Hochverfügbarkeit für KMUs:
Shared-Nothing Clustering für virtuelle Hosts

Grazer Linuxtage 2008
19. April 2008 11:00 - 11:45, FH Johanneum

Rene Mayrhofer
Universität Wien
Gibraltar / eSYS Informationssysteme GmbH

Warum Virtualisierung?

Wirtschaftliche Vorteile durch Konsolidierung

- Energieaufwand
- Hardwareaufwand
- Einsparung an Hardware-Wartungskosten

Organisatorische Vorteile

- weniger Server ⇒ kleineres Rechenzentrum
- Überblick kann besser sein (abhängig von Verwaltungswerkzeugen)

Warum Virtualisierung?

Sicherheit

- einfache Verwaltung virtueller Netzwerke
- mehr virtuelle Server als physikalische möglich ⇒ „billige“ Zonen
- Überwachung virtueller Server „von außen“

Verfügbarkeit

- dynamische Migration virtueller Server
  ⇒ Minimierung der Ausfallszeiten
  ⇒ Lastverteilung
- 2 (oder mehr) physikalische Server zum Hot-Standby Fail-Over für viele virtuelle Server

Warum keine Virtualisierung?

Komplexität

- zusätzliche Softwareschicht oder Hardwareunterstützung
- viele virtuelle Server
  ⇒ hoher Aufwand zur Wartung
  ⇒ komplexe Beziehungen zwischen Servern
- Lern- und Wartungsaufwand für Virtualisierungsschicht

Ausfallssicherheit

- Ausfall eines physikalischen Servers ⇒ Ausfall vieler virtueller Server

Virtualisierungsvarianten

Hardware-Emulation

- flexibel: beliebige Systeme emulierbar (andere Prozessorarchitekturen)
- langsam
- keine Änderung der Gastsysteme nötig

Hardware-Virtualisierung

KVM

- I/O emuliert, Prozessor+Hauptspeicher „durchgereicht“
- *Virtual Machine Monitor* als Zwischenschicht, z.T. mit Hardwareunterstützung
- keine Änderung der Gastsysteme nötig

Virtualisierungsvarianten

Paravirtualisierung

- ähnlich Hardware-Virtualisierung
- I/O nicht emuliert sondern über eigene Schnittstellen implementiert
- Gast-Betriebssystem muss Schnittstellen des VMM verwenden

Betriebssystemvirtualisierung

- ein Kern, mehrere „Container“ / „Zonen“
- geringster Overhead, z.T. gemeinsame Verwendung von Bibliotheken

Solaris Containers

Andere Varianten erfordern Modifikation der Anwendungen!

Fokus in dieser Variante

Virtualisierung zur Verbesserung der

- Sicherheit
- Verfügbarkeit
- Verwaltbarkeit

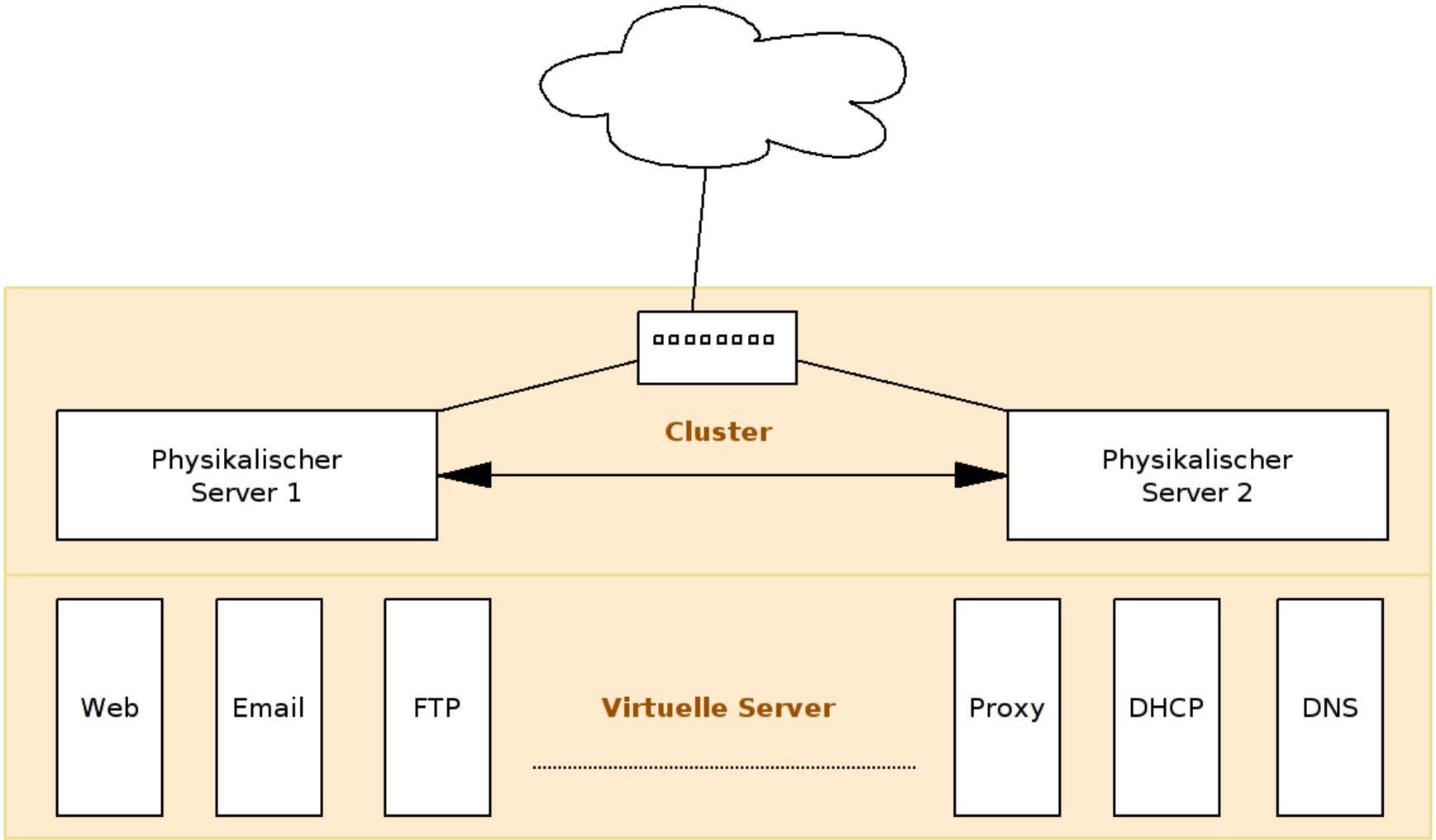
für kleinste bis mittelgroße Unternehmen

⇒ wartungsarm

⇒ automatisches Fail-Over

⇒ kostengünstig und energieeffizient

Shared-Nothing Cluster zur Virtualisierung



Shared-Nothing Cluster zur Virtualisierung

Verzicht auf SAN/NAS

- kostengünstig
- keine zusätzliche Hardware
  ⇒ potentiell wartungsarm
  ⇒ Energieersparnis

Kein „single point of failure“

- alle Komponenten können mehrfach ausgelegt werden

Standard-Hardware

- keine Abhängigkeit von einzelnen Herstellern/Lieferanten
- bei Ausfall leicht austauschbar

Virtualisierung mit Linux Hosts

Container mit einem Kernel pro Node

- VServer
- OpenVZ

Para- und Hardware-Virtualisierung

- Xen
- Virtualbox OSE
- KVM

Synchronisieren zwischen Hosts

DRBD v0.8

- bekannt, verbreitet
- Synchronisation auf der Ebene von Block devices
- unabhängig vom Filesystem

Distributed storage subsystem

Cluster Filesysteme

- GFS(2)
- OCFS2

Verteilte Dateisysteme

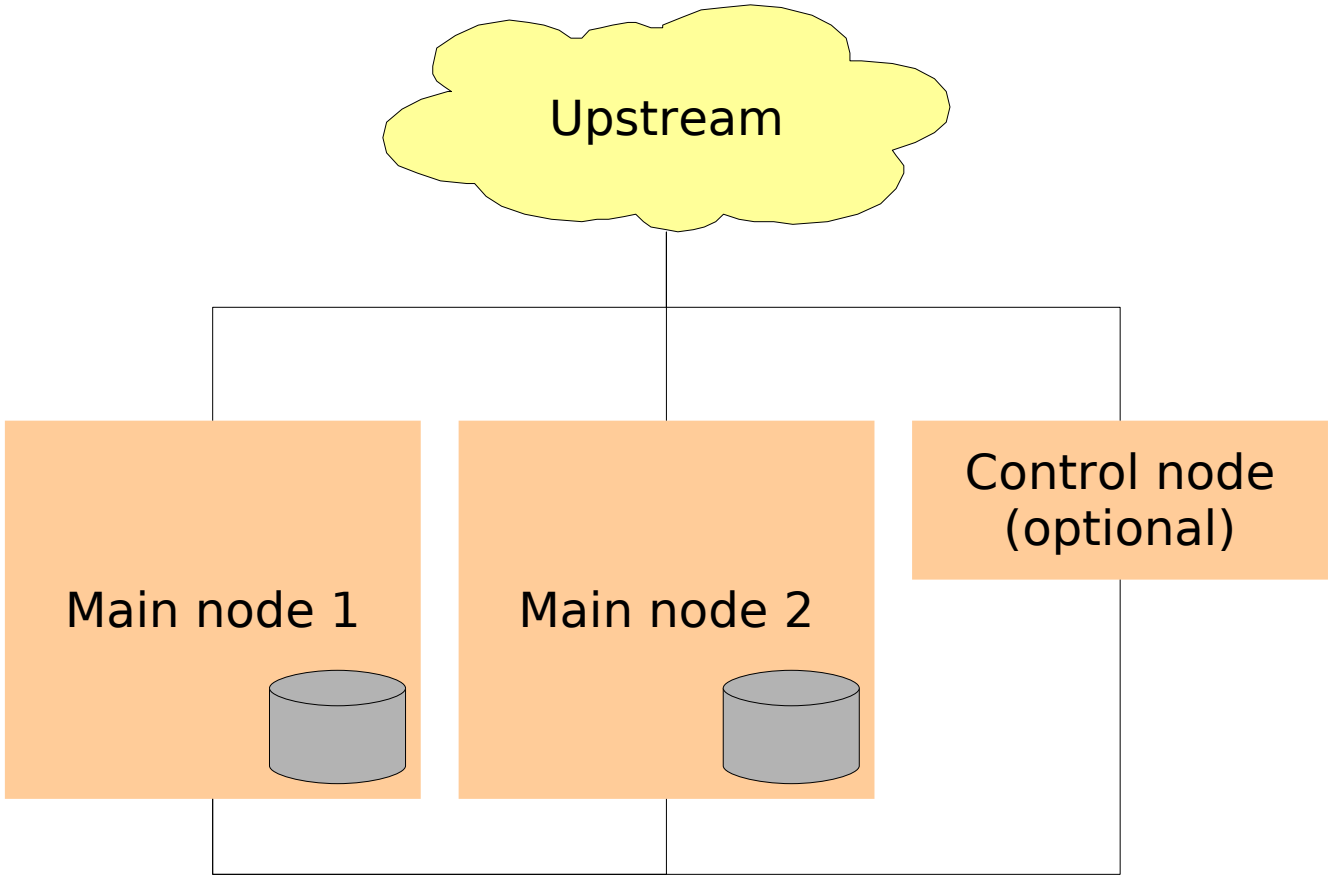
- Lustre (zentrale Metadaten), Ceph, GlusterFS, Kosmos FS (optimiert für reads)

Active/Active Koordination von Ressourcen

Heartbeat 2

- Verwaltung von Ressourcen, z.B. mit gegenseitigem Ausschluss im Cluster
- unterstützt >2 Nodes
- komplexe Abhängigkeiten möglich

Grundstruktur



Basissystem

- (verschiedene) HP Server, 3x250GB SATA Hotplug (je 2 im System)
- Multi-Core CPUs mit Intel VT, 4GB RAM
- Debian etch 4.0, Kernel *2.6.22-3-vserver-686* oder *2.6.18-6-xen-vserver-686*
- Software RAID1, darüber LVM in einer Partition (60GB)

DRBD Konfiguration

```
resource "xen-web1" {
  protocol C;
  handlers {
    outdate-peer "/usr/lib/heartbeat/drbd-peer-
                outdater";
  }
  startup {
    wfc-timeout       120;  ## 2 minutes.
    degr-wfc-timeout  120;  ## 2 minutes.
  }
  disk {
    on-io-error detach;
    fencing resource-only;
  }
  net {
    max-buffers 128;
    cram-hmac-alg "sha256";
    shared-secret "geheim";
    after-sb-0pri disconnect;
    after-sb-1pri disconnect;
    after-sb-2pri disconnect;
    rr-conflict disconnect;
  }

...
```

Sonst Problem mit OOM in dom0!

```
...

  syncer {
    rate 50M;
    after "xen-ns1";
  }

  on jupiter1 {
    device      /dev/drbd10;
    disk        /dev/mapper/xendomains-web1;
    address     192.168.255.30:7798;
    meta-disk   internal;
  }

  on jupiter2 {
    device      /dev/drbd10;
    disk        /dev/mapper/xendomains-web1;
    address    192.168.255.29:7798;
    meta-disk  internal;
  }
}
```

Crashs, Lockups, etc. bei vielen Syncs gleichzeitig

DRBD über LVM (über MD), also eine DRBD Ressource pro LV

LVM und I/O Konfiguration

/etc/lvm/lvm.conf

```
devices {
    dir = "/dev"
    scan = [ "/dev" ]
    filter = [ "a|^/dev/md5$|" ]
    cache = "/etc/lvm/.cache"
    write_cache_state = 1
    sysfs_scan = 1
    md_component_detection = 1
}
```

Xen Konfiguration

/etc/xen/xend-config.sxp

```
(xend-relocation-port 8002)
(xend-relocation-hosts-allow '^10\.50\.50\.')
(network-script network-bridge)
(vif-script vif-bridge)
(dom0-min-mem 500)
(dom0-cpus 0)
```

/etc/sysctl.conf

```
kernel.panic_on_oops = 5
kernel.panic = 5

dev.raid.speed_limit_min = 5000
dev.raid.speed_limit_max = 10000
```

Xen Konfiguration

/etc/xen/web1.cfg

```
kernel  = '/boot/vmlinuz-2.6.18-4-xen-686'
ramdisk = '/boot/initrd.img-2.6.18-4-xen-686'
memory  = '384'
maxmem = '416'

root    = '/dev/sda1 ro'
disk    = [ 'drbd:xen-web1,sda1,w' ]

name    = 'web1'
vif  = [ 'bridge=xenbr0,mac=02:00:00:00:00:08' ]

on_poweroff = 'destroy'
on_reboot   = 'restart'
on_crash    = 'restart'
```

Höher als memory, sonst Problem mit Netzwerk

Gast erzeugen

- Debian: cdebootstrap, xen-create-image

Empfehlung: minimales Template erstellen und wiederverwenden

Gast manuell verwalten

- Starten: `xm create web1.cfg`
- Stoppen:
  ```
  xm shutdown web1
  xm destroy web1
  ```
- Konsole: `xm console web1`

VServer Konfiguration

/etc/vserver/web1/apps/init/style

```
plain
```

/etc/vserver/web1/bcapabilities

```
CAP_SYS_RESOURCE
```

/etc/vserver/web1/context

```
10
```

/etc/vserver/web1/fstab

```
none  /proc    proc    defaults              0 0
none  /tmp     tmpfs   size=16m,mode=1777    0 0
none  /dev/pts devpts  gid=5,mode=620        0 0
```

/etc/vserver/web1/name

```
web1
```

/var/lib/vservers

- Basisverzeichnis, kann gleich sein wie für Xen domU
- Unterschied: Xen domU verwendet Filesystem direkt (z.B. ext3), für VServer vorher mounten!

/etc/security/limits.conf

```
*       hard        sigpending      32
*       hard        msgqueue        204800
root    hard        sigpending      32
root    hard        msgqueue        204800
```

VServer Konfiguration

/etc/init.d/checkroot.sh

```
@@ -141,7 +141,7 @@
        then
                ddev="$(mountpoint -qx $rootdev)"
                rdev="$(mountpoint -d /)"
-               if [ "$ddev" != "$rdev" ] && [ "$ddev" != "4:0" ]
+               if [ "$ddev" != "$rdev" ] && [ "$ddev" != "4:0" ] && [ -e /proc/cmdline ]
                then
                        if [ "$(mountpoint -qx /dev/root)" = "4:0" ]
                        then
@@ -340,7 +340,7 @@
          /proc/*)
                ;;
          /*)
-               if touch "$MTAB_PATH" >/dev/null 2>&1
+               if [ "$rdev" != "147:0" ] && touch "$MTAB_PATH" >/dev/null 2>&1
                then
                        :> "$MTAB_PATH"
                        rm -f ${MTAB_PATH}~
```

Heartbeat Konfiguration

/etc/heartbeat/ha.cf

```
debugfile /var/log/ha-debug
logfile /var/log/ha-log
logfacility     none
keepalive 500ms
deadtime 30
warntime 5
initdead 60
udpport 694
bcast eth1
bcast xenbr0
node    sun
node    jupiter1
node    jupiter2
ping 10.50.50.1
apiauth         pingd       uid=hacluster
respawn         hacluster
            /usr/lib/heartbeat/pingd -m 100 -d 5s

respawn hacluster /usr/lib/heartbeat/dopd
apiauth dopd gid=haclient uid=hacluster
crm yes
deadping 60
debug 0
use_logd yes
```

Sonst syslog voll

schnelle Reaktion

Version 2 Konfiguration

/etc/heartbeat/authkeys

```
auth 1
1 sha1 auchgeheim
```

/etc/heartbeat/cib.xml

```
Here it gets interesting...
```

Heartbeat Konfiguration: CIB Basis

/etc/heartbeat/cib.xml

Bei manuellen Änderungen erhöhen

```
<cib admin_epoch="25" epoch="1" num_updates="0" have_quorum="true">
  <configuration>
    <crm_config>
      <cluster_property_set id="default">
        <attributes>
          <nvpair id="require_quorum" name="require_quorum" value="false"/>
          <nvpair id="no-quorum-policy" name="no-quorum-policy" value="freeze"/>
          <nvpair id="symmetric_cluster" name="symetric_cluster" value="true"/>
          <nvpair id="default-resource-stickiness" name="default-resource-stickiness" value="300"/>
          <nvpair id="default-resource-failure-stickiness" name="default-resource-stickiness"
value="-100"/>
          <nvpair id="stop-orphan-resources" name="stop-orphan-resources" value="false"/>
          <nvpair id="short_resource_names" name="short_resource_names" value="true"/>
          <nvpair id="remove_after_stop" name="remove_after_stop" value="false"/>
          <nvpair id="is_managed_default" name="is_managed_default" value="true"/>
        </attributes>
      </cluster_property_set>
    </crm_config>

    <nodes>  <!-- this will be filled automatically by heartbeat -->  </nodes>
  </configuration>
  <resources />
  <status />
</cib>
```

„true“ bei 3 Nodes

Hin- und Herspringen verhindern

Heartbeat Konfiguration: DRBD Ressource (Xen)

/etc/heartbeat/cib.d/cib-resource-drbd-xen-web1.xml

```
      <!-- this is a clone and not a master-slave because the xen drbd
         resource script will manage promotion -->
       <clone id="c-drbd_web1">
         <meta_attributes id="c-ms-drbd_web1">
           <attributes>
             <nvpair id="c-ms-drbd_web1-1" name="clone_max" value="2"/>
             <nvpair id="c-ms-drbd_web1-2" name="clone_node_max" value="1"/>
             <nvpair id="c-ms-drbd_web1-5" name="notify" value="yes"/>
             <nvpair id="c-ms-drbd_web1-6" name="globally_unique" value="false"/>
<!--         <nvpair id="c-ms-drbd_web1-7" name="target_role" value="stopped"/> -->
           </attributes>
         </meta_attributes>
         <primitive id="drbd_web1" class="ocf" provider="heartbeat" type="drbd">
           <instance_attributes id="ia-drbd_web1">
             <attributes>
               <nvpair id="ia-drbd_web1-1" name="drbd_resource" value="xen-web1"/>
             </attributes>
           </instance_attributes>
         </primitive>
       </clone>
```

Heartbeat Konfiguration: DRBD Constraints (Xen)

/etc/heartbeat/cib.d/cib-constraints-drbd-xen-web1.xml

```
<rsc_location id="c-drbd_web1-on-jupiters" rsc="c-drbd_web1">
     <!-- i.e. it may not run on sun at all -->
     <rule id="pref-c-drbd_web1-on-sun" score="-INFINITY">
         <expression id="pref-c-drbd_web1-on-sun-1" attribute="#uname" operation="eq" value="sun"/>
     </rule>
</rsc_location>
```

Heartbeat Konfiguration: DRBD Ressource (VServer)

/etc/heartbeat/cib.d/cib-resource-drbd-vserver-web1.xml

```
      <master_slave id="ms-drbd_web1">
        <meta_attributes id="ma-ms-drbd_web1">
          <attributes>
            <nvpair id="ma-ms-drbd_web1-1" name="clone_max" value="2"/>
            <nvpair id="ma-ms-drbd_web1-2" name="clone_node_max" value="1"/>
            <nvpair id="ma-ms-drbd_web1-3" name="master_max" value="1"/>
            <nvpair id="ma-ms-drbd_web1-4" name="master_node_max" value="1"/>
            <nvpair id="ma-ms-drbd_web1-5" name="notify" value="yes"/>
            <nvpair id="ma-ms-drbd_web1-6" name="globally_unique" value="false"/>
<!--        <nvpair id="ma-ms-drbd_web1-7" name="target_role" value="stopped"/>-->
          </attributes>
        </meta_attributes>
        <primitive id="drbdms_web1" class="ocf" provider="heartbeat" type="drbd">
          <instance_attributes id="ia-drbdms_web1">
            <attributes>
              <nvpair id="ia-drbdms_web1-1" name="drbd_resource" value="xen-web1"/>
            </attributes>
          </instance_attributes>
        </primitive>
      </master_slave>
```

Heartbeat Konfiguration: DRBD Constraints (VServer)

/etc/heartbeat/cib.d/cib-constraints-drbd-vserver-web1.xml

```
<rsc_location id="ms-drbd_web1-on-jupiters" rsc="ms-drbd_web1">
     <!-- i.e. it may not run on sun at all -->
     <rule id="pref-ms-drbd_web1-on-sun" score="-INFINITY">
         <expression id="pref-ms-drbd_web1-on-sun-1" attribute="#uname" operation="eq" value="sun"/>
     </rule>
</rsc_location>
```

/etc/heartbeat/cib.d/cib-constraints-conflict-drbd-web1.xml

```
<rsc_colocation id="drbd_web1-xen-vs-vserver" to="c-drbd_web1" to_role="started" from="ms-drbd_web1"
from_role="started" score="-infinity"/>
```

Heartbeat Konfiguration: Xen Ressource

/etc/heartbeat/cib.d/cib-resource-xen-web1.xml

```
      <primitive class="ocf" provider="heartbeat" type="Xen" id="web1">
        <meta_attributes id="ma-xen_web1">
          <attributes>
<!--        <nvpair name="target_role" id="ma-xen_web1-1" value="stopped"/> -->
          <!-- TODO: enable later on -->
            <nvpair id="ma-xen_web1-2" name="allow_migrate" value="false"/>
          </attributes>
        </meta_attributes>
        <operations>
           <op id="op-xen_web1-mon" interval="10s" name="monitor" timeout="240s"/>
           <op id="op-xen_web1-start" name="start" timeout="240s"/>
           <op id="op-xen_web1-stop" name="start" timeout="360s"/>
           <op id="op-xen_web1-status" name="status" timeout="240s"/>
        </operations>

        <instance_attributes id="ia-xen_web1">
          <attributes>
            <nvpair id="ia-xen_web1-1" name="xmfile" value="/etc/xen/web1.cfg"/>
          </attributes>
        </instance_attributes>
      </primitive>
```

Heartbeat Konfiguration: Xen Constraints

/etc/heartbeat/cib.d/cib-constraints-xen-1-web1.xml

```
<rsc_order id="drbd_web1-before-xen_web1" from="web1" action="start" to="c-drbd_web1" to_action="start"/>
```

/etc/heartbeat/cib.d/cib-constraints-xen-2-web1.xml

```
<rsc_colocation id="xen_web1-on-drbd_web1" to="c-drbd_web1" to_role="started" from="web1"
score="infinity"/>
```

/etc/heartbeat/cib.d/cib-constraints-xen-3-web1.xml

```
<rsc_location id="xen_web1:connected" rsc="web1">
  <rule id="xen_web1:connected:rule" score_attribute="pingd" >
      <expression id="xen_web1:connected:expr:defined"
            attribute="pingd" operation="defined"/>
   </rule>
</rsc_location>
```

/etc/heartbeat/cib.d/cib-constraints-xen-4-web1.xml

```
<rsc_location id="xen_web1-on-jupiters" rsc="web1">
     <rule id="pref-xen_web1-on-sun" score="-INFINITY">
         <expression id="pref-xen_web1-on-sun-1" attribute="#uname" operation="eq" value="sun"/>
     </rule>
</rsc_location>
```

Heartbeat Konfiguration: VServer Ressource

/etc/heartbeat/cib.d/cib-resource-vserver-web1.xml

```
<primitive id="vserver_web1" class="ocf" type="VServer" provider="local" restart_type="restart">
  <instance_attributes id="vserver_web1">
      <attributes>
          <nvpair id="vserver_web1:name" name="vserver" value="web1"/>
      </attributes>
  </instance_attributes>
  <operations>
      <op id="op-vserver_web1-mon" interval="10s" name="monitor" timeout="30s"/>
      <op id="op-vserver_web1-start" name="start" timeout="60s"/>
      <op id="op-vserver_web1-stop" name="stop" timeout="180s"/>
      <op id="op-vserver_web1-status" name="status" timeout="30s"/>
  </operations>
</primitive>
```

/etc/heartbeat/cib.d/cib-resource-fs-vserver-web1.xml

```
<primitive class="ocf" provider="heartbeat" type="Filesystem" id="fs_web1">
  <instance_attributes id="ia-fs_web1">
    <attributes>
      <nvpair id="ia-fs_web1-1" name="fstype" value="ext3"/>
      <nvpair id="ia-fs_web1-2" name="directory" value="/var/lib/vservers/web1"/>
      <nvpair id="ia-f0_web1-3" name="device" value="/dev/drbd10"/>
    </attributes>
  </instance_attributes>
</primitive>
```

Heartbeat Konfiguration: VServer Constraints

/etc/heartbeat/cib.d/cib-constraints-vserver-1-web1.xml

```
<rsc_order id="fs_web1-before-vserver_web1" from="vserver_web1" action="start" to="fs_web1"
to_action="start"/>
```

/etc/heartbeat/cib.d/cib-constraints-vserver-2-web1.xml

```
<rsc_colocation id="vserver_web1-on-fs_web1" to="fs_web1" to_role="start" from="vserver_web1"
score="infinity"/>
```

/etc/heartbeat/cib.d/cib-constraints-vserver-3-web1.xml

```
<rsc_location id="vserver_web1:connected" rsc="vserver_web1">
  <rule id="vserver_web1:connected:rule" score_attribute="pingd" >
        <expression id="vserver_web1:connected:expr:defined"
            attribute="pingd" operation="defined"/>
   </rule>
</rsc_location>
```

/etc/heartbeat/cib.d/cib-constraints-vserver-4-web1.xml

```
<rsc_location id="vserver_web1-on-jupiters" rsc="vserver_web1">
     <rule id="pref-vserver_web1-on-sun" score="-INFINITY">
         <expression id="pref-vserver_web1-on-sun-1" attribute="#uname" operation="eq" value="sun"/>
     </rule>
</rsc_location>
```

Heartbeat Konfiguration: VServer Constraints

/etc/heartbeat/cib.d/cib-constraints-fs-vserver-1-web1.xml

```
<rsc_order id="drbd_web1-before-fs_web1" from="fs_web1" action="start" to="ms-drbd_web1"
to_action="promote"/>
```

/etc/heartbeat/cib.d/cib-constraints-fs-vserver-2-web1.xml

```
<rsc_colocation id="fs_web1-on-drbd_web1" to="ms-drbd_web1" to_role="master" from="fs_web1"
score="infinity"/>
```

/etc/heartbeat/cib.d/cib-constraints-fs-vserver-3-web1.xml

```
<rsc_location id="fs_web1-on-jupiters" rsc="fs_web1">
     <rule id="pref-fs_web1-on-sun" score="-INFINITY">
         <expression id="pref-fs_web1-on-sun-1" attribute="#uname" operation="eq" value="sun"/>
     </rule>
</rsc_location>
```

/etc/heartbeat/cib.d/cib-constraints-conflict-xen-vserver-web1.xml

```
<rsc_colocation id="xen_web1-vs-vserver_web1" to="vserver_web1" to_role="started" from="web1"
from_role="started" score="-infinity"/>
```

Alle Snippets zusammenfügen

- Verifizieren der „Basis“:
  `cat /etc/heartbeat/cib.xml | crm_verify`
- Updates an der „Basis“:
  `cibadmin -U -x /etc/heartbeat/cib.xml`
- Neue Ressourcen einfügen:
  `cibadmin -C -o resources -x cib-resource-XYZ.xml`
- Neue Constrains einfügen:
  `cibadmin -C -o constraints -x cib-constraints-XYZ.xml`
- Ressourcen aktivieren:
  `crm_resource -r <resource ID> -p target_role -v started`
- vieles mehr...

Und der Rest?

Nicht vergessen **für alle virtuellen Hosts**:

- zentrales Logging (syslog-ng)
- Logauswertung (logcheck, logwatch, etc.)
- HIDS (osiris, aide, tripwire, prelude, etc.)
- Sicherheitsupdates (apt-cron, etc.)

und die physikalischen:

- Logauswertung
- Hardware-Monitoring
- RAID Festplattenwechel und Überwachung

Hört sich nach viel Arbeit an...

Automatisieren!

- automatische Verteilung der Konfiguration auf alle Hosts
- automatisches Erzeugen neuer virtueller Hosts
  - Logical Volume, File System, Template einspielen
  - Grundkonfiguration (Hostname, IP-Adresse, MAC-Adresse)
  - DRBD und Heartbeat Konfiguration

- Aktuell im Einsatz: puppet, Details auf Anfrage

TO DO und zukünftige Erweiterungen

- Xen-DRBD-LVM-MD Kombination stabilisieren (Kernel Updates?)
- neuere VServer-Version verwenden – native IPv6
- I/O Verhalten optimieren (mehr Festplatten?)
- weitere Virtualisierungstechniken testen
  - Virtualbox
  - OpenVZ
  - KVM
- Automatisierungsgrad weiter erhöhen
  - Windows Gäste
- weitere Sicherheitsmaßnahmen (SELinux, zusätzliche HIDS+NIDS, sHype?)

Thank you for your attention!

Slides: http://www.mayrhofer.eu.org/presentations
Later questions: rene@mayrhofer.eu.org

OpenPGP key: 0xC3C24BDE
7FE4 0DB5 61EC C645 B2F1 C847 ABB4 8F0D C3C2 4BDE