High-Availability Linux

Using Heartbeat and DRBD to create a highly-available pair

by John Szakmeister
Intelesys Corp

High Availability Linux Project

"Provide a high-availability (clustering) solution for Linux which promotes reliability, availability, and serviceability (RAS) through a community development effort."

- Became suitable for mission-critical production use in 1999
- Estimated 30,000 installations or more around the world

HA Linux Project (cont.)

- Possible Components in an HA System
  - Membership Services
  - Communication Services
  - Cluster Management
  - Resource (I/O) Fencing
  - Resource Monitoring
  - Storage Sharing/Replication Storage Sharing

HA Linux Project (cont.)

- Components
  - Membership Services
  - Communication Services
  - Cluster Management
  - Resource (I/O) Fencing
  - Resource Monitoring
  - Storage Sharing/Replication Storage Sharing

Disk Replicating Block Device (DRBD)

- Method for replicating storage across a dedicated network
- From the DRBD homepage: *"You could see it as a network RAID-1"*
- Performs "Intelligent Resynchronization" when possible

How it works

- Stacks on top of a block device: `/dev/hda`, `/dev/md1`, etc
- All access takes place through new device: `/dev/drbd0`
- Supports two nodes: primary and secondary
- Starts up in secondary mode, and must be told to become primary

How it works (cont.)

- Secondary nodes basically act like listeners, but still maintain a constant connection between each other
- Once a node becomes primary, all blocks are sent over the network to the other listener
- The listener writes the data to disk
- All reads are carried out from the local block device

How it works (cont.)

- DRBD maintains a block map in the front 128MB of the block device you're stacked on top of
- Has a couple of algorithms for the transfer protocol:
  - Protocol A - Write is reported as completed if it has reached local disk and local tcp send buffer
  - Protocol B - Write is reported as completed if it has reached local disk and remote buffer cache
  - Protocol C - Write is reported as completed if it has reached both local and remote disk

How it works (cont.)

- Protocol C is best for transactional data
- Protocol B is geared to be the general use-case. However, benchmarks have shown Protocol C is actually faster, so the DRBD team recommends using Protocol C for now
- Protocol A is geared towards high-latency networks

Limitations

- Only supports two nodes.
  - Drbd+ (made by LINBIT) can support 3 nodes, but you must pay for it
- No authentication mechanism
  - Make sure to use a private network for the data transfers, if using DRBD locally
  - Use IPSEC or CIPE for long-range connections that provide authentication and encryption
- No Encryption
  - stunnel could be a solution here

Limitations (cont).

- Does *not* support clustered file systems, such as GFS
- Can’t mount the secondary node--even in read-only mode
  - Changes are occurring to the file system, without the mounted file system knowing about it

Features

- You don’t need an HA setup in order to use DRBD
- Can be very useful in a variety of other settings where you’re just concerned about data being replicated
  - Home Network
  - Important Corporate Data
- Easy to set up

Example drbd.conf

```
global {
  # Set the number of
  # available devices
  minor-count 5;
}

resource ha {
  protocol C;
  incon-degr-cmd “echo ‘!DRBD! pri on incon-degr’ | wall ; sleep 60 ; halt -f”

  startup {
    wfc-timeout 30;
    degr-wfc-timeout 120;
  }

  disk {
    on-io-error panic;
  }

  net {
    sndbuf-size 512k;
    max-buffers 2048;
    max-epoch-size 2048;
    on-disconnect reconnect;
  }
```

Example drbd.conf (cont)

```
  syncher {
    rate 128M;
    group 1;
    al-extents 257;
  }

  on doplhin {
    device /dev/drbd0;
    disk /dev/ha/ha0;
    address 192.168.0.1:7788;
    meta-disk internal;
  }

  on growler {
    device /dev/drbd0;
    disk /dev/ha/ha0;
    address 192.168.0.2:7788;
    meta-disk internal;
  }
}
```

Heartbeat

- The core product produced by linux-ha.org
- Implements death-of-node detection, communications, and cluster management in one process
- Runs on every Linux platform, as well as FreeBSD and Solaris

General Heartbeat Operation

- Runs as a daemon communicating to other nodes
- One node is appointed to be the primary node
- Other nodes stand ready to take over primary nodes services, should the primary node fail
- Resource Fencing: STONITH -- Shoot The Other Node In The Head

Heartbeat Configuration

- Heartbeat uses a set of /etc/init.d style scripts to start, stop, and check the status of services
- 2 configuration files
  - ha.cf -- configures general operation of Heartbeat
  - haresources -- configures the resources for the nodes

HA.CF (PART 1)

```
# Log to syslog
logfacility local0

# 2 seconds between heartbeats
keepalive 2

# If we don't hear from the other node in 30 seconds, consider it dead
deadtime 30

# Give ourselves 4 minutes to get everything up and running initially
initdead 240

# Failback to the primary server when it comes back online
auto_failback on

# Send heartbeats across both ethernet devices
bcast eth0 eth1

apiauth ipfail uid=hacluster
apiauth ccm uid=hacluster
apiauth cms uid=hacluster
apiauth ping gid=haclient uid=root
apiauth default gid=haclient
```

HA.CF (PART 2)

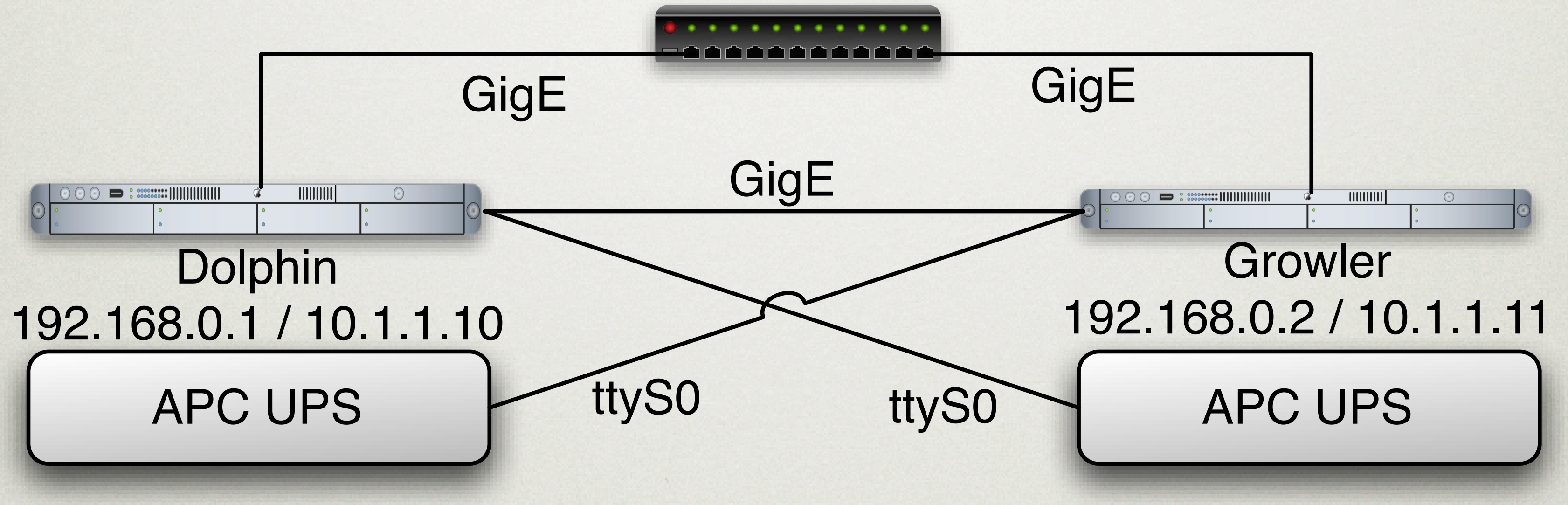
```
# Use the logging daemon
use_logd yes
conn_logd_time 60
compression bz2
compression_threshold 2

# Stonith configuration
stonith_host dolphin apcsmart /dev/ttyS0 growler
stonith_host growler apcsmart /dev/ttyS0 doplhin

# Lastly, define the nodes
node dolphin growler
```

HARESOURCES

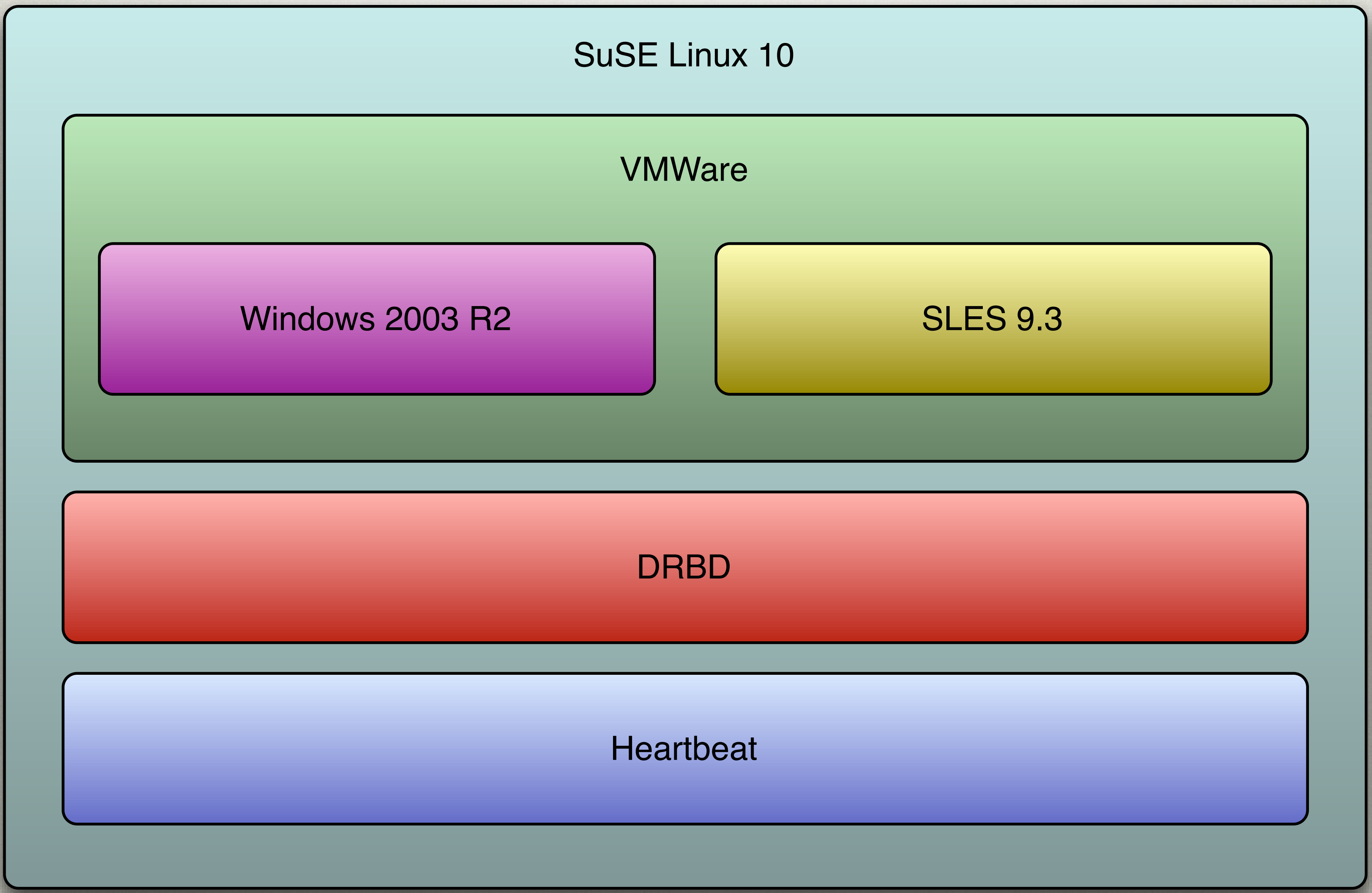
```
dolphin drbddisk::ha Filesystem::/dev/drbd0::/ha::reiserfs::acl,user,xattr 10.1.1.9 vmware
```

Physical Server Configuration

Active-Passive Server Configuration w/STONITH



Logical Server Configuration



SuSE Linux 10

VMWare

Windows 2003 R2

SLES 9.3

DRBD

Heartbeat

Resources

- Linux Magazine, November 2003 -- Whole magazine targeted HA solutions
- The Linux Enterprise Cluster, Karl Kopper, ISBN 1-59327-036-4