher Update-Mechanismus
  - Austauschbar: einfacher Wechsel zwischen Versionen
  - Übertragbar: Lauffähig auf unterschiedlichen Host-Systemen
  - Effizient: minimaler Ressourcenverbrauch, kleine Images
  - ...

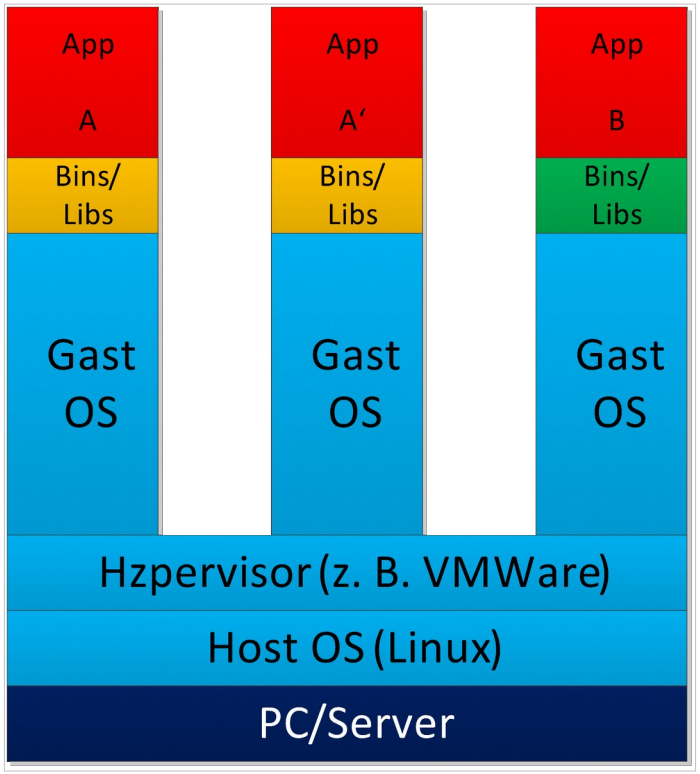
SIEMENS

Docker vs. Virtuelle Maschine

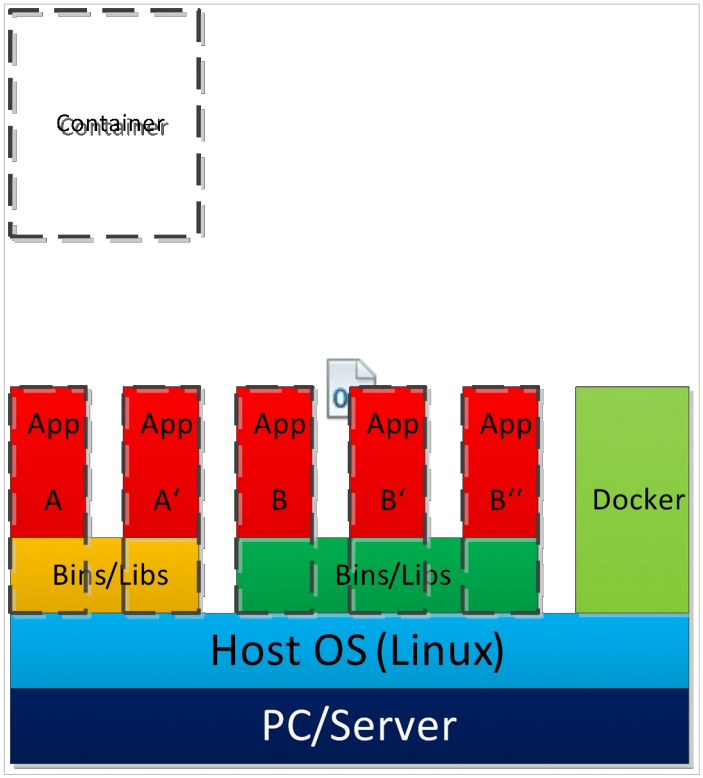
- Virtuelle Maschine
  - Virtualisiert ganzes System
  - Benötigt eigenes Betriebssystem
  - VM-Images können nicht zwischen virtuellen Maschinen geteilt werden
- Docker
  - Virtualisiert nur Applikationen
  - Nutzt Host-Betriebssystem
  - Binärdateien und Bibliotheken können zwischen Container geteilt werden

Docker vs. Virtuelle Maschine

- Virtuelle Maschine

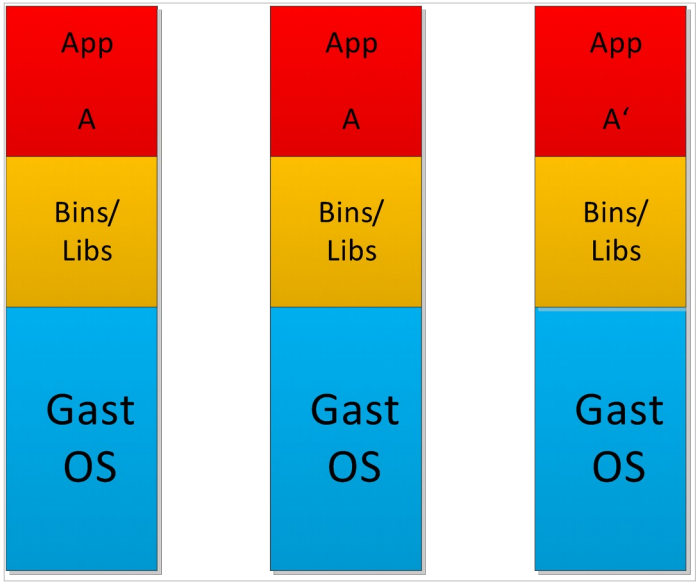


- Docker

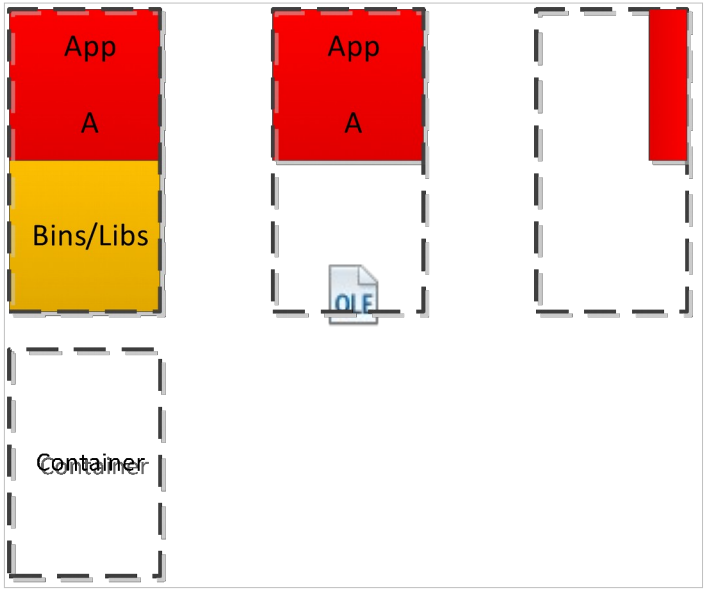


Docker vs. Virtuelle Maschine

- Virtuelle Maschinen
  - Jede Änderung, Kopie oder neue Version erfordert eigenes VM-Image



- Docker
  - Kopien benötigen kein OS und keine Bins/Libs
  - Änderungen werden differenziell gespeichert



SIEMENS

Architektur

- Docker-Daemon (Server)
  - Läuft auf Host-System
  - Verwaltet Container
  - Anwender kommuniziert nicht direkt mit dem Server
- Docker-Client
  - Kommandozeilenprogramm „docker“
  - Kommuniziert mit Docker-Deamon
- Kommunikation zwischen Server und Client über Sockets und RESTful API

SIEMENS

Architektur

- Docker-Images (build component)
  - Schreibgeschütztes Dateisystem
  - Enthält nur Anwendungen, kein Betriebssystem
  - Notwendig für das Erstellen von Docker-Containern
  - Einfach zu erstellen und zu verwalten
  - Einfach über eine Docker-Registry verteilbar

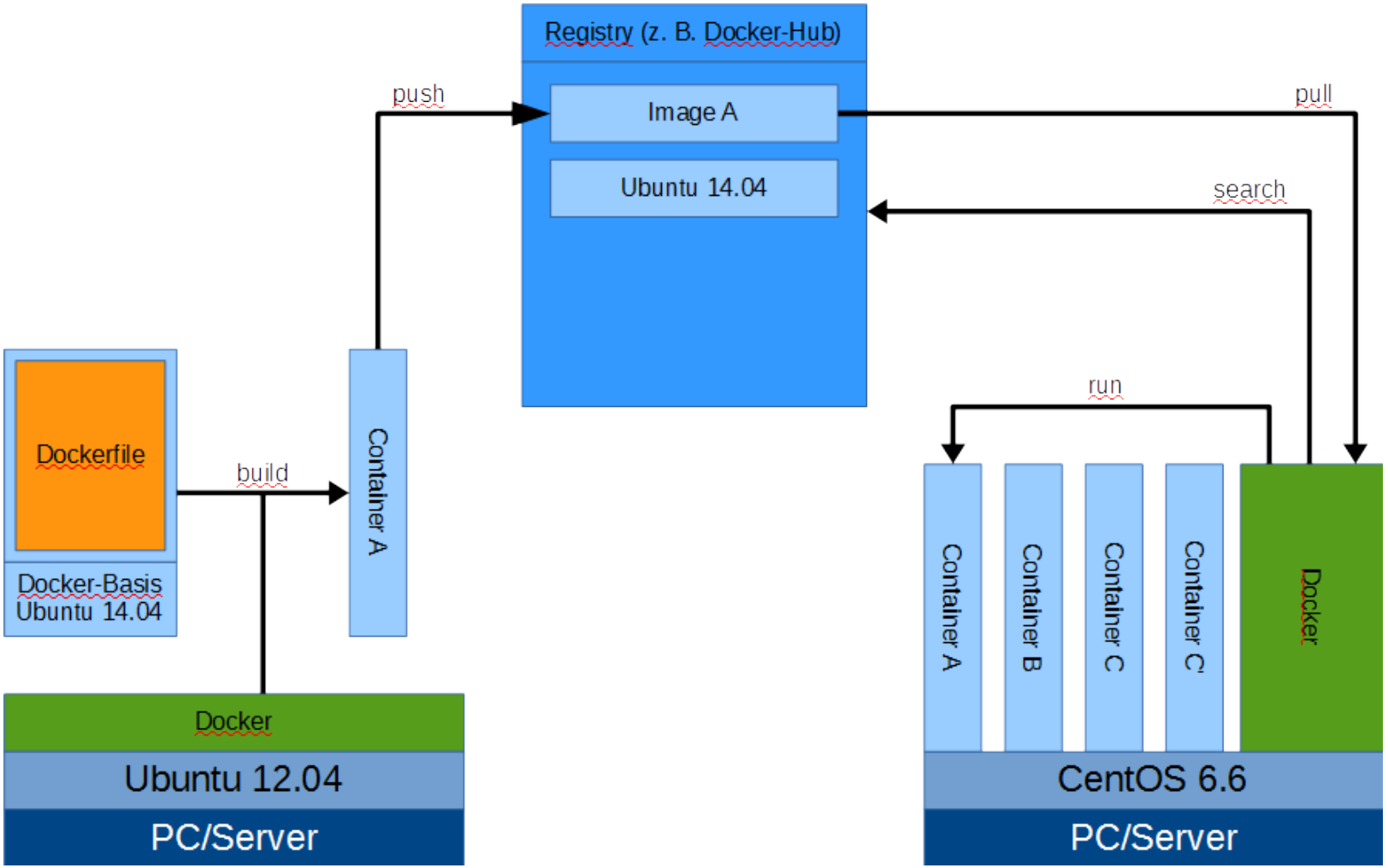
- Docker-Registry (distribution component)
  - Speichert Docker-Images
  - Öffentlich (z. B. Docker-Hub) oder nicht öffentlich

SIEMENS

Architektur

- Docker-Container (run component)
  - Vergleichbar mit einem Inhaltsverzeichnis
  - Enthält alle Komponenten, um eine Anwendung auszuführen
  - Wird auf Basis eines Docker-Images erstellt
  - Container können ausgeführt, angehalten, gestoppt und gelöscht werden
  - Isolierte und sichere Anwendungsumgebung (cgroups)

Architektur

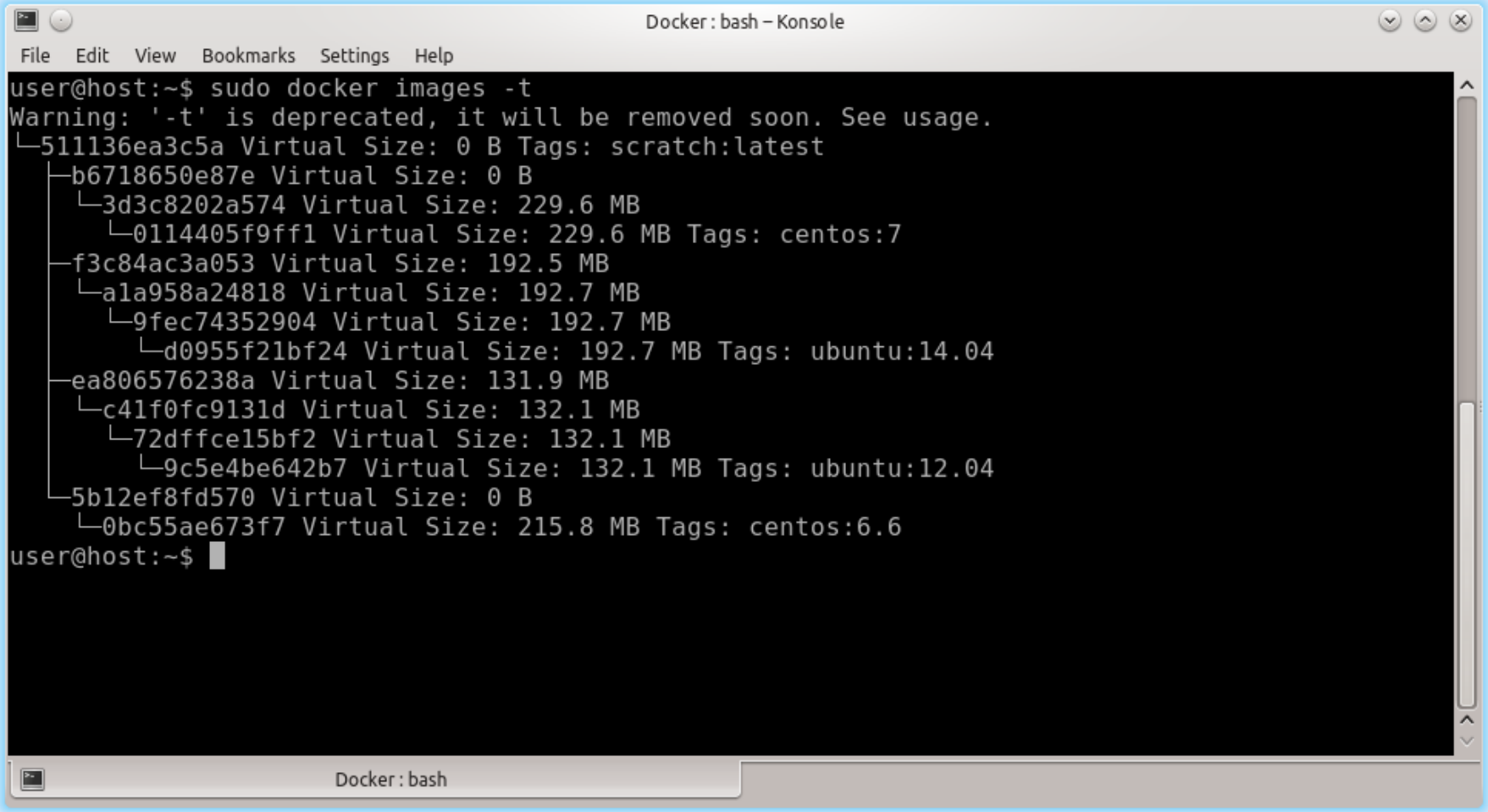


SIEMENS

Architektur – Docker-Images

- Images bestehen aus Layern
- Layer werden mittels UnionFS zu einem Dateisystem zusammen gefügt
- Änderungen an einem Image erzeugen einen neuen differenziellen Layer
- Update eines Images erfordert nur neuen Layer
- Layer starten von einem leeren Image (leeres Basis-Image (Ubuntu, CentOS, RHEL, …)
- Basis-Image stammt meist von der Registry „Docker-Hub“

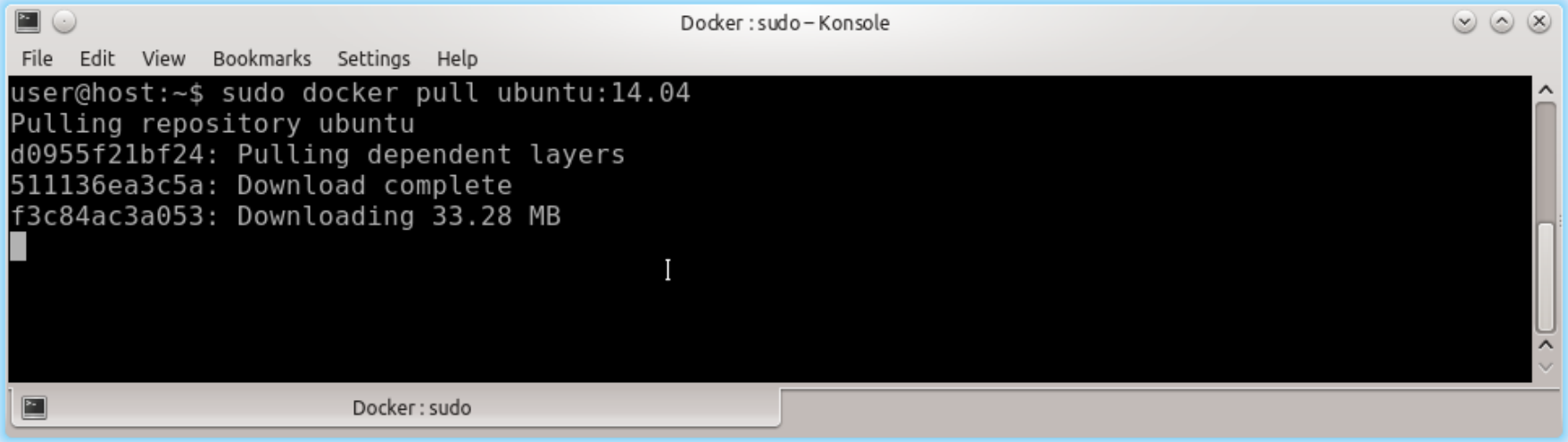
Architektur – Docker-Images



SIEMENS

Architektur – Docker-Images

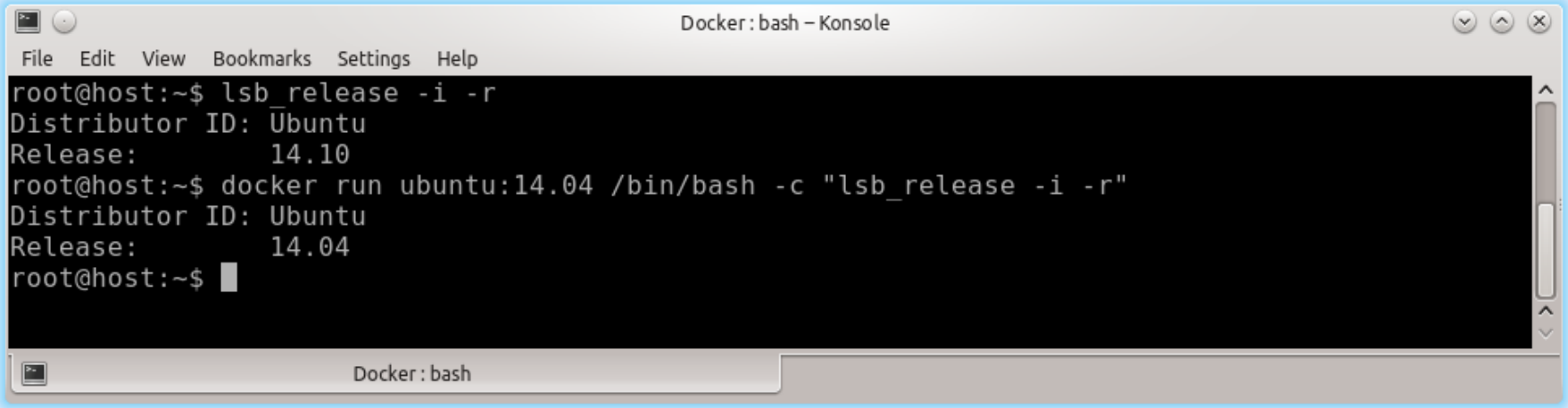
- Registry: Speichert Images
- Images können via Docker push/pull ausgetauscht werden
- Stellt Suchfunktionen für Images bereit



SIEMENS

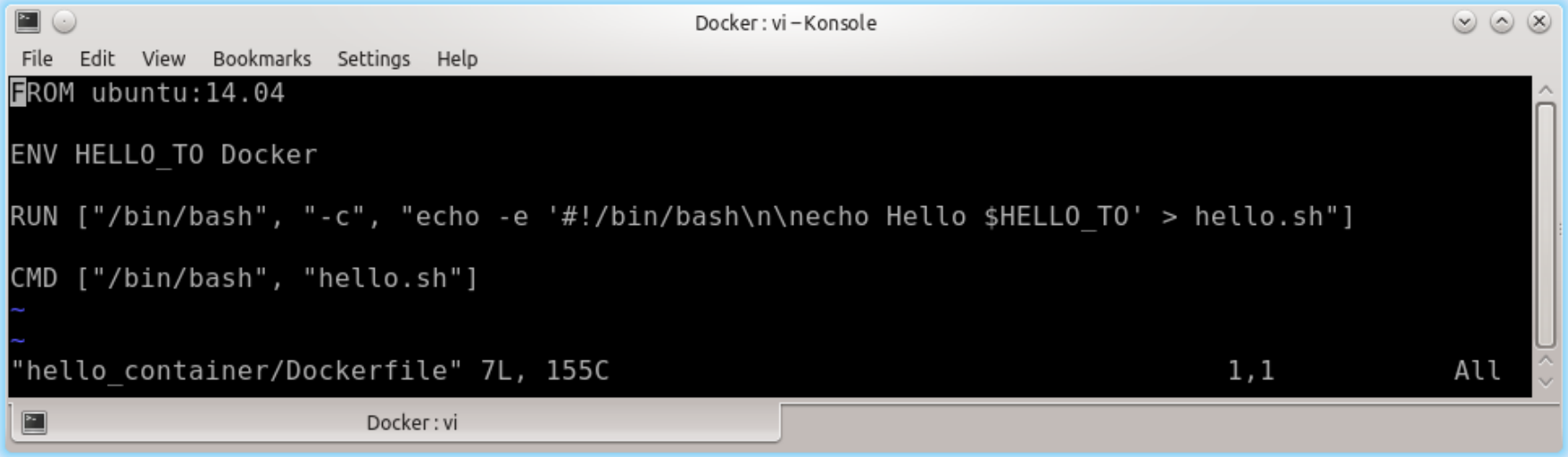
Architektur – Docker-Images

- Container werden von einem Docker Image abgeleitet
- Container erzeugen einen beschreibbaren Layer auf den schreibgeschützten Basis-Images
- Startet das im Container angegebene oder beim Starten übergebene Programm



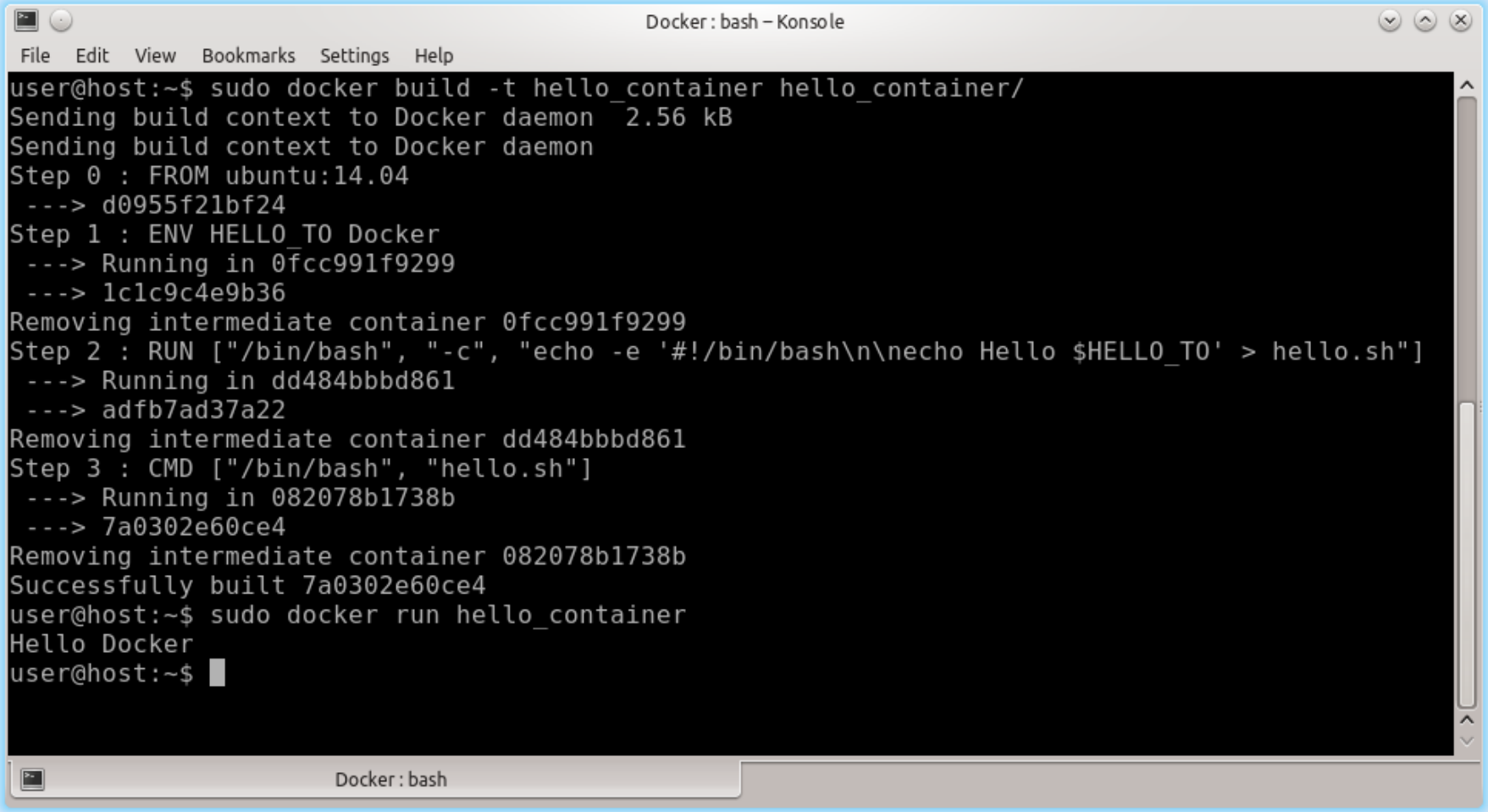
Architektur – Docker-Images

- Dockerfile: Instruktionsfolge zum Erstellen eines Images
  - Run: startet ein Programm innerhalb des Containers
  - Add: fügt dem Image Dateien oder Verzeichnisse hinzu
  - Env: erzeugt Umgebungsvariablen innerhalb des Images
  - Cmd: Anwendung, die beim Starten des Containers ausgeführt werden soll



SIEMENS

Architektur – Docker-Images



SIEMENS

Zombie-Reaping-Problem

- PID 1 Zombie-Reaping-Problem
  - Unix-Prozesse sind in einer Baumstruktur organisiert
  - Jeder Prozess hat einen Elternprozess, außer init(1)
  - Terminiert ein Prozess, wird er zum Zombie, bis der Elternprozess den Grund für das Terminieren erfragt (syscall waitpid(), AKA reaping)
  - Prozesse, die keine Elternprozess mehr haben, werden von init(1) übernommen → entfernt Zombies

SIEMENS

Zombie-Reaping-Problem

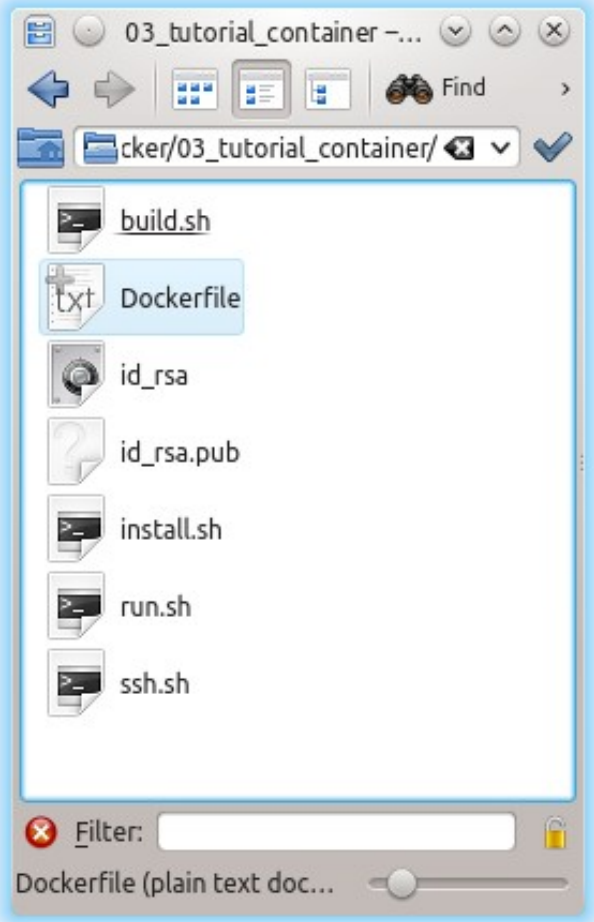
- Innerhalb eines Containers gibt es eine PID 1, aber keinen Init-Prozess!
- Docker-Anwendung wird mit PID 1 gestartet
- Bash „tötet“ adoptierte Kindprozesse (wenn sie zu Zombies werden):
  - DayZ: CMD ["/path-to-your-app"]
  - Besser: CMD ["/bin/bash", "-c", "set -e && /path-to-your-app"]

SIEMENS

Schnellstart

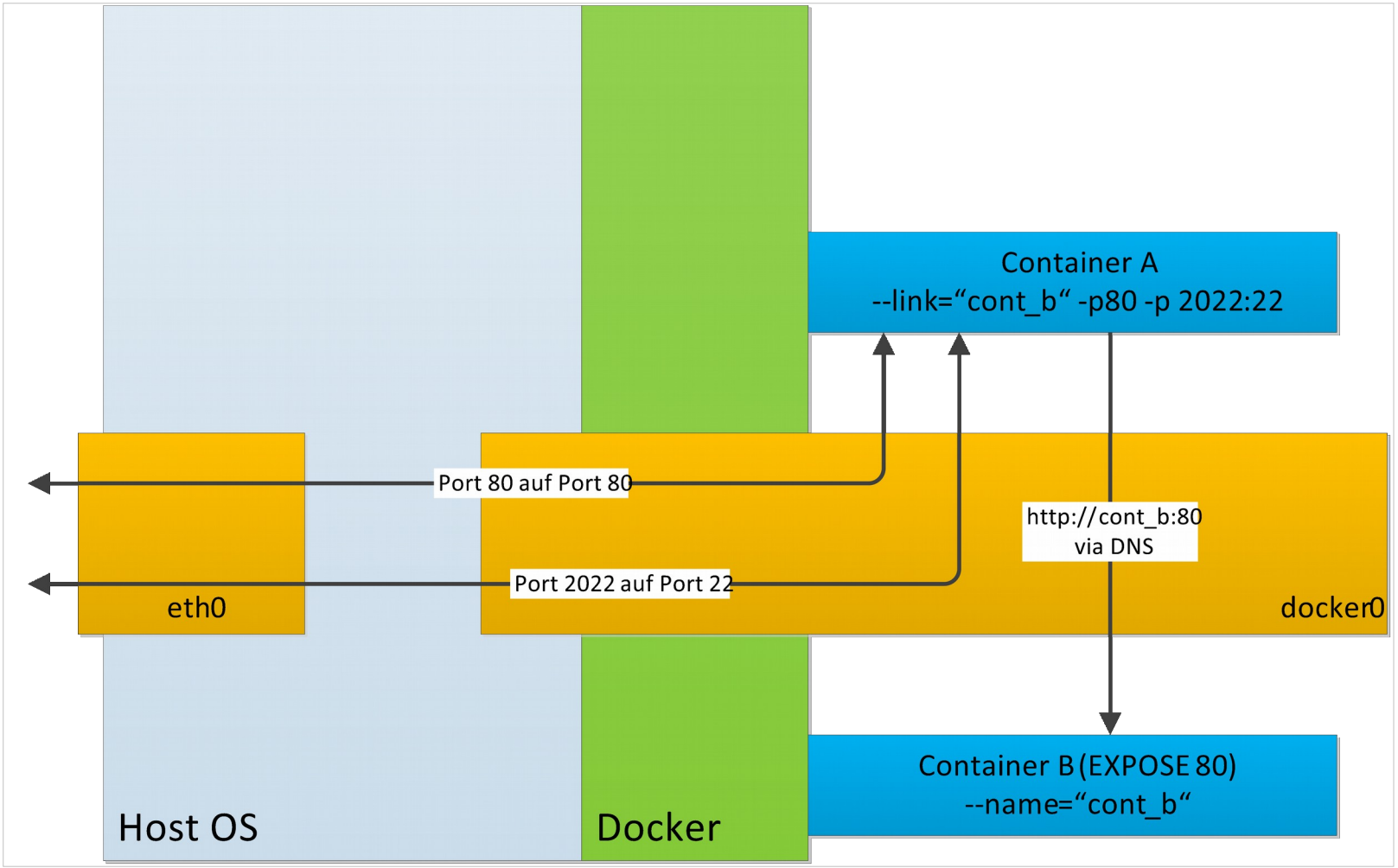
- Docker-Basis-Befehle des CLI (nicht vollständig)
  - pull: holt Image aus der Registry
  - build: erstellt ein neues Image
  - run: startet einen neuen Container
  - ps: zeigt die laufenden Container
  - stop/start: Container stoppen oder starten
  - attach: zu einem laufenden Container verbinden
  - rm: Container löschen
  - rmi: Image löschen
  - images: Images anzeigen

Eigenes Basis-Image erstellen

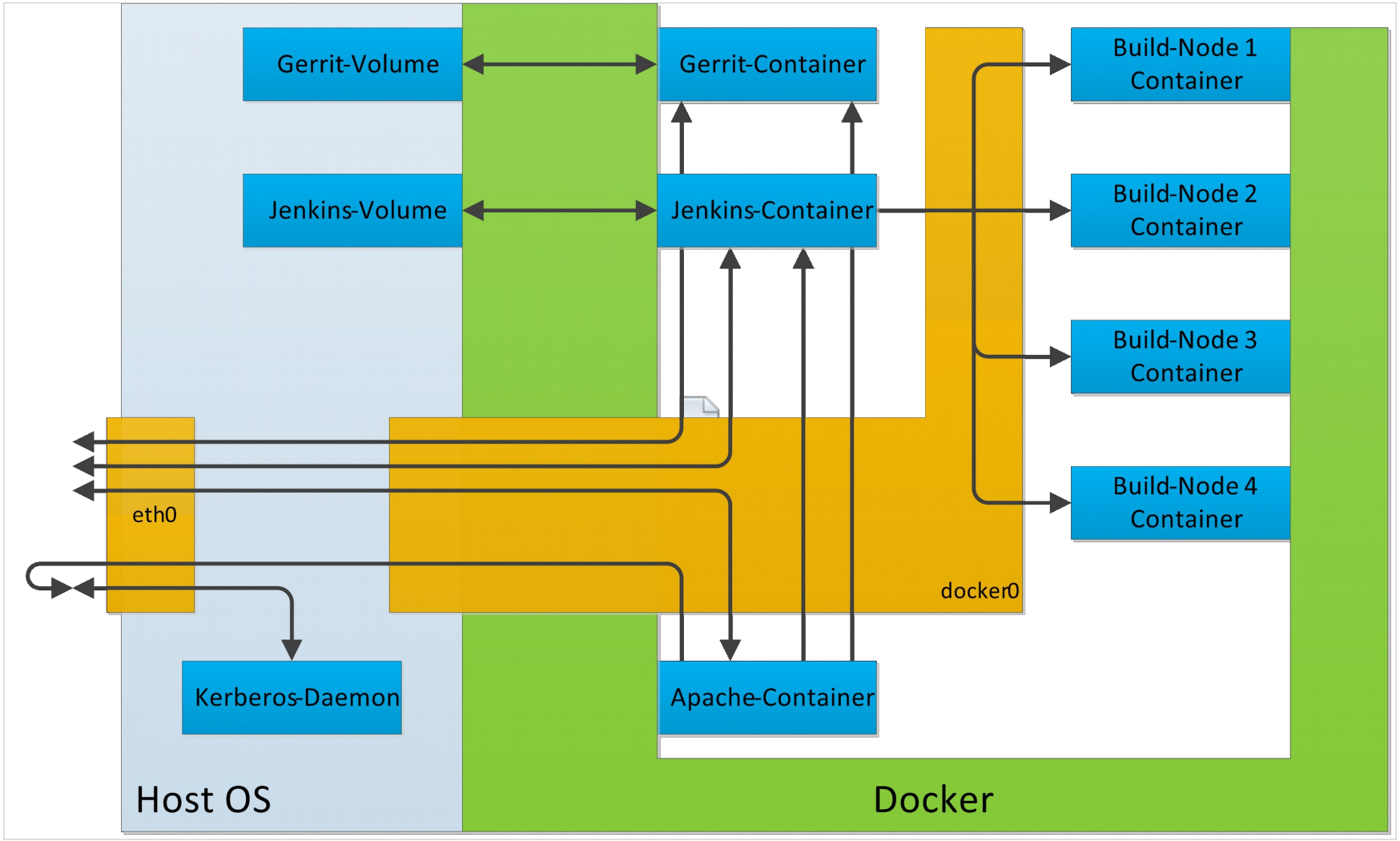


- Minimales Dockerfile
  - Kopiert install.sh und den öffenltichen SSH-Key
  - Install-Skript modifiziert Paketquelle, installiert und konfiguriert SSH-Server und Supervisord
- Build.sh, startet Image build
- Run.sh startet Container
- Ssh.sh ssh in den Container

Netzwerkanbindung



Beispiel mit Gerrit, Jenkins und Build-Nodes



Vielen Dank für Ihre Aufmerksamkeit!

Simon Sudler
PD TI ATS TM4 1

Gleiwitzer Str. 555
90475 Nürnberg

Telefon: +49 (911) 895-4135

E-Mail:
simon.sudler@siemens.com

siemens.com/answers