Hochverfügbare Virtualisierung mit Open Source

Gliederung

- DRBD
- Ganeti
- Libvirt

Virtualisierung und Hochverfügbarkeit

- Hochverfügbarkeit von besonderer Bedeutung
- Defekt an **einem** Server
  => Ausfall **vieler** VMs
  => Datenverlust in **vielen** Vms

- Lösung: Redundanz
- Wie vollständige Redundanz realisieren?

Server-Redundanz

- Mehrere Virtualisierungsserver
- Ein Speichersystem

- Realisierung z.b. mittels iSCSI

- Nachteil: Keine Datenredundanz

DRBD

- Distributed Replicated Block Device
- Verteiltes Blockdevice
- Änderungen werden übers Netzwerk repliziert

  => Jeder Block liegt auf zwei Servern

  => Fällt ein Server aus übernimmt der andere
- Als Blockdevice generisch einsetzbar

DRBD - Modes

- Mode A
  - Warten bis Daten **versendet** wurden

    => Datenverlust möglich
- Mode B
  - Warten bis Daten **empfangen** wurden

    => Datenverlust sehr unwahrscheinlich
- Mode C
  - Warten bis Daten **geschrieben** wurden

    => Datenverlust ausgeschlossen

Single-Primary-Mode

- Ein Node ist **primär**:
  Kann gelesen und beschrieben werden
- Anderer Node ist **sekundär**:
  Kann **nicht** gelesen oder beschrieben werden
- Im Fehlerfall: Rollentausch

Dual-Primary-Mode

- Beide Nodes sind **primär**:
  Beide können gelesen und beschrieben werden
- **Gefahr eines „Split-Brain“!**
- Werden beide gleichzeitig beschrieben haste die Kacke am dampfen!
- Split-Brain auflösen ohne Datenverlust fast unmöglich

Migration von VMs auf DRBD

- Live-Migration von VMs theoretisch einfach:
  - Daten liegen auf beiden Hosts
- Aber: Während der Migration Lesezugriff nötig

  => Migration auf Sekundär-Node nicht möglich
- => Dual-Primary-Mode notwendig
  - Split-Brain-Gefahr!
  - Software zur Durchführung nötig: Ganeti

Ganeti

- VM-Management-Software
- OpenSource
- Ermöglicht Redundanz der Hardware
- Backend: KVM oder Xen

Ganeti

- Administration:
  - CLI
  - HTTP (Web-Manager)
    - Aktuelle Version hat mit Bugs zu kämpfen
- Übernimmt die Verwaltung der DRBD-Backends
  - Split Brain wird vermieden

Ganeti-Migration

- Schnell Live-Migration
  - Persistente Daten werden nicht kopiert
- Hochverfügbar
  - Ausfall eines Servers:
    Starten der VMs auf zweitem Server
  - Kein Datenverlust (aus Sicht von Ganeti)

Libvirt

- Middleware zur Virtualisierung
- Backend: KVM, Xen, VMWare, …
- Storage:
  - iSCSI
  - Ceph (RBD)
  - Sheepdog
  - Gluster
  - ...

Ceph Block Devices

- Zugriff auf Ceph-Cluster
- Nutzung als Block Device
- Snapshots und cloning wird unterstützt

Ceph Block Devices

- Vorteil
  - Mehr als zwei Backing-Nodes
  - Gut erprobt und sehr stabil
- Nachteile
  - QEMU unterstützt (noch) kein locking
    => **Split Brain möglich!**
  - Einschränkte Unterstützung durch virt-manager

Sheepdog

- Verteiltes Speichersystem für QEMU
- OpenSource
- Geringer Administrationsaufwand
  - Nodes finden sich per Multicast
  - Nodes lassen sich sehr einfach hinzufügen
- Praktisch beliebig viele Nodes möglich

Sheepdog

- Vorteile:
  - Mehr als zwei Nodes möglich
  - Geringer Administrationsaufwand
- Nachteile:
  - QEMU unterstützt kein locking
    => **Split Brain möglich!**
  - Eingeschränkte Unterstützung durch virt-manager

GlusterFS

- Verteiltes Dateisystem
- OpenSource
- Zugriff auf Dateien über
  - NFS
  - CIFS
  - Gluster Native
- QEMU unterstützt Gluster Native

GlusterFS

- Vorteile:
  - Mehr als zwei Nodes möglich
- Nachteile:
  - QEMU unterstützt kein locking
    => **Split Brain möglich!**
  - Eingeschränkte Unterstützung durch virt-manager

Libvirt Lock Manager

- Natives Locking bei verteilten Dateisystemen nicht möglich
- Aber: Libvirt besitzt Lock Manager
- Einziger unterstütztes Verfahren: sanlock
  - Braucht selbst wieder Netzwerk-Speicher
  - Relativ Ressourcen intensiv

Fazit

- Ganeti + QEMU + DRBD
  - Ermöglicht Hochverfügbarkeit
  - Verhindert Split Brain
- Libvirt + QEMU
  - Unterstützt verschiedene verteilte Speichersysteme
  - Locking fehlt aber (bzw. ist unpraktikabel)
    => Risiko eines Split Brains

Hochverfügbare Virtualisierung mit Open Source

Gliederung

- DRBD
- Ganeti
- Libvirt

Virtualisierung und Hochverfügbarkeit

- Hochverfügbarkeit von besonderer Bedeutung
- Defekt an **einem** Server
  => Ausfall **vieler** VMs
  => Datenverlust in **vielen** Vms

- Lösung: Redundanz
- Wie vollständige Redundanz realisieren?

Server-Redundanz

- Mehrere Virtualisierungsserver
- Ein Speichersystem

- Realisierung z.b. mittels iSCSI

- Nachteil: Keine Datenredundanz

DRBD

- Distributed Replicated Block Device
- Verteiltes Blockdevice
- Änderungen werden übers Netzwerk repliziert
  - => Jeder Block liegt auf zwei Servern
  - => Fällt ein Server aus übernimmt der andere
- Als Blockdevice generisch einsetzbar

DRBD - Modes

- Mode A
  - Warten bis Daten **versendet** wurden
    => Datenverlust möglich
- Mode B
  - Warten bis Daten **empfangen** wurden
    => Datenverlust sehr unwahrscheinlich
- Mode C
  - Warten bis Daten **geschrieben** wurden
    => Datenverlust ausgeschlossen

Single-Primary-Mode

- Ein Node ist **primär**:
  Kann gelesen und beschrieben werden
- Anderer Node ist **sekundär**:
  Kann **nicht** gelesen oder beschrieben werden
- Im Fehlerfall: Rollentausch

Dual-Primary-Mode

- Beide Nodes sind **primär**:
  Beide können gelesen und beschrieben werden
- **Gefahr eines „Split-Brain“!**
- Werden beide gleichzeitig beschrieben haste die Kacke am dampfen!
- Split-Brain auflösen ohne Datenverlust fast unmöglich

Migration von VMs auf DRBD

- Live-Migration von VMs theoretisch einfach:
  - Daten liegen auf beiden Hosts
- Aber: Während der Migration Lesezugriff nötig

  => Migration auf Sekundär-Node nicht möglich
- => Dual-Primary-Mode notwendig
  - Split-Brain-Gefahr!
  - Software zur Durchführung nötig: Ganeti

Ganeti

- VM-Management-Software
- OpenSource
- Ermöglicht Redundanz der Hardware
- Backend: KVM oder Xen

Ganeti

- Administration:
  - CLI
  - HTTP (Web-Manager)
    - Aktuelle Version hat mit Bugs zu kämpfen
- Übernimmt die Verwaltung der DRBD-Backends
  - Split Brain wird vermieden

Ganeti-Migration

- Schnell Live-Migration
  - Persistente Daten werden nicht kopiert
- Hochverfügbar
  - Ausfall eines Servers:
    Starten der VMs auf zweitem Server
  - Kein Datenverlust (aus Sicht von Ganeti)

Libvirt

- Middleware zur Virtualisierung
- Backend: KVM, Xen, VMWare, …
- Storage:
  - iSCSI
  - Ceph (RBD)
  - Sheepdog
  - Gluster
  - ...

Ceph Block Devices

- Zugriff auf Ceph-Cluster
- Nutzung als Block Device
- Snapshots und cloning wird unterstützt

Ceph Block Devices

- Vorteil
  - Mehr als zwei Backing-Nodes
  - Gut erprobt und sehr stabil
- Nachteile
  - QEMU unterstützt (noch) kein locking
    => **Split Brain möglich!**
  - Einschränkte Unterstützung durch virt-manager

Sheepdog

- Verteiltes Speichersystem für QEMU
- OpenSource
- Geringer Administrationsaufwand
  - Nodes finden sich per Multicast
  - Nodes lassen sich sehr einfach hinzufügen
- Praktisch beliebig viele Nodes möglich

Sheepdog

- Vorteile:
  - Mehr als zwei Nodes möglich
  - Geringer Administrationsaufwand
- Nachteile:
  - QEMU unterstützt kein locking
    => **Split Brain möglich!**
  - Eingeschränkte Unterstützung durch virt-manager

GlusterFS

- Verteiltes Dateisystem
- OpenSource
- Zugriff auf Dateien über
  - NFS
  - CIFS
  - Gluster Native
- QEMU unterstützt Gluster Native

GlusterFS

- Vorteile:
  - Mehr als zwei Nodes möglich
- Nachteile:
  - QEMU unterstützt kein locking
    => **Split Brain möglich!**
  - Eingeschränkte Unterstützung durch virt-manager

Libvirt Lock Manager

- Natives Locking bei verteilten Dateisystemen nicht möglich
- Aber: Libvirt besitzt Lock Manager
- Einziger unterstütztes Verfahren: sanlock
  - Braucht selbst wieder Netzwerk-Speicher
  - Relativ Ressourcen intensiv

Fazit

- Ganeti + QEMU + DRBD
  - Ermöglicht Hochverfügbarkeit
  - Verhindert Split Brain
- Libvirt + QEMU
  - Unterstützt verschiedene verteilte Speichersysteme
  - Locking fehlt aber (bzw. ist unpraktikabel)
    => Risiko eines Split Brains