Clustering und Failover mit Linux

Markus Oswald
<moswald@iirc.at>

Grazer Linux-Tage 2003 - 25. April

www.linuxtage.at

Worum geht es?

- Load-Balanced Cluster
- Failover Cluster
- Shared Storage

Worum es nicht geht?

- Computational Cluster
- Beowulf
- Distributed Computing

Markus Oswald, <moswald@iirc.at>

Inhalt

Theorie (I/II/III)

Anwendungen (I/II)

Load-Balancing Cluster
- Aufbau
- Software
- Konfiguration (I/II)
- High Availability

Failover Cluster
- Aufbau
- Software
- Funktion: heartbeat
- Funktion: DRBD
- Konfiguration

Denkbarer Einsatz

Beispiel-Installationen (I/II)

Markus Oswald, <moswald@iirc.at>

Theorie I

Warum Clustern?

- Ausfallsicherheit
  - Ein Server kann schnell ausfallen
  - Es geht nicht immer ohne rebooten (Hardware, Kernel)
- Lastverteilung
  - Ein Server schafft die Last oft einfach nicht mehr
- Skalierbarkeit
  - Ein Pool aus mehreren Servern lässt sich dynamisch den Anforderungen anpassen
- Kosten
  - Viele kleine Server sind oft billiger als ein großer

Markus Oswald, <moswald@iirc.at>

Theorie II

Cluster Typen:

Load-Balanced Cluster

Mehrere Server teilen sich die Last

Technik: LVS (round-robin DNS, iptables, ...)

Failover Cluster

Backup Server übernimmt bei Ausfall die Dienste

Technik: heartbeat (Failsave, Kimberlite, ...)

Markus Oswald, <moswald@iirc.at>

Theorie III

Typischer Einsatz:

Load-Balanced Cluster
- Webserver
- Mailserver

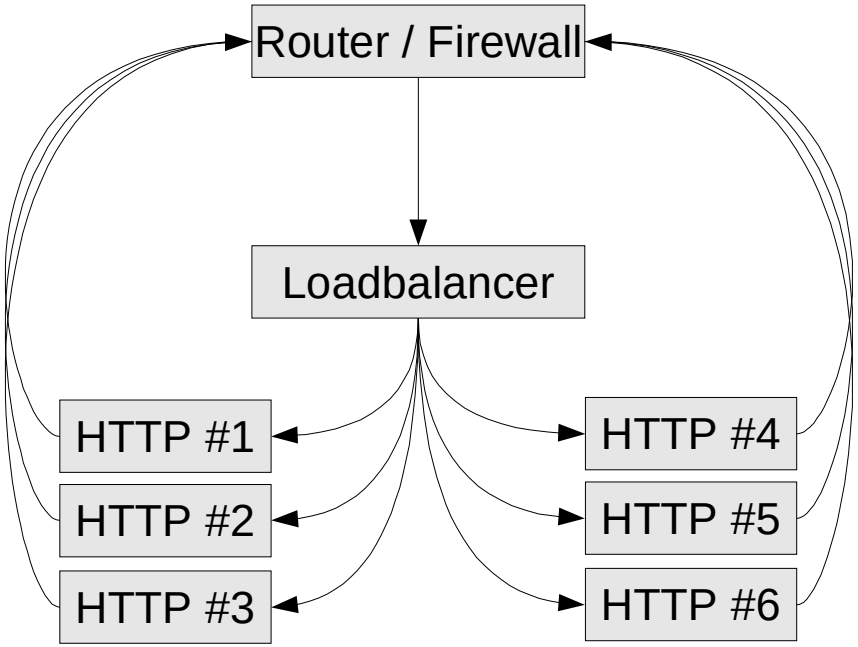
Failover Cluster
- DB Server
- Fileserver
- Firewall / Router

Markus Oswald, <moswald@iirc.at>

Anwendungen I

Load Balanced:

Webserver-Farm

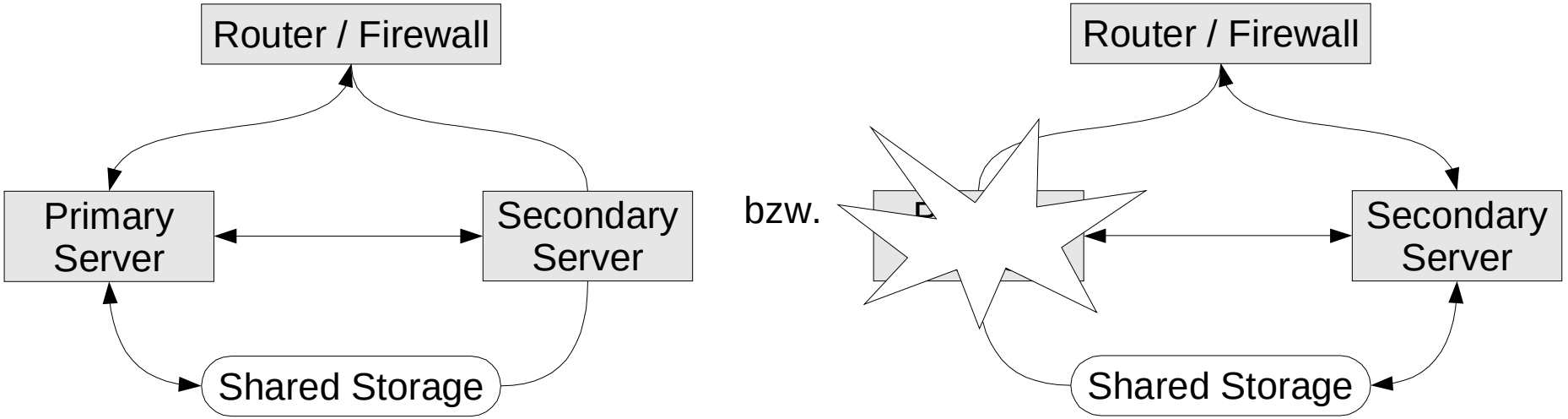


Markus Oswald, <moswald@iirc.at>

Anwendungen II

Failover:

DB Server



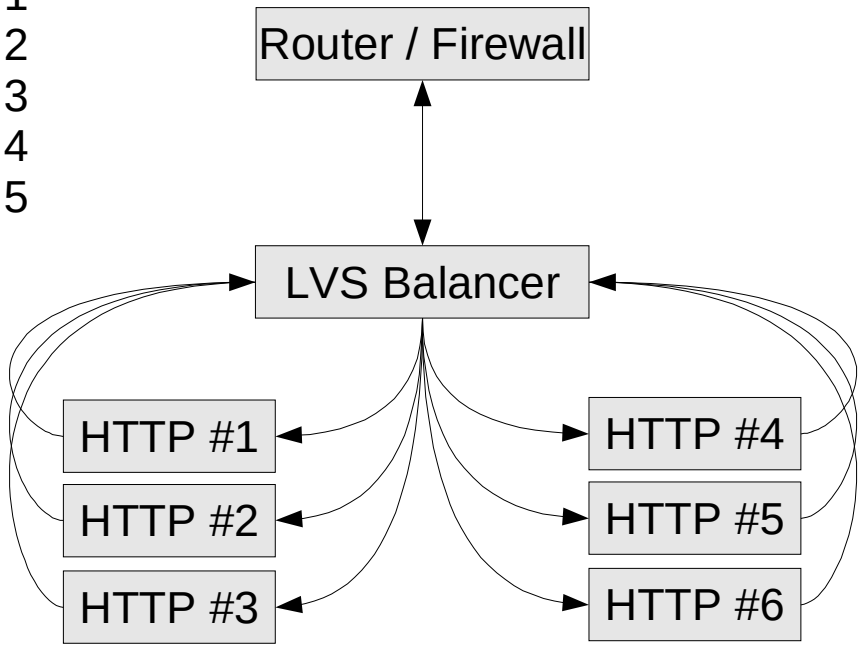
Markus Oswald, <moswald@iirc.at>

Load Balancing – Aufbau

Bei NAT Forwarding:

LVS Balancer:	eth0:	207.175.44.110
	eth1:	192.168.10.250
HTTP #1:	eth0:	192.168.10.1
HTTP #2:	eth0:	192.168.10.2
HTTP #3:	eth0:	192.168.10.3
HTTP #4:	eth0:	192.168.10.4
HTTP #5:	eth0:	192.168.10.5

....



Markus Oswald, <moswald@iirc.at>

Load Balancing – Software

Kernel:

- Linux 2.2.x mit IPVS Patch
- Linux 2.4.x mit IPVS Patch
- Linux 2.4.x mit Netfilter und IPVS Netfilter Modulen

Konfiguration:

- ipvsadm

High-Availability:

- Keepalived

Markus Oswald, <moswald@iirc.at>

Load Balancing – Konfiguration I

Konfiguration mittels ipvsadm ist recht einfach:

```
ipvsadm -A -t 207.175.44.110:80 -s rr
ipvsadm -a -t 207.175.44.110:80 -r 192.168.10.1:80 -m
ipvsadm -a -t 207.175.44.110:80 -r 192.168.10.2:80 -m
ipvsadm -a -t 207.175.44.110:80 -r 192.168.10.3:80 -m
ipvsadm -a -t 207.175.44.110:80 -r 192.168.10.4:80 -m
ipvsadm -a -t 207.175.44.110:80 -r 192.168.10.5:80 -m
```

Verteilt alle Verbindungen auf 207.175.44.110:80 gleichmäßig über 192.168.10.1 bis 192.168.10.5

```
bash$ ipvsadm -L -n
IP Virtual Server version 1.0.2 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  207.175.44.110:80 rr
  -> 192.168.10.1:80              Masq    10     4          29
  -> 192.168.10.2:80              Masq    10     3          24
  -> 192.168.10.3:80              Masq    10     4          25
  -> 192.168.10.4:80              Masq    10     3          27
  -> 192.168.10.5:80              Masq    10     4          21
```


Markus Oswald, <moswald@iirc.at>

Load Balancing – Konfiguration II

Weitere Optionen:

`-f, --fwmark-service`
Firewall-Mark anstelle von Adresse, Port und Protokoll verwenden um virtuelle Server zu unterscheiden

`-s, --scheduler`
Algorithmus zur Verteilung der Verbindungen. Derzeit gibt es 7 Möglichkeiten:

- rr - Robin Robin
- wrr - Weighted Round Robin
- lc - Least-Connection
- wlc - Weighted Least-Connection
- lblcr - Locality-Based Least-Connection with Replication
- dh - Destination Hashing
- sh - Source Hashing

`-p, --persistent`
Mehrere Requests eines Clients kommen immer zum gleichen Server.
Sinnvoll bei FTP oder SSL Verbindungen

`-w, --weight`
Kapazität eines Servers – z.B. bei unterschiedlich starken Servern im Pool

Markus Oswald, <moswald@iirc.at>

Load Balancing – High Availability

Hochverfügbarkeit mittels Keepalived:

- Überwacht Services der Server
- Entfernt Server aus dem Pool oder fügt diese wieder hinzu
- Benachrichtigt den Administrator über Ausfäll via Email
- Handelt LVS Failover bei redundanten Setups

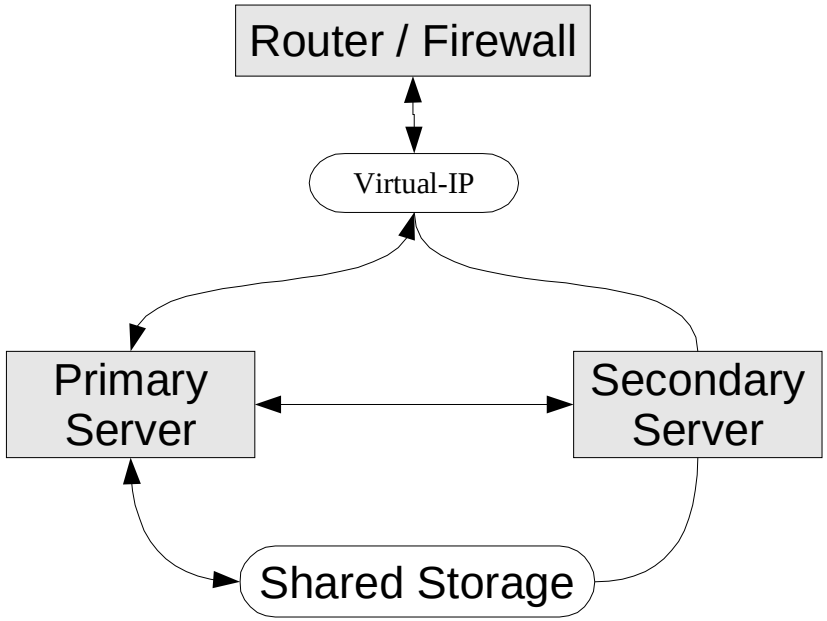
Konfiguration einigermaßen einfach und gut dokumentiert:

```
virtual_server 207.175.44.110 80 {
delay_loop 10
lb_algo rr
lb_kind NAT
nat_mask 255.255.255.0
protocol TCP

  real_server 192.168.10.1 80 {
      weight 10
      HTTP_GET {
          url {
            path /status/status.php
            digest c4ca4238a0b923820dcc509a6f75849b
          }
          connect_timeout 10
          connect_port 80
          nb_get_retry 3
          delay_before_retry 3
      }
  }
}
```


Markus Oswald, <moswald@iirc.at>

Failover – Aufbau




```
eth0      192.168.1.10                      eth0      192.168.1.11
eth0:0    192.168.1.15    - - - - - - ->
eth1      10.0.0.1                          eth1      10.0.0.2
```


Markus Oswald, <moswald@iirc.at>

Failover – Software

Kernel:

- Linux 2.2.x
- Linux 2.4.x

High-Availability:

- heartbeat

Shared-Storage:

- DRBD (Kernel-Patch)
- Hardware (externes SCSI Array, SAN, ...)
- rsync (z.B. für Firewalls, Router)

Markus Oswald, <moswald@iirc.at>

Failover – Funktion: heartbeat

Was macht heartbeat?

- Host monitoring via serial, UDP, PPP/UDP heartbeats
- IP address takeover
- Ressourcen Management
- Unterstützt STONITH

Markus Oswald, <moswald@iirc.at>

Failover – Funktion: DRBD

Was macht DRBD:

- Wird als Block-Device angesprochen, daher völlig transparent
- Daten sind via Netzwerk gespiegelt
- Arbeitet "auf" Partitionen oder ganzen Platten
- Dateisystemunabhängig
- Zwischen 50 und 98% des theoretisch maximal möglichen Datendurchsatzes

"RAID 1 over Ethernet"

Markus Oswald, <moswald@iirc.at>

Failover – Konfiguration

Failover Konfiguration mit heartbeat und DRBD:

- Installation des Base-Systems (2x mögl. idente Maschinen)
- Kernel mit DRBD Patch übersetzen
- Partition für DRBD vorbereiten, formatieren, synchronisieren
- heartbeat konfigurieren (Dienste + DRBD!)
- Failover testen
  - sind alle Daten konsistent?
  - starten alle Dienste?
  - funktioniert das Failback so wie es soll?
- Stresstesting

Markus Oswald, <moswald@iirc.at>

Denkbarer Einsatz

LVS:

- Webserver - HTTP / HTTPS
- Mailserver – SMTP / POP / IMAP
- Proxy-Server

heartbeat:

- Firewalls
- Loadbalancer
- Router

heartbeat + DRBD:

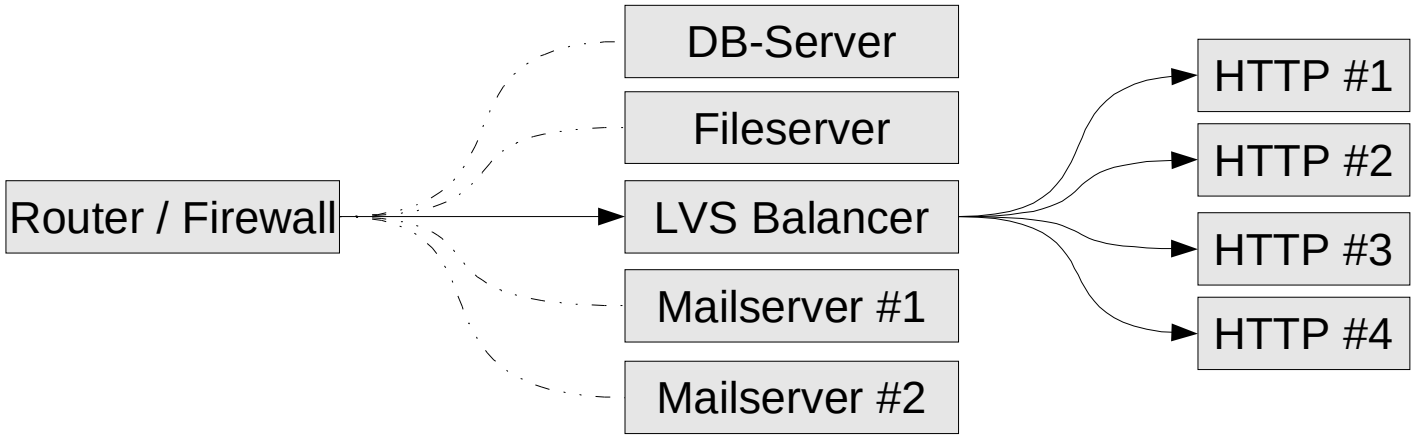
- Datenbanken – MySQL / PostgreSQL
- Webserver
- Mailserver
- Fileserver

Markus Oswald, <moswald@iirc.at>

Beispiel-Installationen I

InBox.cc – Online Büro

Balancer CPU-Last < 5% bei Spitzen von +10 Mbit



LVS Balancer: Debian 3.0, Kernel 2.4.18 mit LVS Patch

Webserver: Debian 3.0, Windows 2000

Markus Oswald, <moswald@iirc.at>

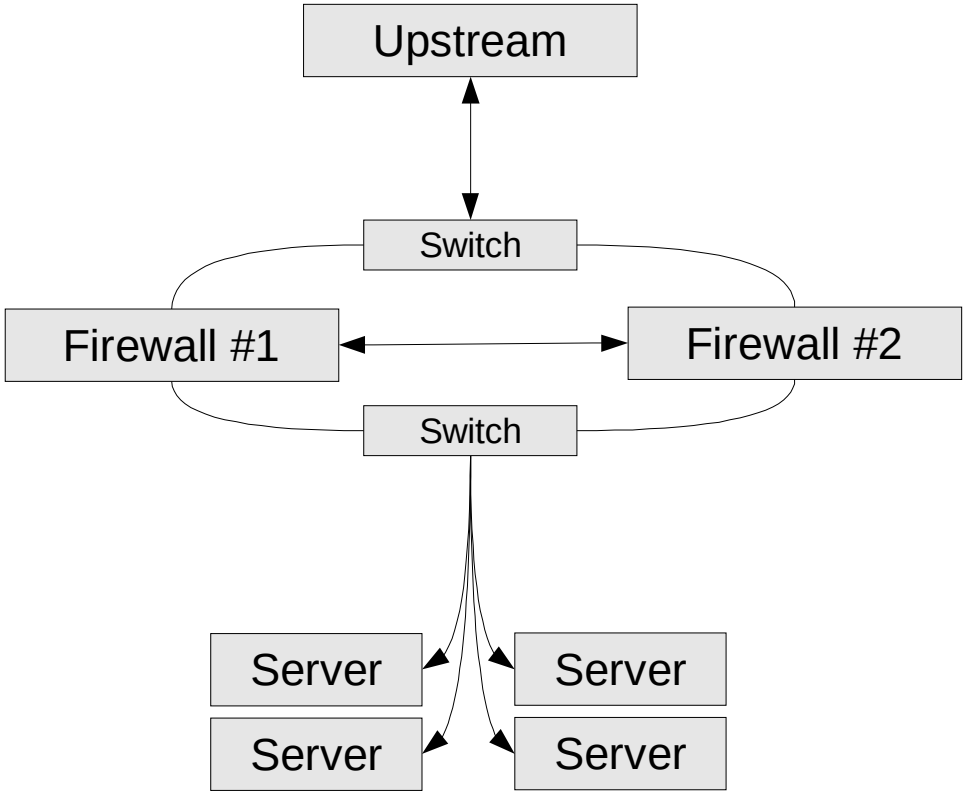
Beispiel-Installationen II

EDIS.AT – Internet Provider

Firewalls:
Gibraltar Linux 0.96

basierend auf Debian
Diskless Operation
(bootet von CD, Konfiguration auf Floppy)
Failoverzeiten < 10 Sekunden

Hardware:
FSC Primergy L100 Rack-Server



Markus Oswald, <moswald@iirc.at>

Fragen?

Markus Oswald, <moswald@iirc.at>

Links

LVS:	http://www.linuxvirtualserver.org/
Keepalived:	http://www.keepalived.org
heartbeat:	http://www.linux-ha.org/heartbeat/
DRBD:	http://www.linbit.com
Gibraltar:	http://www.gibraltar.at
Debian:	http://www.debian.org

E-Mail: Markus Oswald, <moswald@iirc.at>