Understand the Algorithm to Determine Mastership In UCCX Failover Scenarios

Contents

Introduction

This document describes the algorithm used to determine mastership after the failover or recovery from island mode is initiated in Unified Contact Center Express (UCCX).

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- UCCX
- Failover Mechanism

Components Used

This document is not restricted to specific software and hardware versions.

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

Background Information

Two scenarios where a master must be elected are possible and the algorithm used to determine mastership between the two nodes is different for each scenario.

Normal Engine Failover

This scenario is encountered when there is either no master (such as when a cluster is started) or there is one master, when nodes are failover over in a HAoLAN or HAoWAN environment without

island mode.

The algorithm waterfalls to determine the mastership (i.e Try 1, else Try 2 else Try 3, else Try 4 in cases of contention):

Step 1. Determine the UCCX Engine status of both nodes, whichever is in better status – that is the new master. If both are alike, then move to step 2.

Step 2. Determine the Hardware model of the two nodes. The better hardware is the new master. If both are alike, move to step . Since many UCCX installations are now virtual, this step is not often used.

Step 3. Determine the Node 1 i.e Publisher (the first UCCX node installed). The new master is the Publisher node. CVD is programmed to make Node 1 as the Default master. This is taken from the PrimaryEngineComputerName parameter in the cluster config (ClusterSpecificConfig) on CET. If this value is incorrect, Node2 always take mastership. Refer : CSCuw95068.

Step 4. If Step 3 cannot determine the correct hostname of Publisher, make Node 2 as the master (Subscriber).

The logic is:

Step 1**.** Check the service status of the nodes. If Node 1 is IN_SERVICE and node 2 is in PARTIAL_SERVICE, Node 1 becomes the master. If the states are the same (IN_SERVICE or PARTIAL_SERVICE), go to STEP 2.

Step 2**.** The hardware specification of the 2 UCCX nodes is checked. The server with the better specification is handed the mastership. If the hardware specifications are the same, go to STEP 3.

Step 3**.** The PUBLISHER becomes the MASTER if the hostname of the PUBLISHER matches the PrimaryEngineComputerName on CET (ClusterSpecificConfig). If there is no MATCH go to step 4.

Step 4**.** Make Subscriber master if above step fails.

Island Mode Recovery

This scenario is encountered during recovery from island mode when there are two masters. When this occurs, the above algorithm is not executed. Rather, the UCCX Publisher node (the first UCCX node installed) retains mastership and the subscriber drops mastership.

> **Note**: An important thing to note is that the hostname of the primary node must match the PrimaryEngineComputerNam entry in the ClusterSpecificConfig object. Otherwise, the secondary node is chosen as the master. Use the CET tool to connect to the primary node to check whether the entry is correct and to change it if necessary.

Additionally, when the system checks which node is in better service status, as mentioned in step 1, this is the way to check the specific services checked

- Engine Service

- Manager Manager component within Engine

If both of these services are IN_SERVICE, then this node is considered for mastership.

This is a snippet from the logs where the algorithm is used to explain the scenario: In this scenario Node 1 was told to be the master before WAN outage; and when the WAN came back, Node 2 became the MASTER.

When the WAN link went down:

First, both the nodes were Master. Node 1 was the Master; Node 2 also became the master:

Cisco Unified CCX Engine on node 2 change master from false to true

```
3162: Dec 15 12:41:17.607 IST %MCVD-CLUSTER_MGR-7-UNK:JavaService167
```

This is also the time when the Node suspects a crash of the other node:

```
3111: Dec 15 12:41:17.481 IST %MCVD-CVD-4-HEARTBEAT_SUSPECT_NODE_CRASH:CVD suspects node crash:
state=Heartbeat State,nodeInfo=Node id=1 ip=172.30.72.2 convId=69 cmd=16 viewLen=1,dt=1022
```

When the WAN link came back and CONVERGENCE started:

```
9777: Dec 15 12:42:28.859 IST %MCVD-CVD-4-MASTER_DETECTS_NODE_JOIN:More than one master
detected, when processing node join: name=Cisco Unified CCX Database,nodeId=2,masterCnt=1
9778: Dec 15 12:42:28.859 IST %MCVD-CVD-7-UNK:Split after network partition is detected, new
nodeId=2

Node id=002, addresses=[172.30.83.2], MAC addresses=[279f2d5ba86d], compName=UCCXSUB, state=IN
SERVICE, en=true, rmiPort=6999, masterPort=1994
       VersionInfo: [ Version=8.5.1.11003-32, crsRelease=8.5.1.11003-32, crsServiceRelease=,
crsEngineeringSpecial=, dbEdition=IDS, dbVersion=V11, installTime=1348139852000,
upgradeTime=1348139852000, jtapiClientVersion=8.6(2.10000)-2 ]
       cT=969, uT=969, rT=528, serVer=3, cvdVer=3, points=0
  Component201: type=CRS Historical Datastore, state=IN SERVICE, en=true, prim=false, node=002,
activationTime=1348141153000, parent=null, uT=492, rT=193, rootDir /opt/cisco/uccx,
version=8.5.1.11003-32, serVer=1
    Service163: name=Cisco Unified CCX Database, Feature Service, isActivationSupported=false,
node=002, state=IN SERVICE, master, parent=null, type=DB Services, logDir:
/common/informix/crs/???, en=true, uT=928, rT=0, version=8.5.1.11003-32, serVer=4
  Component202: type=Cisco Recording, state=IN SERVICE, en=true, prim=false, node=002,
activationTime=1348140987000, parent=null, uT=439, rT=198, rootDir /opt/cisco/uccx,
version=8.5.1.11003-32, serVer=1

9823: Dec 15 12:42:38.866 IST %MCVD-CLUSTER_MGR-7-UNK:Post Convergence Event:
CONVERGENCE_STARTED, name=Cisco Unified CCX Engine
9824: Dec 15 12:42:38.866 IST %MCVD-CLUSTER_MGR-7-UNK:Cl Mgr: Cisco Unified CCX Engine
Convergence Started
9825: Dec 15 12:42:38.866 IST %MCVD-CLUSTER_MGR-7-UNK:try to process
MasterConvergenceCompletedCmdImpl: name Cisco Unified CCX Engine, nodeId=1, type=MASTER_DROPPED,
uniqueId=66, master=false, updateTick=3101, baseTick=3100, nodeCurrentTick=3101
9826: Dec 15 12:42:38.866 IST %MCVD-CLUSTER_MGR-7-UNK:process MasterConvergenceCompletedCmdImpl:
name Cisco Unified CCX Engine, nodeId=1, type=MASTER_DROPPED, uniqueId=66, master=false,
updateTick=3101, baseTick=3100, nodeCurrentTick=3101
```

```
9827: Dec 15 12:42:38.866 IST %MCVD-CLUSTER_MGR-7-UNK:JavaService66: Cisco Unified CCX Engine on
node 1 change master from true to false
```

This is when the Convergence started. Therefore, the algorithm explained earlier is used to elect the master. Here, notice the states of both nodes:

```
Node id=001, addresses=[172.30.72.2], MAC addresses=[95eab6e4c4cb], compName=UCCXPUB,
state=PARTIAL SERVICE, en=true, rmiPort=6999, masterPort=1994
       VersionInfo: [ Version=8.5.1.11003-32, crsRelease=8.5.1.11003-32, crsServiceRelease=,
crsEngineeringSpecial=, dbEdition=IDS, dbVersion=V11, installTime=1348064353000,
upgradeTime=1348064353000, jtapiClientVersion=8.6(2.10000)-2 ]
       cT=3275, uT=3275, rT=534, serVer=3, cvdVer=3, points=0

Node id=002, addresses=[172.30.83.2], MAC addresses=[279f2d5ba86d], compName=UCCXSUB, state=IN
SERVICE, en=true, rmiPort=6999, masterPort=1994
       VersionInfo: [ Version=8.5.1.11003-32, crsRelease=8.5.1.11003-32, crsServiceRelease=,
crsEngineeringSpecial=, dbEdition=IDS, dbVersion=V11, installTime=1348139852000,
upgradeTime=1348139852000, jtapiClientVersion=8.6(2.10000)-2 ]
       cT=969, uT=969, rT=528, serVer=3, cvdVer=3, points=0
```

Therefore, going by the algorithm, the mastership is handed to Node 2 (point 1 in the algorithm). This explains why the UCCX Node 2 became the Master after the convergence.

However, you have to check why the Node 1 was in Partial Service. It was in Partial Service due to the Telephony Subsystem:

```
name=Unified CM Telephony Subsystem, Feature Service, isActivationSupported=false, node=001,
state=PARTIAL SERVICE
```