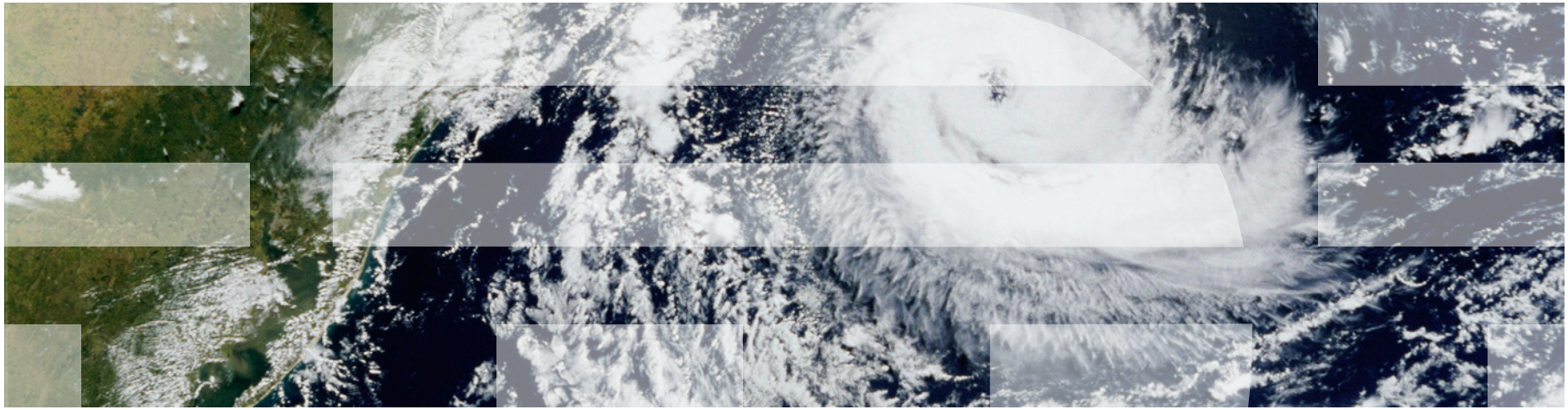
Spectrum Scale CCR Internals

(CCR - Clustered Configuration Repository)

Ralf Eberhard



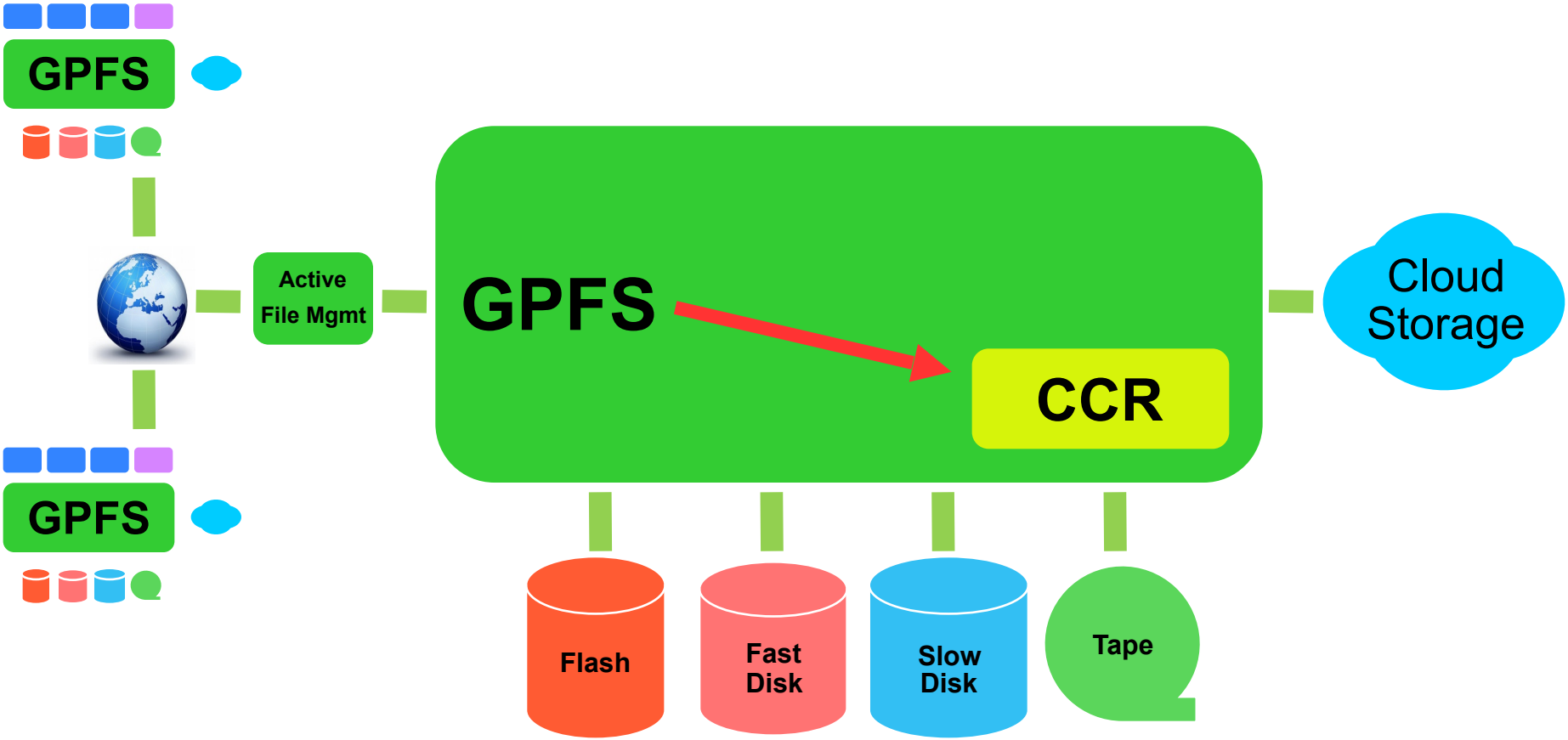
Agenda

- What is the CCR?
- CCR system context
- Outline of the CCR
- History of the CCR
- Paxos algorithm inside the CCR
- CCR current use cases
- Useful CCR commands (including examples)
- Recover single quorum node
- Limitations
- Known issues
- Questions & Answers

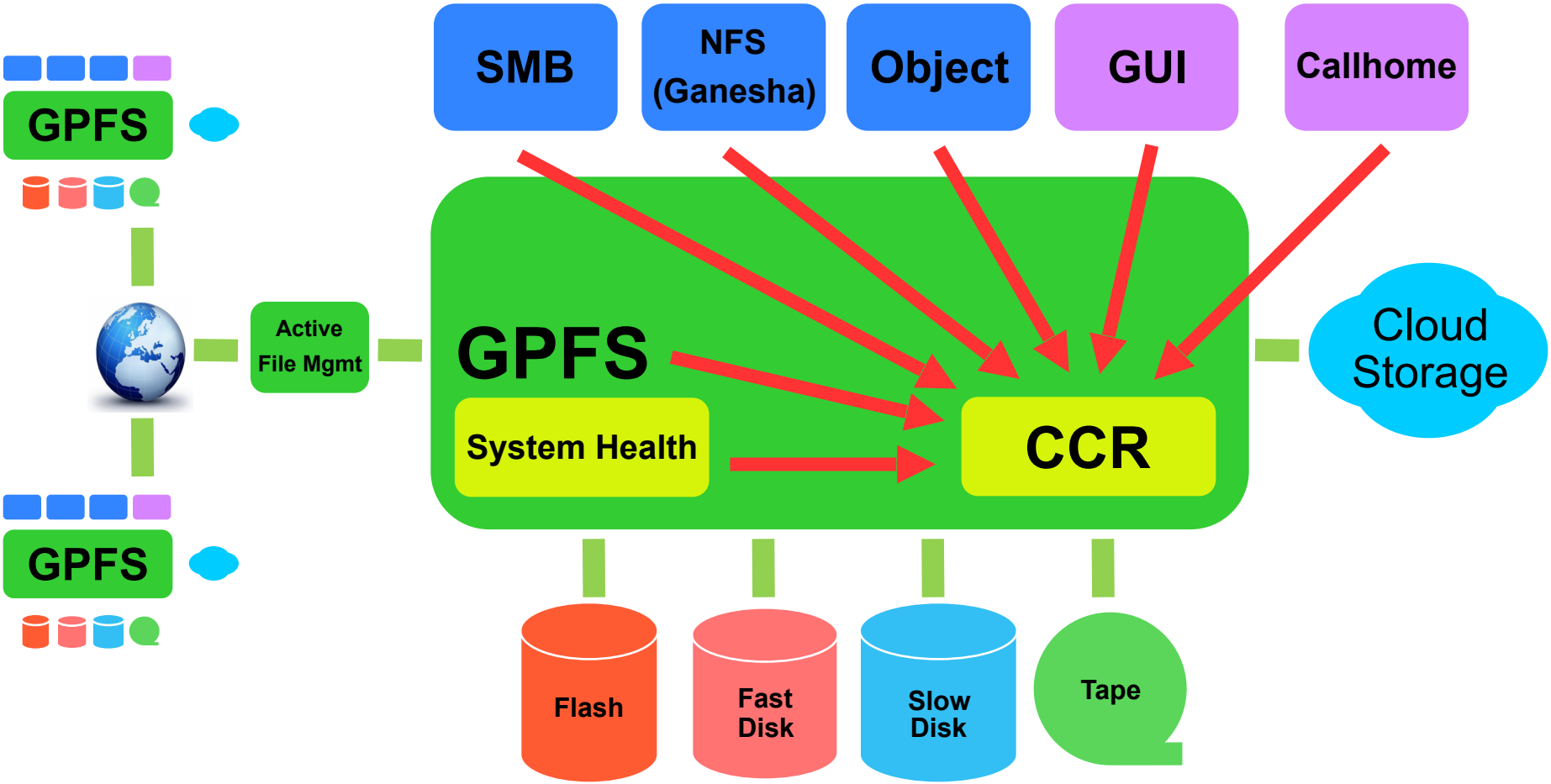
What is the CCR (Clustered Configuration Repository)?

- New configuration store for Spectrum Scale, replacing old server-based mechanism
- Stores **committed files** and flat **key-value pairs** on all quorum nodes based on Paxos algorithm
- Drawbacks of old server-based configuration mechanism:
  - **Single point of failure for updates**
  - **No dynamic tiebreaker change (needs GPFS down, to enable/disable tiebreaker disks)**
- CCR Requirements:
  - Fault tolerant
  - No more single point of failure for updates, due to multiple quorum nodes
  - **Adjust tiebreaker disks (quorum) dynamically**
  - Usable by **other services (like file protocols, GUI, System Health…)**
  - **Cluster manager election**
- **CCR is an internal component only; not meant to be used by the customer**

Spectrum Scale – System context – CCR – 4.1.0



Spectrum Scale – System context – CCR – today



Outline of the CCR

- **Client-Server** architecture (*mmccr*: client, inside *mmfsd* and *mmsdrserv*: server)
- CCR Server **active** only **on quorum nodes**
- **Two types of objects** can be committed to the CCR:
  - Flat key-value pairs (vput, vget, vdel, vlist)
  - Files (e.g. mmsdrfs, fput, fget, fdel, flist)
- Interface:
  - **setup, init** to prepare the CCR
  - **vget, vput, vdel, vlist** to retrieve, store, delete and list key-value pairs
  - **fget, fput, fdel, flist** to retrieve, store, delete and list files (no file content)
  - **chnodes, chdisks** to change the quorum node and shared disk configuration
  - Get and put operations supporting atomic operations too („conditional get/put)
- **Tiebreaker disk changes during GPFS runtime**
- **CCR configuration** stored in flat files in CCR and in Paxos state (compressed)
- CCR needs **GSKit** (SSL authentication)
- **No external** GPFS command (no man-page)

CCR – History

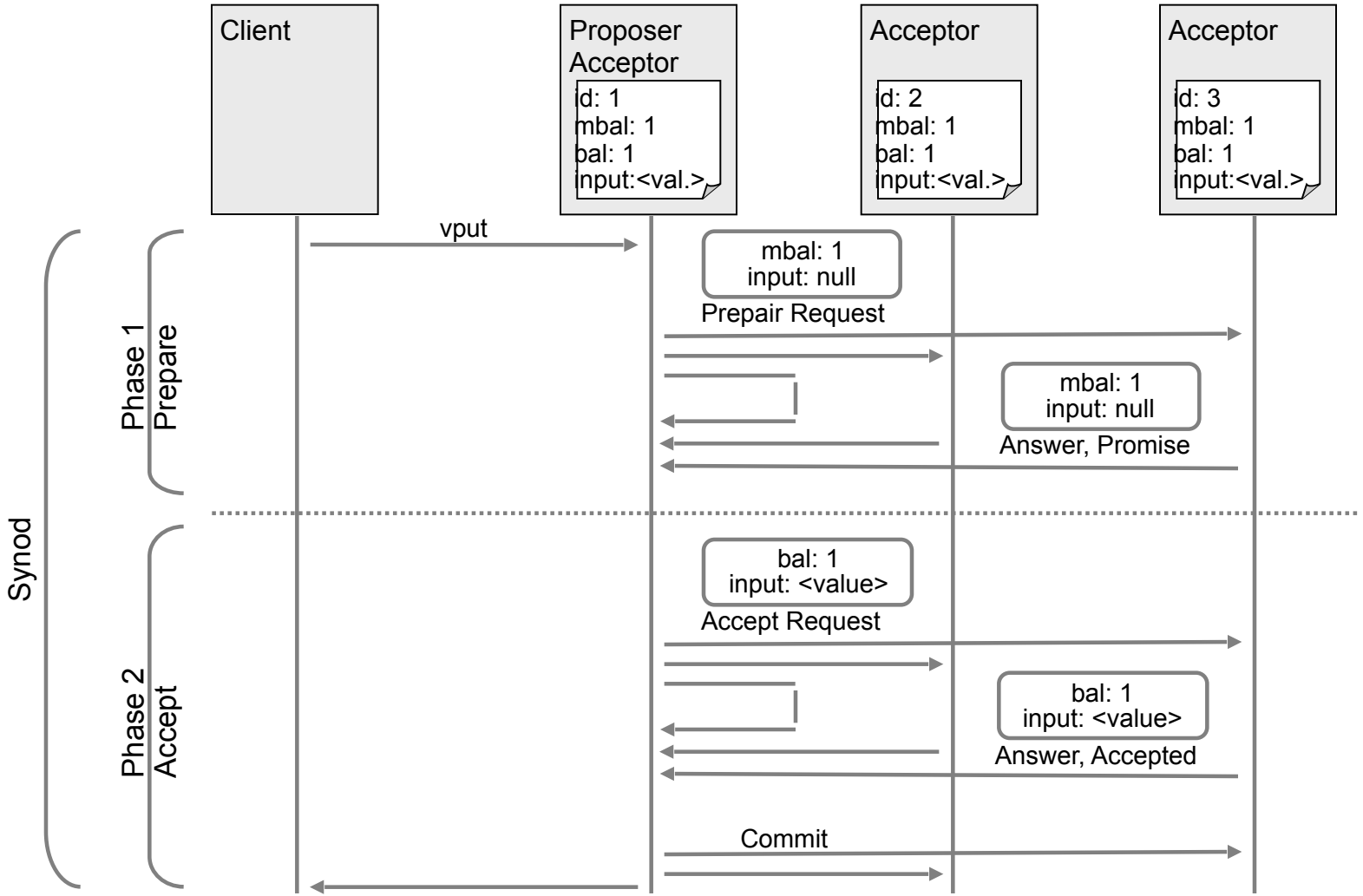
- CCR **introduced** in 4.1.0
- CCR **enhancements**, until 5.0.x:
  - Cached connections 4.1.1
  - Many RAS, FDTC changes (e.g. filtered trace entries, improved/added trace messages, …) → 4.1.0+
  - Parallel IO → 4.2.1
  - Cluster Disaster Recovery (3 of 4 use cases done) → 4.1.1+
  - Check/status functionality → 4.2.0
  - Optimisation's based on PMR's and defects → 4.1.0+
  - Last Disaster Recovery case (CCR not available and no backup available), planned for future release
  - All use cases, that needs a CCR disable today (e.g. changing hostname/mgmt IP address of quorum nodes → workaround available), planned for future release

CCR - Paxos algorithm

- Developed by **Leslie Lamport** (Microsoft Research), also known as the “La” in LaTeX
- **Fault tolerant „replicated state machine” by providing means for a set of nodes (the quorum nodes) to agree on a sequence of updates (the inputs) to some state maintained redundantly by each quorum node (a consensus algorithm)**
- The „replicated state machine“ is called **„paxos state“**
- “Used” by Google, SVC, Microsoft, XtreemFS and some others

Paxos Algorithm sequence inside CCR

What happens during a key-value put (CCR command: vput)?



Spectrum Scale – CCR – current use cases by other components

- Distribute configuration files cluster wide
- Cluster wide lock mechanism (CES and classical GPFS mm-commands)
- Cluster manager election
- Cluster manager challenges written to tiebreaker disk and answered via TB disk
- Progress information
- State machine information (e.g. CES IP assignments -> *cesiplist* file)
- GUI HA

Spectrum Scale – CCR – useful commands

- **CCR client side**:
  - Following commands don't change CCR state
  - *CCR_DEBUG=9 mmccr …*
  - *mmccr echo -n <COMMA_SEPARATED_NODE_LIST>*
  - *mmccr check* command (-Y → machine parsable, -e → extended checks)
- **CCR server side:**
  - GPFS logs and particular traces very helpful (trace level 4 default)
  - *mmsdrserv*: passive daemon, trace file: */var/adm/ras/mmsdrserv.log* contains just errors
  - For *mmsdrserv*: *mmccr msglevel -n <COMMA_SEPARATED_NODE_LIST> MSGLEVEL*
  - *mmccr dump* command (included in *mmfsadm dump all*): Prints CCR's state in memory to stdout (very helpful for CCR developer)
- **CCR general:**
  - *mmchcluster --ccr-disable* seems to be a "**golden hammer**". Be **careful** with it: Hides real CCR issues and other components (like CES, GUI …) need the CCR (configuration lost)
  - *mmsdrbackup* callback script creates CCR backup file (restoring with *mmsdrrestore -A* …)

Spectrum Scale – CCR – useful commands – examples

- **CCR_DEBUG and echo**:

```
[root@node-11 ~]# CCR_DEBUG=9 mmccr echo -n node-11,node-12,node-13,node-14,node-15
Using /var/mmfs/ccr
Size of file '/var/mmfs/ccr/ccr.nodes': 114 bytes
readNodeList('/var/mmfs/ccr/ccr.nodes') size 3 nErr 0
ccrNodeId: 1
readDiskList('/var/mmfs/ccr/ccr.disks') size 0 nErr 0
ccrio init: ip=10.0.100.11 port=1191 node=1 (0)
setCcrSecurity: ccrSecEnabled=1
Auth: secReady 1 cipherList 'AUTHONLY' keyGenNumber 2
parseNodeList: node-11 -> id 1 host node-11 port 1191 ipaddr 10.0.100.11
parseNodeList: node-12 -> id 2 host node-12 port 1191 ipaddr 10.0.100.12
parseNodeList: node-13 -> id 3 host node-13 port 1191 ipaddr 10.0.100.13
parseNodeList: node-14 -> id -1 host node-14 port 1191 ipaddr 10.0.100.14
parseNodeList: node-15 -> id -1 host node-15 port 1191 ipaddr 10.0.100.15
connecting to node 10.0.100.11:1191 (timeout -1, handshaketimeout -1)
No cached out-connection found for address: 10.0.100.11:1191 (0)
connected to node 10.0.100.11:1191 (sock 4-0x7f5532341ac0)
sending msg 2 'debug' len 6 (sock 4)
sent msg 2 'debug' (sock 4) ok
receiving from 10.0.100.11:1191 (sock 4) 0x7F5532341AC0
waiting for hdr (len 8)
waiting for data (len 4)
received msg 0 'ok' len 4 (sock 4)
closing connection to 10.0.100.11:1191 (sock 4-0x7f5532341ac0)
closeSocket: shutdown socket 4 successful
closeSocket: closed socket 4 successful
debug response: err 0 type 0 (ok) len 4
echo
...

[root@node-11 ~]# tail /var/adm/ras/mmfs.log.latest
...
2017-03-30_07:56:46.290+0200: [N] echo
```

Spectrum Scale – CCR – useful commands – examples

- **check command (-Y → machine parsable, -e → extended checks)**:

CCR client node:

[root@node-15 ~]# mmccr check -Y -e
mmccr::HEADER:version:reserved:reserved:NodeId:CheckMnemonic:ErrorCode:ErrorMsg:ListOfFailedEntities:ListOfSucceedEntities:Severity:
mmccr::0:1:::-1:CCR_CLIENT_INIT:0:::/var/mmfs/ccr,/var/mmfs/ccr/committed,/var/mmfs/ccr/ccr.nodes,Security:OK:
mmccr::0:1:::-1:FC_CCR_AUTH_KEYS:0:::/var/mmfs/ssl/authorized_ccr_keys:OK:
mmccr::0:1:::-1:FC_CCR_PAXOS_CACHED:0:::/var/mmfs/ccr/cached,/var/mmfs/ccr/cached/ccr.paxos:OK:
mmccr::0:1:::-1:PC_QUORUM_NODES:0:::10.0.100.11,10.0.100.12,10.0.100.13:OK:

CCR quorum node:

[root@node-11 ~]# mmccr check -Y
mmccr::HEADER:version:reserved:reserved:NodeId:CheckMnemonic:ErrorCode:ErrorMsg:ListOfFailedEntities:ListOfSucceedEntities:Severity:
mmccr::0:1:::1:CCR_CLIENT_INIT:0:::/var/mmfs/ccr,/var/mmfs/ccr/committed,/var/mmfs/ccr/ccr.nodes:OK:
mmccr::0:1:::1:FC_CCR_AUTH_KEYS:0:::/var/mmfs/ssl/authorized_ccr_keys:OK:
mmccr::0:1:::1:FC_CCR_PAXOS_CACHED:0:::/var/mmfs/ccr/cached,/var/mmfs/ccr/cached/ccr.paxos:OK:
mmccr::0:1:::1:FC_CCR_PAXOS_12:0:::/var/mmfs/ccr/ccr.paxos.1,/var/mmfs/ccr/ccr.paxos.2:OK:

[root@node-11 ~]# mmccr check -Y -e
mmccr::HEADER:version:reserved:reserved:NodeId:CheckMnemonic:ErrorCode:ErrorMsg:ListOfFailedEntities:ListOfSucceedEntities:Severity:
mmccr::0:1:::1:CCR_CLIENT_INIT:0:::/var/mmfs/ccr,/var/mmfs/ccr/committed,/var/mmfs/ccr/ccr.nodes,Security:OK:
mmccr::0:1:::1:FC_CCR_AUTH_KEYS:0:::/var/mmfs/ssl/authorized_ccr_keys:OK:
mmccr::0:1:::1:FC_CCR_PAXOS_CACHED:0:::/var/mmfs/ccr/cached,/var/mmfs/ccr/cached/ccr.paxos:OK:
mmccr::0:1:::1:FC_CCR_PAXOS_12:0:::/var/mmfs/ccr/ccr.paxos.1,/var/mmfs/ccr/ccr.paxos.2:OK:
mmccr::0:1:::1:PC_LOCAL_SERVER:0:::node-11.localnet.com:OK:
mmccr::0:1:::1:PC_IP_ADDR_LOOKUP:0:::node-11.localnet.com,0.000:OK:
mmccr::0:1:::1:PC_QUORUM_NODES:0:::10.0.100.11,10.0.100.12,10.0.100.13:OK:
mmccr::0:1:::1:FC_COMMITTED_DIR:0::0:8:OK:
mmccr::0:1:::1:TC_TIEBREAKER_DISKS:0::::OK:

Spectrum Scale – CCR – useful commands – examples

- **dump command:**

```
[root@node-11 ~]# mmccr dump
CCR message level 1 trace level 4
CCR server: workerRunning 1 started 1 quit 0 opQueue 0
  minReleaseLevel 1700
  auth: secReady 1 cipherList 'AUTHONLY' keyGenNumber 2
Cached out-connections: 2
  stats: nAddGood 42 nAddBad 0 nGetGood 40 nGetBad 0 nGetNotFound 2
  1: connection 0x7f159c000960 addr 10.0.100.12:1191 sock 15 authenticated 1
  2: connection 0x7f1590000960 addr 10.0.100.13:1191 sock 16 authenticated 1
...
CCR state:
  ccrNodeId 1 ccrAddr 10.0.100.11:1191 ccrClusterId 317908494283594829
  nodes: 3
    (1, 'node-11.localnet.com', ('10.0.100.11', 1191))
    (2, 'node-12.localnet.com', ('10.0.100.12', 1191))
    (3, 'node-13.localnet.com', ('10.0.100.13', 1191))
PaxosServer state:
  nodeId 1, mySlot 0, mySlotVersion 0, epoch 3 newEpochNeeded 1, slotAssignmentVersion 0
  lastReadTime 1490802736 (51072 sec ago) slotChangeTime 1490799565 (54243 sec ago) challengePending 0
  lastSrvRefresh attempt 0 success 1490802736 pending ballot seq 0 start 0
  dblk: seq 87 mbal (0.0) bal (0.0) challenge 0 inp ((n0,e0),0):(none):-1:None
  commit: <config ver 0: <[(N1,S0,V0,L1), (N2,S1,V0,L1), (N3,S2,V0,L1)] [] min=1> leader=1 lease=23333 ver=4 horiz=-1 updates={(n1,e0): 4, (n1,e1): 63, (n1,e2): 80, (n1,e3): 86, (n2,e0): 34, (n2,e1): 75} vals={'foo1': (10, 'foo1value 2'), 'mmRunningCommand': (3, '')} vdel=8 files={1: (v1, ((n1,e0),0), e7e9c9f0), 2: (v1, ((n1,e0),1), ffffffff), 3: (v1, ((n1,e0),2), ffffffff), 4: (v1, ((n1,e0),3), 81dcc91), 5: (v2, ((n1,e1),e), 5f92abf0), 6: (va, ((n2,e1),26), 2de12e53), 7: (v1, ((n2,e0),21), cee097c7), 8: (v2c, ((n1,e3),56), 235e835c)} fdel=0>
  filesSyncSeq: 87
...
I/O config <[(N1,S0,V0,L1), (N2,S1,V0,L1), (N3,S2,V0,L1)] [] min=1>
  myNodeId 1 mySlot 0 mySlotVersion 0 active 1 listener 1
   0 good: file '/var/mmfs/ccr/ccr.paxos.[12]' cached 1 nextToWriteIdx 0 bR: 4096 bW: 4096
        epoch=3 saVer=0 summary=<seq=87 mbal=(0.0) bal=(0.0) inp=((n0,e0),0):(none):-1:None chal=0 committed=<config ver 0: <[(N1,S0,V0,L1), (N2,S1,V0,L1), (N3,S2,V0,L1)] [] min=1> leader=1 lease=23333 ver=4 horiz=-1 updates={(n1,e0): 4, (n1,e1): 63, (n1,e2): 80, (n1,e3): 86, (n2,e0): 34, (n2,e1): 75} vals={'foo1': (10, 'foo1value 2'), 'mmRunningCommand': (3, '')} vdel=8 files={1: (v1, ((n1,e0),0), e7e9c9f0), 2: (v1, ((n1,e0),1), ffffffff), 3: (v1, ((n1,e0),2), ffffffff), 4: (v1, ((n1,e0),3), 81dcc91), 5: (v2, ((n1,e1),e), 5f92abf0), 6: (va, ((n2,e1),26), 2de12e53), 7: (v1, ((n2,e0),21), cee097c7), 8: (v2c, ((n1,e3),56), 235e835c)} fdel=0>>
   1 good: remote file at owner 2
   2 good: remote file at owner 3
...
  Paxos IO collector IC: 0
Stats:
  vgets 6 vputs 2 vdels 0 fgets 3 fputs 2 fdels 0 configs 0
  synods 4 (noop 0 forgn 0 retry 0 stale 0) ballots 4 comWtOk 0 comWtTO 0
  fxfers 0 backups 0 purged op-queue reqs: 0
Active trace filter: 0
```

Spectrum Scale – CCR – recover one single quorum node (3 quorum node cluster)

```
[root@node-11 ~]# mmlscluster

GPFS cluster information
========================
  GPFS cluster name:         gpfs-cluster-1.localnet.com
  GPFS cluster id:           317908494283594829
  GPFS UID domain:           localnet.com
  Remote shell command:      /usr/bin/ssh
  Remote file copy command:  /usr/bin/scp
  Repository type:           CCR

GPFS cluster configuration servers:
-----------------------------------
  Primary server:    node-11.localnet.com (not in use)
  Secondary server:  (none)

 Node  Daemon node name      IP address   Admin node name       Designation
--------------------------------------------------------------------------------
   1   node-11.localnet.com  10.0.100.11  node-11.localnet.com  quorum
   2   node-12.localnet.com  10.0.100.12  node-12.localnet.com  quorum
   3   node-13.localnet.com  10.0.100.13  node-13.localnet.com  quorum
   4   node-14.localnet.com  10.0.100.14  node-14.localnet.com
   5   node-15.localnet.com  10.0.100.15  node-15.localnet.com

[root@node-12 ~]# mmgetstate -a
mmgetstate: This node does not belong to a GPFS cluster.
mmgetstate: Command failed. Examine previous error messages to determine cause.

[root@node-12 ~]# mmsdrrestore -p node-11
Thu Mar 30 08:54:26 CEST 2017: mmsdrrestore: Processing node node-12.localnet.com
genkeyData1                                   100% 3531     3.5KB/s   00:00
genkeyData2                                   100% 3531     3.5KB/s   00:00
mmsdrrestore: Node node-12.localnet.com successfully restored.

[root@node-12 ~]# mmgetstate -a

 Node number  Node name        GPFS state
------------------------------------------
       1      node-11          active
       2      node-12          active
       3      node-13          active
       4      node-14          active
       5      node-15          active
```

Spectrum Scale – CCR – Limitations

- Committed **files stored just on quorum nodes, not on tiebreaker disks** (just file metadata)
  - Limitation for clusters with just two quorum nodes and one (or more) tiebreaker disk
  - File update succeeds if a majority of TB disks and one quorum node (e.g. the first) participates to the synod
  - Possible, that the second quorum node didn't get this file update, e.g. node is upgraded
  - When the first node gets down, while the second one is not available, the CCR on the second node returns errors during GPFS startup, because it reads the most recent Paxos state from the tiebreaker disk and tries to pull the updated file from the first (not available) quorum node
  - Solution: Upgrade (quorum) node by node and move the cluster manager node role to the node NOT being upgraded BEFORE you upgrade the second quorum node. Startup the upgraded node including GPFS while the first one is still active.
- During GPFS autoload=yes **CCR quorum** must be fulfilled to get a consistent configuration, e.g. a majority of quorum nodes must be started within a short time period, after that *mmstartup* must be invoked by hand. server-based: Nodes keep in arbitrating until primary and secondary configuration server are up und running.
- Changing node hostname/mgmt IP address needs *--ccr-disable.* Disable the CCR deletes the entire configuration files in the CCR, hence CES must be disabled before CCR disable. Workaround for this: Temporarily change the quorum node.

Spectrum Scale – CCR – Known issues

- For hard-powered off VM's and/or nodes: Files in the CCR committed directory truncated to zero-length:
  - fsync after file data write returns zero, but file data not written to disk (results in zero-length files, when the file has been written short time before the hard-power off happens)
  - In case enough good copies available (enough other quorum nodes): auto recovery, e.g. good copy from another quorum node
  - In case of just one quorum node (e.g. single node cluster) or too many truncated copies: Recovery procedure will be available soon to bring back this cluster into a working state

Spectrum Scale – CCR – Q&A

Thank you for your attention!