HALLO!

redhat.

RED HAT®
STORAGE

PLAYING NICE WITH OTHERS: Samba HA with Pacemaker

An Operetta in Three Parts

José A. Rivera
Software Engineer
SAMBA Team Member

2015.05.20

Overture

OVERTURE

Who's this guy?

José helps package and hack away at Samba full-time for Red Hat. He also kind of talks a lot.

- 9-ish years of working with Microsoft protocols
  - Even wrote some of the definitive documentation!
- Just about to complete his 1st year on the Samba Team
  - Has yet to run screaming
- Never driven a motorcycle

OVERTURE

Looking ahead

OVERTURE

Starting on the same page

HA - High Availability

- A characteristic of a system which says the system can be reliably used with a minimum of downtime.

Failover

- Switching from a failed service to a redundant service due to abnormal termination of the initial service.

Active/Active

- An HA cluster configuration in which failover of services occurs between always-on and (typically) homogenous software nodes.

TDB – Trivial Database

- Samba's primary DB backend.

CTDB – Clustered TDB

- A Samba project that provides a way of distributing its TDBs across clustered nodes.

VIPs – Virtual IP Addresses

- Also known as public IP addresses, these are IP addresses which clients will use to connect to the clustered services and can typically change which node they are assigned to.

Act I. Raccontare

HISTORY

HISTORY

The need for CTDB

Samba wanted a way to serve the same data from multiple nodes simultaneously.

- It was common before to do active/passive clustering using a distributed storage backend.
- Other open source clustered storage solutions at the time only offered POSIX semantics, which was a problem when you wanted to do SMB.
- Other database solutions did not meet the needs of Samba's workloads.

HISTORY

The need for CTDB

CTDB was built to bring active/active clustering to Samba.

- It needed to provide a number of things, including:
  - A common identity for all Samba instances
  - Synchronization of SMB/Windows metadata
  - Cross-node messaging
- To this day, relies on a separate, shared filesystem in its recovery mechanism to avoid split-brain scenarios.
  - In particular, it must implement proper POSIX byte-range locks; e.g. GPFS, GFS2

HISTORY

Refactoring: CTDB 2.0

In 2012, CTDB version 2.0 was released. This did a number of things:

- Consolidated a number of disparate maintenance branches.
- Lots of cool internal stuff (e.g. read-only records, performance optimizations, new test infrastructures).
- A strong push towards the modularization of CTDB's various features and functionality.

Huge thanks to Amitay Isaacs <amitay@samba.org> and Martin Schwenke <martin@meltin.net>!

HISTORY

Where are we going and why am I in this handbasket?

Modularization facilitates integration!

- Modularization allows for individual feature components of CTDB to be turned off without disrupting other components.
- This eases the integration of Samba into other clustered environments, as long as we provide those features we turned off elsewhere.
- Why not integrate Samba into a fully open source, Linux-based clustered environment?

HISTORY

Where are we going and why am I in this handbasket?

Modularization facilitates integration!

- Modularization allows for individual feature components of CTDB to be turned off without disrupting other components.
- This eases the integration of Samba into other clustered environments, as long as we provide those features we turned off elsewhere.
- Why not integrate Samba into a fully open source, Linux-based clustered environment?
  - Note the logo in the lower right-hand corner. :)

Act II. Cambiare

CHANGE

CHANGE

Introducing Pacemaker

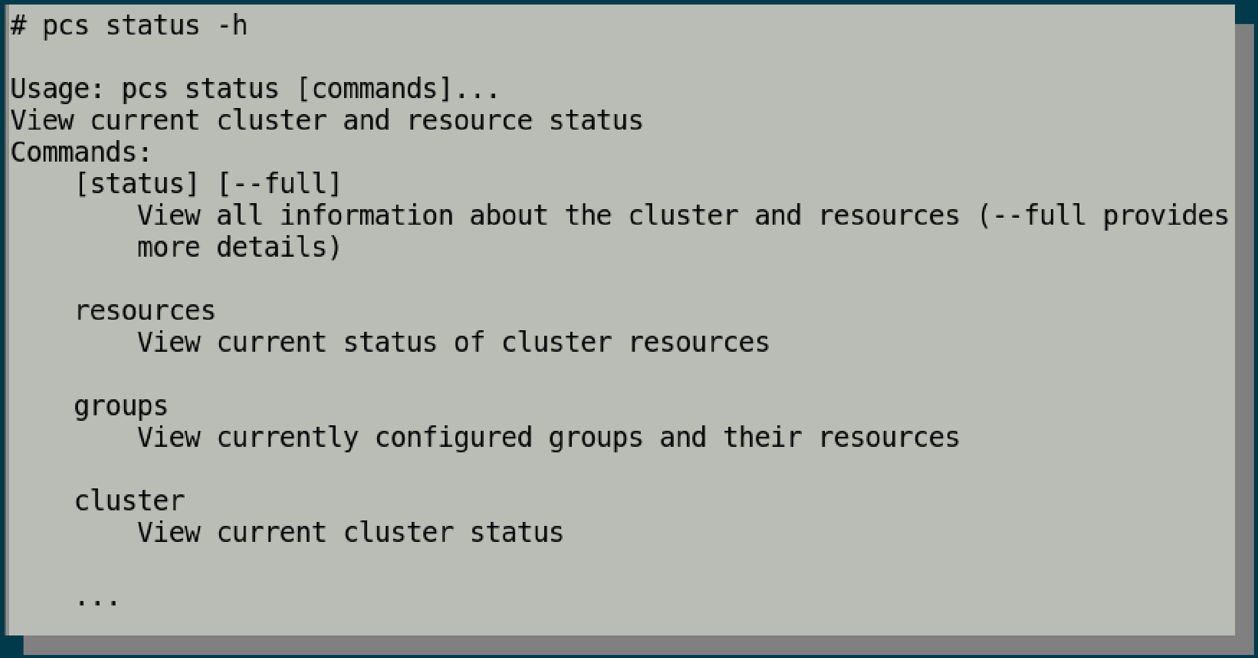
Pacemaker is a flexible and extensible HA resource manager...

- A “resource” is defined via a resource agent (RA).
  - RAs can be defined as anything from storage volumes to IP addresses to daemon processes.
- Resources can be centrally managed from a single interface, either from any node in the Pacemaker cluster or a remote management node.
- Resources (and nodes!) can have automated logging of and recovery from failures.

...and it's all extremely and easily configurable.

CHANGE

Introducing Pacemaker

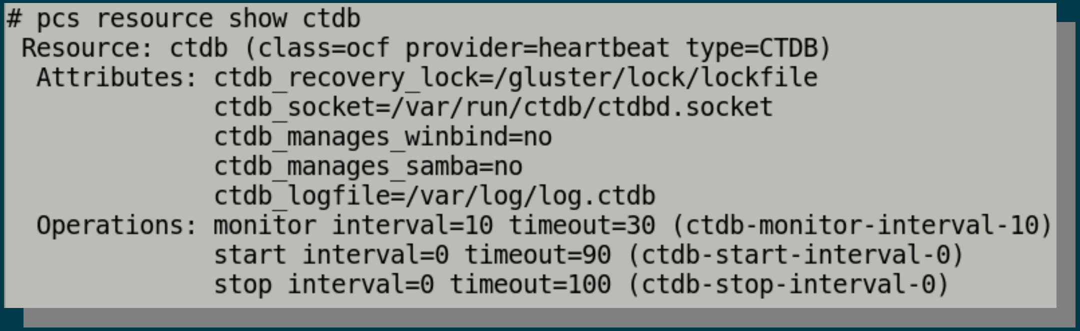


```
# pcs status -h

Usage: pcs status [commands]...
View current cluster and resource status
Commands:
    [status] [--full]
        View all information about the cluster and resources (--full provides
        more details)

    resources
        View current status of cluster resources

    groups
        View currently configured groups and their resources

    cluster
        View current cluster status

    ...
```

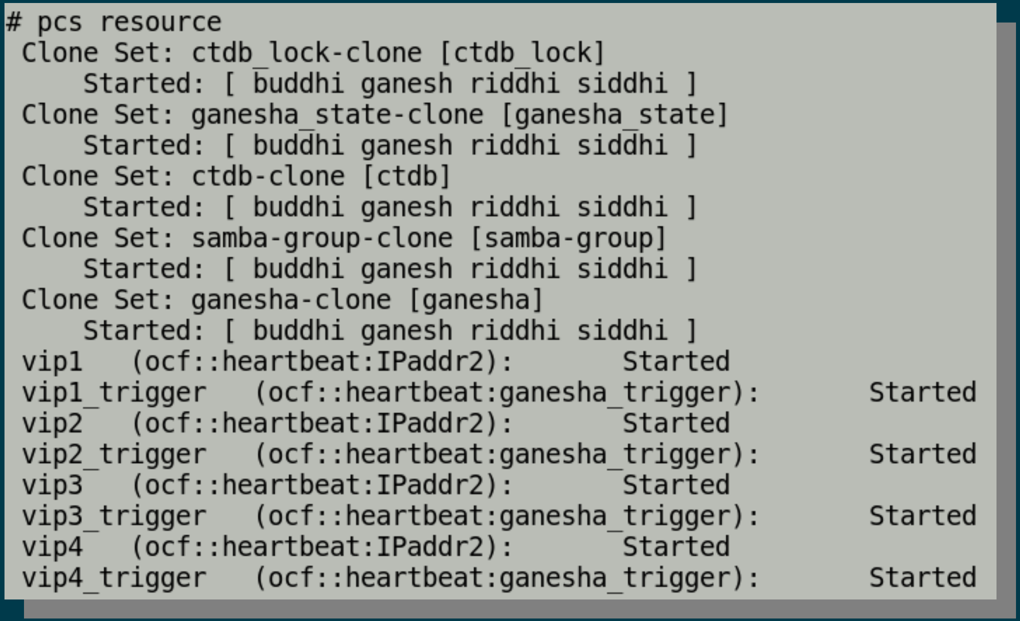
Pacemaker CLI Examples

CHANGE

Introducing Pacemaker



```
# pcs resource show ctdb
 Resource: ctdb (class=ocf provider=heartbeat type=CTDB)
  Attributes: ctdb_recovery_lock=/gluster/lock/lockfile
              ctdb_socket=/var/run/ctdb/ctdbd.socket
              ctdb_manages_winbind=no
              ctdb_manages_samba=no
              ctdb_logfile=/var/log/log.ctdb
  Operations: monitor interval=10 timeout=30 (ctdb-monitor-interval-10)
              start interval=0 timeout=90 (ctdb-start-interval-0)
              stop interval=0 timeout=100 (ctdb-stop-interval-0)
```

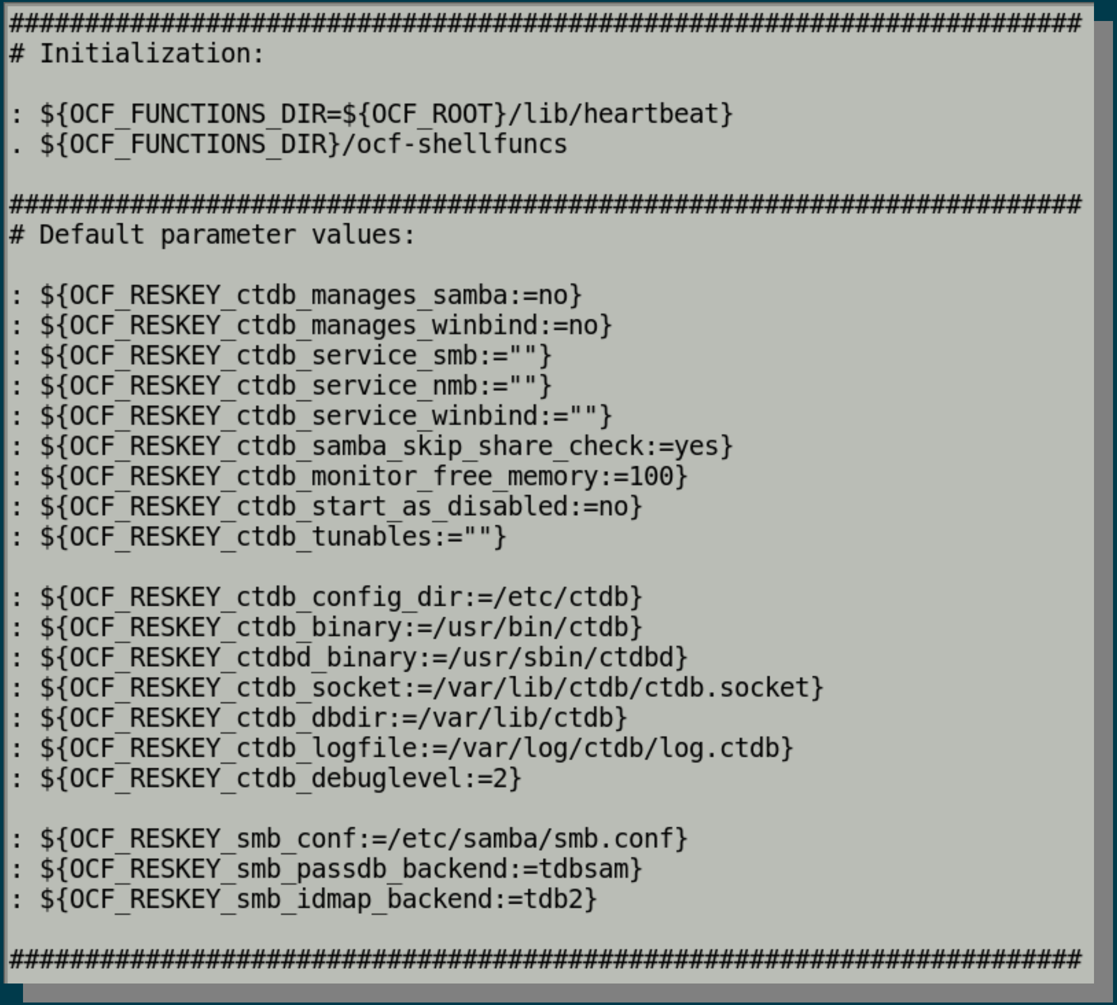


```
# pcs resource
 Clone Set: ctdb_lock-clone [ctdb_lock]
     Started: [ buddhi ganesh riddhi siddhi ]
 Clone Set: ganesha_state-clone [ganesha_state]
     Started: [ buddhi ganesh riddhi siddhi ]
 Clone Set: ctdb-clone [ctdb]
     Started: [ buddhi ganesh riddhi siddhi ]
 Clone Set: samba-group-clone [samba-group]
     Started: [ buddhi ganesh riddhi siddhi ]
 Clone Set: ganesha-clone [ganesha]
     Started: [ buddhi ganesh riddhi siddhi ]
 vip1   (ocf::heartbeat:IPaddr2):       Started
 vip1_trigger   (ocf::heartbeat:ganesha_trigger):       Started
 vip2   (ocf::heartbeat:IPaddr2):       Started
 vip2_trigger   (ocf::heartbeat:ganesha_trigger):       Started
 vip3   (ocf::heartbeat:IPaddr2):       Started
 vip3_trigger   (ocf::heartbeat:ganesha_trigger):       Started
 vip4   (ocf::heartbeat:IPaddr2):       Started
 vip4_trigger   (ocf::heartbeat:ganesha_trigger):       Started
```

Pacemaker CLI Examples

CHANGE

Introducing Pacemaker

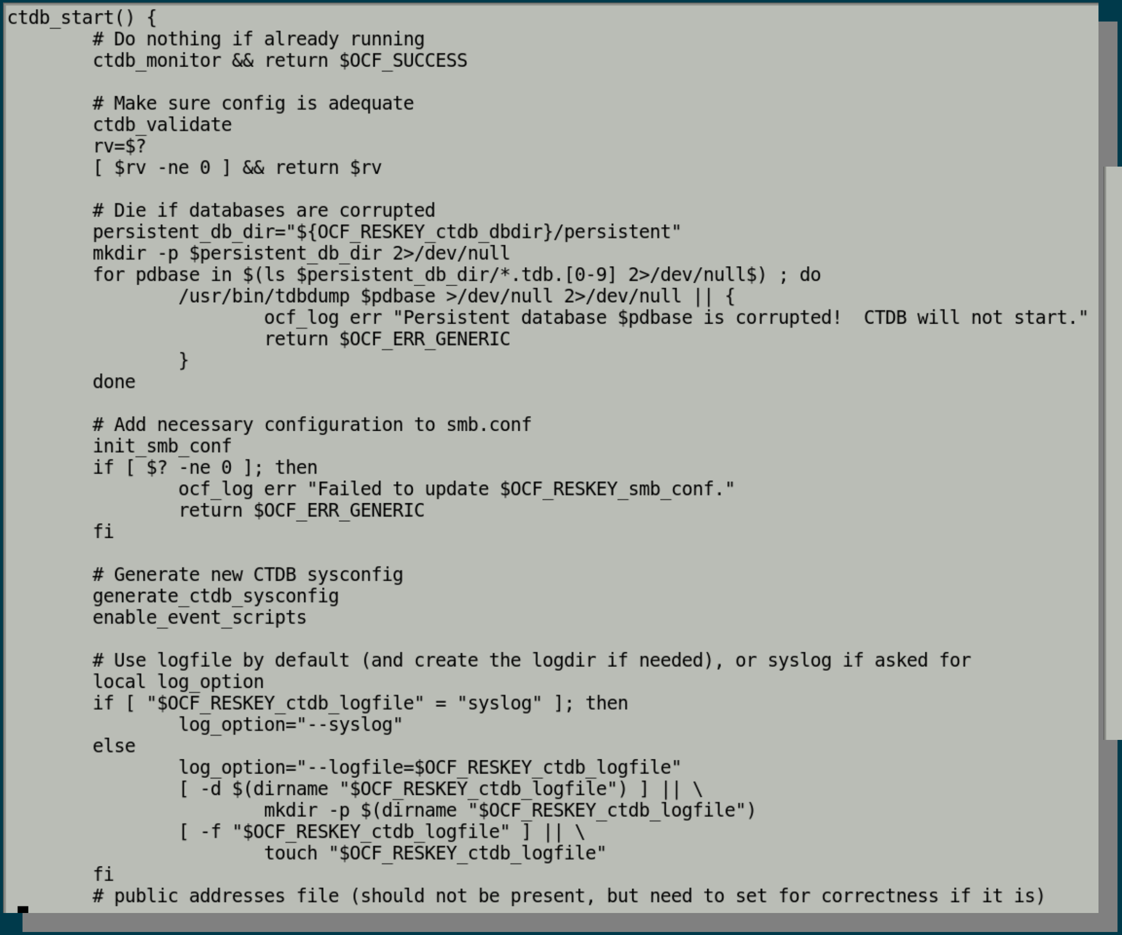


```
######################################################################
# Initialization:

: ${OCF_FUNCTIONS_DIR=${OCF_ROOT}/lib/heartbeat}
. ${OCF_FUNCTIONS_DIR}/ocf-shellfuncs

######################################################################
# Default parameter values:

: ${OCF_RESKEY_ctdb_manages_samba:=no}
: ${OCF_RESKEY_ctdb_manages_winbind:=no}
: ${OCF_RESKEY_ctdb_service_smb:=""}
: ${OCF_RESKEY_ctdb_service_nmb:=""}
: ${OCF_RESKEY_ctdb_service_winbind:=""}
: ${OCF_RESKEY_ctdb_samba_skip_share_check:=yes}
: ${OCF_RESKEY_ctdb_monitor_free_memory:=100}
: ${OCF_RESKEY_ctdb_start_as_disabled:=no}
: ${OCF_RESKEY_ctdb_tunables:=""}

: ${OCF_RESKEY_ctdb_config_dir:=/etc/ctdb}
: ${OCF_RESKEY_ctdb_binary:=/usr/bin/ctdb}
: ${OCF_RESKEY_ctdbd_binary:=/usr/sbin/ctdbd}
: ${OCF_RESKEY_ctdb_socket:=/var/lib/ctdb/ctdb.socket}
: ${OCF_RESKEY_ctdb_dbdir:=/var/lib/ctdb}
: ${OCF_RESKEY_ctdb_logfile:=/var/log/ctdb/log.ctdb}
: ${OCF_RESKEY_ctdb_debuglevel:=2}

: ${OCF_RESKEY_smb_conf:=/etc/samba/smb.conf}
: ${OCF_RESKEY_smb_passdb_backend:=tdbsam}
: ${OCF_RESKEY_smb_idmap_backend:=tdb2}

######################################################################
```

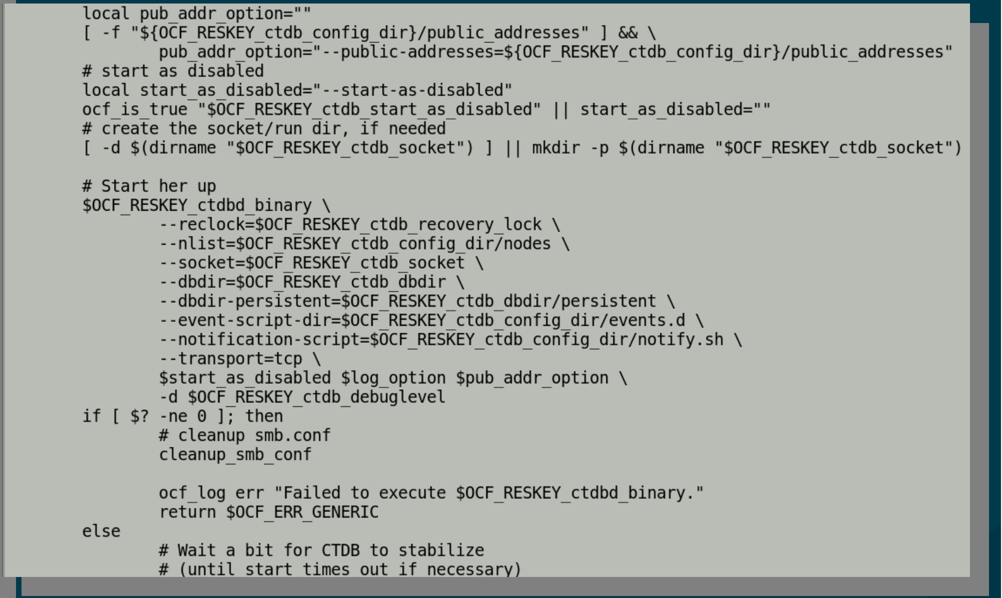
CTDB Resource Agent Samples

CHANGE

Introducing Pacemaker



```
ctdb_start() {
        # Do nothing if already running
        ctdb_monitor && return $OCF_SUCCESS

        # Make sure config is adequate
        ctdb_validate
        rv=$?
        [ $rv -ne 0 ] && return $rv

        # Die if databases are corrupted
        persistent_db_dir="${OCF_RESKEY_ctdb_dbdir}/persistent"
        mkdir -p $persistent_db_dir 2>/dev/null
        for pdbase in $(ls $persistent_db_dir/*.tdb.[0-9] 2>/dev/null$) ; do
                /usr/bin/tdbdump $pdbase >/dev/null 2>/dev/null || {
                        ocf_log err "Persistent database $pdbase is corrupted!  CTDB will not start."
                        return $OCF_ERR_GENERIC
                }
        done

        # Add necessary configuration to smb.conf
        init_smb_conf
        if [ $? -ne 0 ]; then
                ocf_log err "Failed to update $OCF_RESKEY_smb_conf."
                return $OCF_ERR_GENERIC
        fi

        # Generate new CTDB sysconfig
        generate_ctdb_sysconfig
        enable_event_scripts

        # Use logfile by default (and create the logdir if needed), or syslog if asked for
        local log_option
        if [ "$OCF_RESKEY_ctdb_logfile" = "syslog" ]; then
                log_option="--syslog"
        else
                log_option="--logfile=$OCF_RESKEY_ctdb_logfile"
                [ -d $(dirname "$OCF_RESKEY_ctdb_logfile") ] || \
                        mkdir -p $(dirname "$OCF_RESKEY_ctdb_logfile")
                [ -f "$OCF_RESKEY_ctdb_logfile" ] || \
                        touch "$OCF_RESKEY_ctdb_logfile"
        fi
        # public addresses file (should not be present, but need to set for correctness if it is)
```



```
        local pub_addr_option=""
        [ -f "${OCF_RESKEY_ctdb_config_dir}/public_addresses" ] && \
                pub_addr_option="--public-addresses=${OCF_RESKEY_ctdb_config_dir}/public_addresses"
        # start as disabled
        local start_as_disabled="--start-as-disabled"
        ocf_is_true "$OCF_RESKEY_ctdb_start_as_disabled" || start_as_disabled=""
        # create the socket/run dir, if needed
        [ -d $(dirname "$OCF_RESKEY_ctdb_socket") ] || mkdir -p $(dirname "$OCF_RESKEY_ctdb_socket")

        # Start her up
        $OCF_RESKEY_ctdbd_binary \
                --reclock=$OCF_RESKEY_ctdb_recovery_lock \
                --nlist=$OCF_RESKEY_ctdb_config_dir/nodes \
                --socket=$OCF_RESKEY_ctdb_socket \
                --dbdir=$OCF_RESKEY_ctdb_dbdir \
                --dbdir-persistent=$OCF_RESKEY_ctdb_dbdir/persistent \
                --event-script-dir=$OCF_RESKEY_ctdb_config_dir/events.d \
                --notification-script=$OCF_RESKEY_ctdb_config_dir/notify.sh \
                --transport=tcp \
                $start_as_disabled $log_option $pub_addr_option \
                -d $OCF_RESKEY_ctdb_debuglevel
        if [ $? -ne 0 ]; then
                # cleanup smb.conf
                cleanup_smb_conf

                ocf_log err "Failed to execute $OCF_RESKEY_ctdbd_binary."
                return $OCF_ERR_GENERIC
        else
                # Wait a bit for CTDB to stabilize
                # (until start times out if necessary)
```

CTDB Resource Agent Samples

CHANGE

Introducing Pacemaker

```
# CTDB
pcs resource create ctdb ocf:heartbeat:CTDB \
        params \
                ctdb_recovery_lock="/gluster/lock/lockfile" \
                ctdb_socket="/var/run/ctdb/ctdbd.socket" \
                ctdb_manages_winbind="no" \
                ctdb_manages_samba="no" \
                ctdb_logfile="/var/log/log.ctdb" \
        op monitor interval="10" timeout="30" \
        op start interval="0" timeout="90" \
        op stop interval="0" timeout="100" \
        --clone ctdb-clone ctdb meta interleave="true" globally-unique="false"

# CTDB: We need our shared recovery lock file
pcs constraint colocation add ctdb-clone with ctdb_lock-clone INFINITY
pcs constraint order ctdb_lock-clone then ctdb-clone INFINITY
```

CTDB Resource Definition

CHANGE

Introducing Pacemaker

So I said “easily” a few slides back… well, “easy” is relative.

CHANGE

Introducing Pacemaker

- Clusters are not simple things. Designing, configuring, and administering them does carry some complexity.
- Cluster Labs, the people behind Pacemaker, want to maintain a community of RAs that are as “dumb” and simple as possible.
  - Makes things a lot easier to debug and predict.
- Pacemaker also applies relatively simple logical rules and constraints to determine where, when, and how resources are managed.

CHANGE

Dialing back CTDB

CHANGE

Dialing back CTDB

Configuring CTDB so that it only serves as a distributed database backend provider is as simple as not telling it to do other things.

- Don't configure CTDB_PUBLIC_ADDRESSES
  - Disables VIP management
- Don't configure CTDB_MANAGES_SAMBA
  - Disables management of smbd and nmbd
- Don't configure CTDB_MANAGES_WINBIND
  - Disables management of winbindd

Hat tip: Michael Adam <obnox@samba.org>

CHANGE

Filling in the gaps

Now we need to find other resources to provide the features which we told CTDB not to provide.

CHANGE

Filling in the gaps

```
# Virtual IPs
pcs resource create vip${ipcount} ocf:heartbeat:IPaddr2 \
        params \
                ip=${ip} \
                flush_routes="true" \
        op monitor interval=60s \
        meta resource-stickiness="0"
```

VIP Management: IPaddr2

- One resource per address.
- Pacemaker moves the resource for failover.
- Only fails back if resource is not evenly distributed.

- Daemons are a grouped resource and cloned to all nodes.
- Colocate the group with a CTDB instance and start it after CTDB start.

```
# Samba
pcs resource create nmb lsb:nmb \
        op start timeout="60" interval="0" \
        op stop timeout="60" interval="0" \
        op monitor interval="60" timeout="60"
pcs resource create smb lsb:smb \
        op start timeout="60" interval="0" \
        op stop timeout="60" interval="0" \
        op monitor interval="60" timeout="60"
pcs resource group add samba-group nmb smb
pcs resource clone samba-group meta interleave="true"

pcs constraint colocation add samba-group-clone with ctdb-clone INFINITY
pcs constraint order ctdb-clone then samba-group-clone INFINITY
```

Daemon Management

CHANGE

Playing nice with others

Finally, we're ready to configure other resources, which can take advantage of Pacemaker's VIP and daemon management capabilities.

```
# Ganesha
pcs resource create ganesha ganesha \
        params \
                config="/etc/glusterfs-ganesha/nfs-ganesha.conf" \
        --clone ganesha-clone ganesha meta interleave="true" \
                                           globally-unique="false" \
                                           notify="true"

# Ganesha: We need our shared state FS
pcs constraint colocation add ganesha-clone with ganesha_state-clone INFINITY
pcs constraint order ganesha_state-clone then ganesha-clone INFINITY
```

Example: NFS-Ganesha

Intermezzo

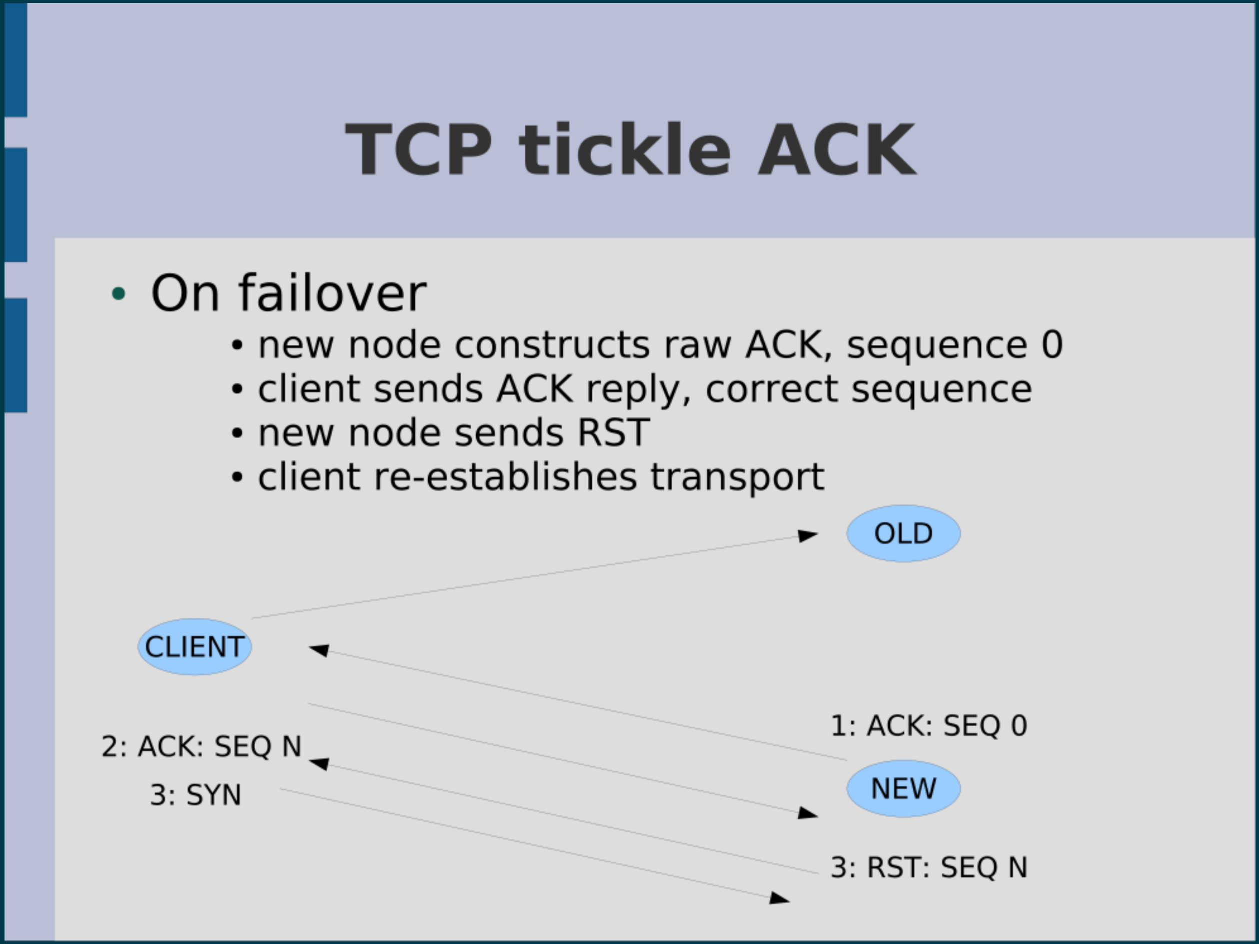
A SHORT DEMO...?

Act III. Al Futuro

LOOKING AHEAD

LOOKING AHEAD

Remember Tickle ACKs?



TCP tickle ACK

- On failover
  - new node constructs raw ACK, sequence 0
  - client sends ACK reply, correct sequence
  - new node sends RST
  - client re-establishes transport

OLD
CLIENT
1: ACK: SEQ 0
2: ACK: SEQ N
NEW
3: SYN
3: RST: SEQ N

Clustered NAS meets GPFS by tridge
(https://www.samba.org/~tridge/ctdb.pdf)

LOOKING AHEAD

Remember Tickle ACKs?

Tickle ACKs have been implemented in Pacemaker, as a feature of the portblock RA.

- The TCP sequence is executed correctly.
- Requires a user-specified directory to track active TCP connections.
  - Either shared directory or local directory synchronized via something like csync2
- Determines active TCP connections via periodic (default 10 seconds) calls to netstat.

LOOKING AHEAD

Remember Tickle ACKs?

Possibly better implemented using conntrackd?

- conntrack - stateful packet inspection tools for iptables.
- Instances can keep iptables state of other nodes.
  - You can filter which connections you want to track
  - This remote state can be then dumped into the local iptables
- Current synchronization mechanisms are “soft real-time” asynchronous replication protocols.
  - The various mechanisms provide different levels of trade-offs between reliable replication and bandwidth usage

LOOKING AHEAD

Planned enhancements

A few enhancements are already designed, awaiting implementation:

- portblock w/tickle ACKs
- Deterministic VIP failover and failback
  - Default method is not strictly deterministic
  - No failback by default
- Robust CLI and configuration
  - Add a layer of abstraction/simplification for common use cases

LOOKING AHEAD

What if...?

Longer-term:

- Manage storage volumes
  - At least monitor status
  - Possibly start/stop or mount/unmount
- Move new tickle ACK implementation into a different RA
  - A new tickle RA?
  - Maybe IPaddr2? A new IPaddr3?
- Remove the need for a shared filesystem from CTDB?
  - Unix DGRAM sockets?
- SMB3 Continuous Availability? :) :) (hi Team!)

Fine

(Das Ende)

THANK YOU!

https://github.com/jarrpa/storage-ha

jarrpa@samba.org || jarrpa@redhat.com
IRC: jarrpa in #samba-technical on irc.freenode.net
Twitter: @jarrpa