

Virtualisierung und Container im Serverbetrieb

1. Grundlagen der Virtualisierung

1.1 Was ist Virtualisierung?

Virtualisierung ermöglicht es, mehrere virtuelle Maschinen (VMs) auf einem physischen Server (Host) zu betreiben. Jede VM verhält sich wie ein eigenständiger Computer mit eigenem Betriebssystem.

Vorteile:

- Bessere Hardwareauslastung
- Kosteneinsparung (weniger physische Server)
- Schnellere Bereitstellung neuer Systeme
- Einfachere Wartung und Backups
- Isolation von Systemen

Nachteile:

- Overhead durch Virtualisierungsschicht
- Single Point of Failure (Host-Ausfall)
- Höhere Komplexität

1.2 Virtualisierungsarten

Typ-1 Hypervisor (Bare Metal)

Läuft direkt auf der Hardware ohne darunterliegendes Betriebssystem.

- **Beispiele:** VMware ESXi, Microsoft Hyper-V, Citrix XenServer, Proxmox VE
- **Vorteile:** Höhere Performance, geringerer Overhead
- **Einsatz:** Produktivumgebungen, Rechenzentren

Typ-2 Hypervisor (Hosted)

Läuft als Anwendung auf einem Host-Betriebssystem.

- **Beispiele:** VirtualBox, VMware Workstation, Parallels Desktop
- **Vorteile:** Einfache Installation, gut für Entwicklung/Tests
- **Nachteile:** Geringere Performance durch zusätzliche OS-Schicht

2. VMware vSphere & ESXi

2.1 ESXi (Bare Metal Hypervisor)

VMware ESXi ist der Typ-1 Hypervisor von VMware, der direkt auf der Hardware installiert wird.

Wichtige Begriffe:

- **ESXi-Host:** Der physische Server mit installiertem ESXi
- **Datastore:** Speicherort für VM-Dateien (VMFS, NFS, iSCSI)
- **vSwitch:** Virtueller Switch für Netzwerkverbindungen

Wichtige Dateien einer VM:

- `.vmx` - Konfigurationsdatei
- `.vmdk` - Virtuelle Festplatte
- `.nvram` - BIOS-Einstellungen
- `.vmsd` - Snapshot-Informationen

2.2 vSphere (Management-Plattform)

VMware vSphere ist die komplette Virtualisierungsplattform bestehend aus:

- **ESXi** (Hypervisor)
- **vCenter Server** (zentrale Verwaltung)

vCenter-Funktionen:

- **vMotion:** Live-Migration von VMs zwischen Hosts
- **Storage vMotion:** Migration von VM-Speicher im laufenden Betrieb
- **High Availability (HA):** Automatischer Neustart von VMs bei Host-Ausfall
- **DRS (Distributed Resource Scheduler):** Automatische Lastverteilung
- **Ressourcenpools:** Logische Gruppierung und Zuweisung von Ressourcen

2.3 Praxisbeispiel: VM erstellen

1. Im vSphere Client einloggen
2. Host auswählen → Rechtsklick → "New Virtual Machine"
3. Wizard durchlaufen:
 - Name und Speicherort
 - Compute-Ressource (Host/Cluster)
 - Datastore auswählen
 - Kompatibilität wählen
 - Guest OS auswählen
 - Hardware anpassen (CPU, RAM, Disk, Netzwerk)
4. VM starten und OS installieren

3. Microsoft Hyper-V

3.1 Überblick

Hyper-V ist Microsofts Typ-1 Hypervisor, verfügbar in:

- Windows Server (2016, 2019, 2022)
- Windows 10/11 Pro/Enterprise (Desktop-Version)

3.2 Wichtige Konzepte

Hyper-V Manager:

- GUI-Tool zur Verwaltung lokaler und Remote-Hosts

Virtual Switch-Typen:

- **External:** Verbindung zum physischen Netzwerk
- **Internal:** Kommunikation zwischen VMs und Host
- **Private:** Nur Kommunikation zwischen VMs

Generation 1 vs. Generation 2 VMs:

- **Gen 1:** Legacy BIOS, Kompatibilität mit älteren OS
- **Gen 2:** UEFI, Secure Boot, bessere Performance (nur 64-bit)

3.3 Wichtige Features

- **Live Migration:** VMs zwischen Hosts verschieben
- **Checkpoints:** Snapshots des VM-Zustands
- **Replica:** Disaster Recovery durch VM-Replikation
- **Dynamic Memory:** Automatische RAM-Anpassung
- **Resource Metering:** Ressourcenverbrauch nachverfolgen

3.4 PowerShell-Befehle

```
powershell
```

```
# Hyper-V Feature installieren
```

```
Install-WindowsFeature -Name Hyper-V -IncludeManagementTools
```

```
# VM erstellen
```

```
New-VM -Name "TestVM" -MemoryStartupBytes 2GB -Generation 2
```

```
# VM starten/stoppen
```

```
Start-VM -Name "TestVM"
```

```
Stop-VM -Name "TestVM"
```

```
# Checkpoint erstellen
```

```
Checkpoint-VM -Name "TestVM" -SnapshotName "Vor_Update"
```

```
# VMs auflisten
```

```
Get-VM
```

4. VirtualBox

4.1 Überblick

Oracle VirtualBox ist ein kostenloser Typ-2 Hypervisor für:

- Windows, macOS, Linux, Solaris

Einsatzgebiete:

- Entwicklung und Tests
- Schulungsumgebungen
- Private Nutzung

4.2 Netzwerkmodi

- **NAT:** VM hat Internet, aber ist von außen nicht erreichbar
- **Bridged:** VM erhält eigene IP im physischen Netzwerk
- **Internal Network:** Nur VMs untereinander
- **Host-Only:** Nur VM und Host kommunizieren
- **NAT Network:** Mehrere VMs teilen sich NAT mit untereinander Kommunikation

4.3 Wichtige Features

- **Snapshots:** Systemzustände speichern und wiederherstellen
 - **Guest Additions:** Treiber für bessere Integration (gemeinsame Ordner, Clipboard)
 - **Port Forwarding:** Zugriff auf VM-Dienste vom Host
 - **Linked Clones:** Platzsparende VM-Kopien
-

5. Container-Technologie

5.1 Was sind Container?

Container sind leichtgewichtige, isolierte Umgebungen für Anwendungen. Im Gegensatz zu VMs teilen sich Container den Kernel des Host-Systems.

VM vs. Container:

Aspekt	Virtual Machine	Container
Startzeit	Minuten	Sekunden
Größe	GBs	MBs
Isolation	Komplett (eigenes OS)	Prozess-Level
Performance	Overhead durch Hypervisor	Nahezu native Performance
Einsatz	Verschiedene OS, Legacy-Apps	Microservices, Cloud-native Apps

5.2 Docker Grundlagen

Wichtige Begriffe:

- **Image:** Vorlage für Container (unveränderlich)
- **Container:** Laufende Instanz eines Images
- **Dockerfile:** Bauanleitung für ein Image
- **Registry:** Speicherort für Images (z.B. Docker Hub)
- **Volume:** Persistenter Speicher für Container-Daten

Grundlegende Docker-Befehle:

```
bash
```

```
# Image herunterladen
```

```
docker pull nginx
```

```
# Container starten
```

```
docker run -d -p 80:80 --name webserver nginx
```

```
# Laufende Container anzeigen
```

```
docker ps
```

```
# Alle Container anzeigen
```

```
docker ps -a
```

```
# Container stoppen
```

```
docker stop webserver
```

```
# Container löschen
```

```
docker rm webserver
```

```
# Image löschen
```

```
docker rmi nginx
```

```
# In Container einloggen
```

```
docker exec -it webserver /bin/bash
```

```
# Logs anzeigen
```

```
docker logs webserver
```

Beispiel Dockerfile:

```
dockerfile
```

```
FROM ubuntu:22.04
```

```
RUN apt-get update && apt-get install -y nginx
```

```
COPY index.html /var/www/html/
```

```
EXPOSE 80
```

```
CMD ["nginx", "-g", "daemon off;"]
```

5.3 Docker Compose

Werkzeug für Multi-Container-Anwendungen mit YAML-Konfiguration.

Beispiel docker-compose.yml:

yaml

version: '3.8'

services:

web:

image: nginx

ports:

- "80:80"

volumes:

- ./html:/usr/share/nginx/html

database:

image: mysql:8.0

environment:

MYSQL_ROOT_PASSWORD: geheim123

MYSQL_DATABASE: webapp

volumes:

- db_data:/var/lib/mysql

volumes:

db_data:

Befehle:

bash

Starten

`docker-compose up -d`

Stoppen

`docker-compose down`

Logs anzeigen

`docker-compose logs -f`

6. Portainer

6.1 Was ist Portainer?

Portainer ist eine grafische Weboberfläche zur Verwaltung von Docker-Umgebungen.

Features:

- Container-Management (starten, stoppen, löschen)
- Image-Verwaltung
- Volume- und Netzwerk-Verwaltung
- Stack-Deployment (docker-compose)
- Multi-Host-Verwaltung
- Rollenbasierte Zugriffskontrolle

6.2 Installation

```
bash

# Volume für Portainer-Daten erstellen
docker volume create portainer_data

# Portainer starten
docker run -d \
  -p 9000:9000 \
  --name portainer \
  --restart always \
  -v /var/run/docker.sock:/var/run/docker.sock \
  -v portainer_data:/data \
  portainer/portainer-ce

# Zugriff über: http://server-ip:9000
```

7. Vergleich und Entscheidungshilfe

7.1 Bare Metal Hypervisors

Hypervisor	Vorteile	Nachteile	Einsatz
VMware ESXi/ vSphere	Ausgereift, viele Features, großer Support	Lizenzkosten, komplex	Enterprise-Umgebungen
Hyper-V	Windows-Integration, kostenlos in Server-Lizenz	Hauptsächlich für Windows-VMs optimiert	Microsoft-Umgebungen
Proxmox VE	Open Source, kostenlos, Container+VMs	Kleinere Community	KMU, gemischte Umgebungen

7.2 Wann was verwenden?

VMs (ESXi/Hyper-V) wählen wenn:

- Unterschiedliche Betriebssysteme benötigt werden
- Komplette Isolation erforderlich ist
- Legacy-Anwendungen betrieben werden
- Langlebige, stabile Infrastruktur nötig ist

Container (Docker/Portainer) wählen wenn:

- Microservices-Architektur verwendet wird
- Schnelle Skalierung wichtig ist
- Cloud-native Anwendungen entwickelt werden
- Ressourceneffizienz im Vordergrund steht

VirtualBox wählen wenn:

- Entwicklung/Tests auf Desktop-Systemen
 - Schnelles Ausprobieren verschiedener OS
 - Keine Produktivumgebung
-

8. Prüfungsrelevante Themen

8.1 Typische Prüfungsfragen

1. Erkläre den Unterschied zwischen Typ-1 und Typ-2 Hypervisoren.

- Typ-1 läuft direkt auf Hardware (Bare Metal), Typ-2 auf Host-OS
- Typ-1 performanter für Produktivumgebungen
- Beispiele nennen können

2. Nenne Vorteile der Virtualisierung.

- Bessere Ressourcenauslastung
- Kostenreduktion
- Schnellere Bereitstellung
- Vereinfachte Backups/Disaster Recovery
- Testumgebungen

3. Was ist der Unterschied zwischen Container und VM?

- VMs haben eigenes OS, Container teilen Kernel
- Container sind leichtgewichtiger und schneller
- VMs bieten stärkere Isolation

4. Erkläre vMotion.

- Live-Migration von VMs zwischen ESXi-Hosts
- Keine Downtime
- Erfordert shared Storage und vCenter

5. Welche Docker-Komponenten gibt es?

- Image (Vorlage)
- Container (laufende Instanz)
- Registry (Speicher für Images)
- Volume (persistente Daten)

8.2 Praktische Aufgaben üben

1. VM in ESXi/Hyper-V erstellen und konfigurieren
2. Snapshot/Checkpoint erstellen und wiederherstellen
3. Virtuelle Netzwerke einrichten
4. Docker-Container starten und konfigurieren
5. Docker Compose Stack deployen
6. Portainer installieren und Container verwalten

9. Wichtige Ressourcen

Offizielle Dokumentationen:

- VMware vSphere: docs.vmware.com
- Microsoft Hyper-V: docs.microsoft.com/hyper-v
- Docker: docs.docker.com
- Portainer: docs.portainer.io

Kostenlose Übungsumgebungen:

- VMware Hands-on Labs (HOL)
- Microsoft Learn Sandbox
- Docker Playground (play-with-docker.com)
- Katacoda (jetzt O'Reilly Learning)

Zertifizierungen:

- VMware Certified Professional (VCP)
 - Microsoft Certified: Azure Administrator
 - Docker Certified Associate (DCA)
-

10. Checkliste für die Prüfungsvorbereitung

- ☐ Unterschied Typ-1 / Typ-2 Hypervisor erklären können
- ☐ Vor- und Nachteile der Virtualisierung kennen
- ☐ ESXi-VM erstellen und verwalten können
- ☐ vCenter-Funktionen (vMotion, HA, DRS) erklären
- ☐ Hyper-V Virtual Switch-Typen unterscheiden
- ☐ Generation 1 vs. 2 VMs kennen
- ☐ VirtualBox Netzwerkmodi verstehen
- ☐ VM vs. Container Unterschiede erklären
- ☐ Grundlegende Docker-Befehle beherrschen
- ☐ Dockerfile-Syntax verstehen
- ☐ Docker Compose verwenden können
- ☐ Portainer-Funktionen kennen
- ☐ Anwendungsfälle für VMs vs. Container begründen können