APPC Resiliency
An ODL Cluster-based Approach

Sharon Chisholm
APPC Non-Clustered Deployment

- VM1
  - APPC (ODL Framework & APPC OSGU Bundles)

- VM2
  - MySQL DB

- VM3
  - Node Red/DBuilder
APPC Clustered Deployment

VM1
- APPC (ODL Framework & APPC OSGU Bundles)

VM4
- MySQL DB

VM6
- Node Red/DBuilder

Load-balancer

MySQL DB

Node Red/DBuilder

APPC

ODL Framework & APPC OSGU Bundles

Load-balancer
ODL Clustering


https://wiki.opendaylight.org/view/OpenDaylight_Controller/Clustering

Summary

• Each cluster has members
• One is elected leader
• Only leader can write
• Cluster is functional if \( \frac{N}{2} + 1 \) out of \( N \) cluster members are available
• Otherwise, reports isolated leader status and writes may start failing

Used in conjunction with a loader-balancer to maximize availability

For geo-redundancy

• members in backup site should be given non-voting status, so as to not impact latency in primary site members.
Findings/Recommendations

- Enhance APP-C so that requests can be retried
- No getting stuck in weird states on failure
- Make requests idempotent
- Enhance Dispatcher so single failure won’t result in so many failures
- Closed loop action in isolated leader
- Alarm?
- Cluster-aware OAM
- Start/stop/status/health check of single or all APPC members
- Make default deployment
- Document?
Backup
Notes on ODL Clustering

• OpenDaylight Cluster does not provide load balancing of the incoming requests, but one can be set up to actually distributes the load of incoming request among cluster instances.

• Cluster Members
  • We should have minimum 3 nodes
  • OpenDaylight requires majority of nodes to be up to work as cluster. With 2 nodes if any 1 node fails, the other 1 cannot be a majority of 2 nodes. In 3 node cluster if 2 nodes fail, it cannot work as cluster. APP-C can still accept the request and process it, but it cannot write data on MD-SAL store.
  • Every node needs an identifier. OpenDaylight uses nodes’ role for this purpose.
  • Seed Nodes are nodes who will work together as cluster. When cluster nodes come up they talk to each other and then elect a leader node.
Notes on ODL Clustering

• Data shards
• used to contain all or certain segment of OpenDaylight’s MD-SAL data store. If we do not specify a module in the modules.conf file and do not specify a shard in module-shards.conf, then (by default) all the data is placed in the default shard (which must also be defined in module-shards.conf file). Each shard has replicas configured. We can specify the details of where the replicas reside in module-shards.conf file.

• Voting Status (Primary/Secondary)
• Primary/Secondary status useful for geo-redundancy
• With APIs provided by cluster-admin module of OpenDaylight, we can modify voting state of cluster nodes. All the voting nodes are considered as primary nodes, and non-voting nodes are considered secondary nodes. OpenDaylight uses “Strong Consistency” for transactions among primary nodes but uses “Eventual Consistency” for secondary nodes.