Clustered NAS meets GPFS

Andrew Tridgell
LTC ALRT Team

Scaling NAS

- What if?
    - you have 30,000 NAS users
    - you have 100 NAS servers
    - every day you run out of space on one of them
- What can you do?
    - Get a really big, all-active, clustered NAS box

SOFS
Scale Out File Servives

- Highly available, highly scalable NAS
    - built on top of a Linux cluster
    - uses IBMs GPFS cluster filesystem
    - highly available, fast automatic failover
    - very efficient CIFS clustering for Samba
    - all-active design – no waiting for failover node to kick in
    - scales to multiple petabytes of storage
    - fully protocol coherent for CIFS and NFS
    - also supports http, ftp serving

SOFS Components

- Hardware
  - HS21 blades
  - Dsxx SAN storage, FC connected
  - gigabit and/or infiniband
- Software
  - RHEL5 Linux on each node
  - GPFS 3.2 cluster filesystem
  - Samba 3.0, with clustering extensions
  - CTDB clustering suite
  - SOFS management GUI
  - winbind for Active Directory integration
- Protocols
  - CIFS, NFS, http, ftp
  - rr-DNS for load balancing

Clustering Samba

- Samba architecture
  - lots of small 'tdb' databases
  - each tdb holds meta-data for POSIX<->CIFS semantic maping
- Easy clustering?
  - just put the tdb files on GPFS?
  - much too slow!
- CTDB
  - 'clustered tdb', small distributed database
  - meta-data stored in memory on each node
  - scales well

CTDB features

- Database
  - simple database API
  - automatic recovery on cluster changes
- IP failover
  - handles public IP assignment, gratuituous ARP
  - tickle-ACKs for fast failover
- Protocol hooks
  - CTDB offers 'event scripts' for protocol exensions
  - handles NFS lock recovery

All-active NAS

- Active-passive?
  - the common solution for robust NAS in the past
  - a hot spare waits for a server to fail
  - on failure, STOMITH and take over role
  - admins pray that hot spare actually works
- All-active
  - All nodes in the cluster serve entire namespace all the time
  - when a node fails, all other nodes are already serving the same files
  - less reliance on divine intervention :-)

Scaling Results

- smbtorture NBENCH test
  - 32 clients
  - 1 to 4 nodes

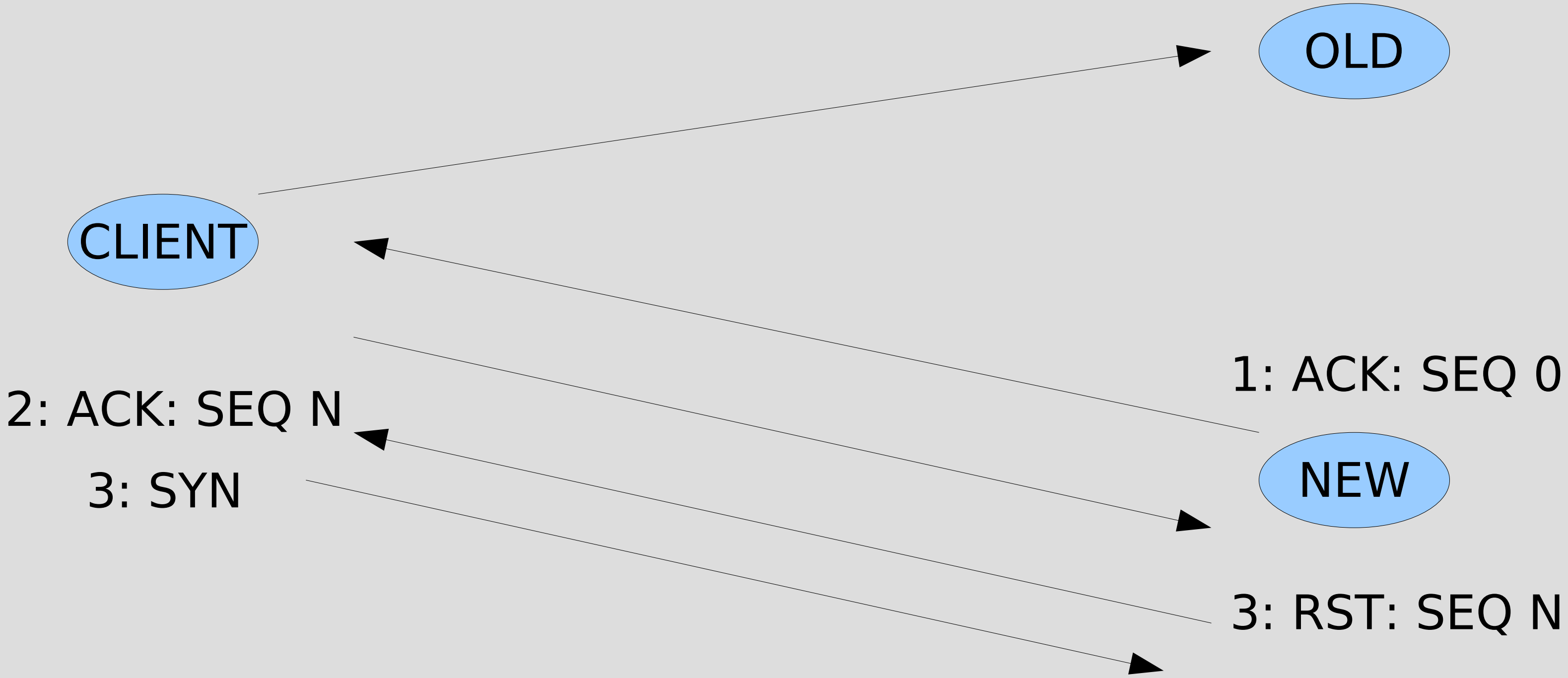
```
OLD (pre-CTDB) approach
1 node     95.0 Mbytes/sec
2 nodes     2.1 MBytes/sec
3 nodes     1.8 MBytes/sec
4 nodes     1.8 MBytes/sec

NEW (CTDB) approach
1 node      109 Mbytes/sec
2 nodes     210 MBytes/sec
3 nodes     278 MBytes/sec
4 nodes     308 MBytes/sec
```

Fast NAS failover

- Fast failover
  - winXP box copying files on NAS box
  - look at what node it is connected to
  - disable that node
  - copy continues after approx 1 sec pause
- How does it work?
  - usual IP takeover shenanigans (grat arp etc)
  - added magic is 'TCP tickle-ACK'

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



Using CTDB

```
Usage: ctdb [options] <control>
Options:
   -n <node>          choose node number, or 'all' (defaults to local node)
   -Y                 generate machinereadable output
   -t <timelimit>     set timelimit for control in seconds (default 3)
Controls:
  status                                       show node status
  ping                                         ping all nodes
  getvar          <name>                       get a tunable variable
  setvar          <name> <value>               set a tunable variable
  listvars                                     list tunable variables
  statistics                                   show statistics
  statisticsreset                              reset statistics
  ip                                           show which public ip's that ctdb manages
  process-exists  <pid>                        check if a process exists on a node
  getdbmap                                     show the database map
  catdb           <dbname>                     dump a database
  getmonmode                                   show monitoring mode
  setmonmode      <0|1>                        set monitoring mode
  setdebug        <debuglevel>                 set debug level
  getdebug                                     get debug level
  attach          <dbname>                     attach to a database
  dumpmemory                                   dump memory map to logs
  getpid                                       get ctdbd process ID
  disable                                      disable a nodes public IP
  enable                                       enable a nodes public IP
  ban             <bantime|0>                  ban a node from the cluster
  unban                                        unban a node from the cluster
  shutdown                                     shutdown ctdbd
  recover                                      force recovery
  freeze                                       freeze all databases
  thaw                                         thaw all databases
  isnotrecmaster                               check if the local node is recmaster or not
  killtcp         <srcip:port> <dstip:port>    kill a tcp connection.
  gratiousarp     <ip> <interface>             send a gratious arp
  tickle          <srcip:port> <dstip:port>    send a tcp tickle ack
  gettickles      <ip>                         get the list of tickles registered for this ip
  regsrvid        <pnn> <type> <id>            register a server id
  unregsrvid      <pnn> <type> <id>            unregister a server id
  chksrvid        <pnn> <type> <id>            check if a server id exists
  getsrvids                                    get a list of all server ids
```

SOFS databases

- SOFS uses 9 CTDB databases
  - 4 persistent, 5 temporary
  - maps Windows/CIFS semantics to POSIX

```
[root@fscc-hs21-12 ~]# ctdb getdbmap
Number of databases:9
dbid:0x435d3410 name:notify.tdb path:/var/ctdb/notify.tdb.0
dbid:0x42fe72c5 name:locking.tdb path:/var/ctdb/locking.tdb.0
dbid:0x1421fb78 name:brlock.tdb path:/var/ctdb/brlock.tdb.0
dbid:0x17055d90 name:connections.tdb path:/var/ctdb/connections.tdb.0
dbid:0xc0bdde6a name:sessionid.tdb path:/var/ctdb/sessionid.tdb.0
dbid:0x7bbbd26c name:passdb.tdb path:/var/ctdb/persistent/passdb.tdb.0 PERSISTENT
dbid:0xb775fff6 name:secrets.tdb path:/var/ctdb/persistent/secrets.tdb.0 PERSISTENT
dbid:0xe98e08b6 name:group_mapping.tdb path:/var/ctdb/persistent/group_mapping.tdb.0 PERSISTENT
dbid:0x2672a57f name:idmap2.tdb path:/var/ctdb/persistent/idmap2.tdb.0 PERSISTENT
```

CTDB Tunables

- Lots of tunables
  - rarely need to be modified

```
[root@fscc-hs21-12 ~]# ctdb listvars
MaxRedirectCount    = 3
SeqnumFrequency     = 1
ControlTimeout      = 60
TraverseTimeout     = 20
KeepaliveInterval   = 2
KeepaliveLimit      = 5
MaxLACount          = 7
RecoverTimeout      = 5
RecoverInterval     = 1
ElectionTimeout     = 3
TakeoverTimeout     = 5
MonitorInterval     = 15
MonitorRetry        = 5
TickleUpdateInterval = 20
EventScriptTimeout  = 20
RecoveryGracePeriod = 60
RecoveryBanPeriod   = 300
DatabaseHashSize    = 10000
RerecoveryTimeout   = 10
EnableBans          = 1
DeterministicIPs    = 1
```

Status Monitoring

- 'ctdb status'
  - shows state of each node
  - most commonly used ctdb command

```
[root@fscc-hs21-12 ~]# ctdb status
Number of nodes:4
pnn:0 9.155.61.96      OK (THIS NODE)
pnn:1 9.155.61.97      OK
pnn:2 9.155.61.98      BANNED
pnn:3 9.155.61.99      OK
Generation:159484266
Size:4
hash:0 lmaster:0
hash:1 lmaster:1
hash:2 lmaster:2
hash:3 lmaster:3
Recovery mode:NORMAL (0)
Recovery master:1
```

Public IPs

- IP Failover
    - 'HEALTHY' nodes get public IPs
    - these IPs are setup in rr-DNS

```
[root@fscc-hs21-12 ~]# ctdb ip
Public IPs on node 0
10.13.26.1 0
10.13.26.2 1
10.13.26.3 2
10.13.26.4 3
10.13.26.5 0
10.13.26.6 1
```

Demo!

- Some flash movies available
  - http://samba.org/~tridge/ctdb_movies

Questions?

- For more information on CTDB see

  http://ctdb.samba.org/