System High Availability Architecture

James Ni
08/21/2008
System HA Architecture

- MySQL
- HA-Aware Application
- Replication Replica (local)
- Replication Replica (standby)
- Kernel level heartbeat
- HA-Aware Application
- Health Monitor
- System HA Framework
- Heartbeat
- sm
- Heartbeat
- ipvs admin
- ipvs kernel
- IPVS
- DRBD
- MySQL
- Linux HA
- Active Node
- Standby Node
- VIP Interface
- MySQL data
- DRBD Synch traffic
- HA API
- Control interface
- Replica synch. msgs
- Appl. State data replication
System HA Architecture

- **System HA Framework**
  - Replica: an in-memory, high speed data repository used by applications to store and replicate critical data for fault recovery and system failover
  - Health Monitor: An HA sub-component to monitor the healthiness of HA-aware application processes
  - HA State Machine: coordinates state changes between all the various components including IPVS, Heartbeat, management, node role arbitration etc.
  - Application APIs: interface between Application processes and HA framework for various HA service access
  - HA kernel module control interfaces: for HA framework to control all HA related linux kernel modules (DRBD, IPVS, Heartbeat etc.)
System HA Architecture

- **Linux HA kernel modules**
  - **DRBD**: Block devices designed as a building block to form high availability (HA) clusters
    - Buffer file system applications (e.g. MySQL) from the disc mount
    - Do whole block device mirroring via network
  - **LVS**: Linux virtual service
    - IPVS admin: control interface for controlling IPVS
    - IPVS kernel: Linux kernel module to perform the IPVS functions
  - **Linux Heartbeat**
    - A linux kernel module to provide cluster heart-beating functions
    - Also provides message delivery etc. functions
    - HA framework gains access to Linux heartbeat services via HA APIs
System HA Architecture

- **Active**
  - Application
  - HA IFs
  - HA APIs

- **Standby**
  - Application
  - HA IFs
  - HA APIs

**Function Interface**

**Message Interface**

**HA Framework**
System HA Architecture

- HA Framework Overview
- System Operation Component
  - HA framework
- System Operation Application Interface
- System Operation Agent
- System Operation Master
- Application HA Contexts
- Replication Checkpoints
  - Application specific replication triggers
- Replication Repertory
  - In memory hashed replication data storage
- HA APIs
  - APIs for applications to access HA framework functionalities
HA Framework Overview

- **Two functional layers**
  - Control layer: connectivity, self-maintenance
  - Service layer: services to applications

- **Functions and Services**
  - Aggregates system fault detections
  - Provides heart-beating mechanism in HA cluster
  - Provides HA replication mechanism to applications via APIs
  - Manages replication repertory
  - Controls failover procedures (restoration, activation auditing)
  - Provides logic addressing mechanism (address mapping) to all applications

- **Usage**
  - Register and use
  - Limited standard functions/methods to be implemented by application classes
HA framework – fault detection, classification and aggregation

- **Fault detection**
  - Error condition reporting: software breaks, signal capture, core dump
  - Operation status monitoring: keep-alive/heartbeat

- **Fault severity classification and aggregation**
  - Minor errors, log and keep going on
  - Major errors, restart the affected application process
  - Fatal errors, failover or reboot the affected node
HA Framework – System Operation Component

- The main control logic of the HA framework
- Provided functions like:
  - Interface with the manufacture provided chassis manager to manage all chassis inventories and hardware failures if needed
  - Manage the chassis physical and logical addressing schemes for HA logic and inter-component communications
  - Monitor the overall system operation, process operation healthiness
  - Restart individual malfunctioned processes if HA failover is not necessary
  - Reboot / Restart node on fatal service interrupting failures if redundancy is not available
  - Detect the presence and absence of all nodes via heart-beating mechanism
  - Collect all hardware and software error conditions and translate them to appropriate failover triggers
  - Trigger the HA failover and control the failover timing, sequence and pace
  - Maintain system-wide inter-node, inter-component synchronization
  - Manage all system-wide global variables and provide access interfaces to other components
  - Provide transportation mechanism for data context replication.
HA Framework – System Operation Component

- A distributed component
  - Master process runs on standby node (FSM)
  - Agents process runs on active nodes (FSM)
  - Application interface functions runs in each monitored application process (FSM)
Hash indexed storage for HA replication data

Implementation:
- Using private design, simple and efficient (preferred)
- Using standard database, flexible, time saving

Repertory location
- On dedicated nodes (preferred)
  - Enables self-reboot non-interruption HA
  - Repertory persist on failover – no need to do batch replication when new standby node is available
  - No batch replication needed
  - Failover might be slow
  - Dedicated nodes needed to hold repertory
- On standby node
  - Self-reboot non-interruption HA is not supported
  - Repertory not persist – replication data lost on failover
  - Large memory requirement on standby node
  - Fast failover can be achieved
HA framework – Application APIs

- **Application interface class as a base class of subsystem class**
  - Provides application error reporting mechanism
  - Provides application heartbeat mechanism
  - Provides APIs to application for replication, restoration, activation and auditing
  - FSM implementation to interface with system operation agent for all HA related handlings
HA framework – replication procedure

- **Batch replication** (not required in dedicated replication repertory case)

- **Incremental replication**
HA framework – restoration procedure

- Restore service contexts into runtime condition from repertory on failover

```
Restoration on standby module

Preordering all application instances to be restored

Check readiness and restoration timing

Try to restore instance n of application M (n=1,2,...) (M=a,b,...)

Trigger to restore dependence

Try to restore instance x of application Y

Trigger to restore dependence

Try to restore instance y of application Y

Trigger to restore dependence

Try to restore instance s of application R

Restore instance n of application M

Restoration done notification

Restore instance x of application Y

Restoration done notification

Restore instance s of application R

Restoration done notification

Repeat for all un-restored application instances
```
HA framework – activation procedure

- Activate restored service contexts after failover
**HA framework – auditing procedure**

- Auditing for consistency after failover
  - Internal auditing
  - External auditing (if applicable)
HA framework – post-failover cleanup procedure

- **Cleanup all orphan application contexts**
- **Internal cleanup**
  - Master Component
    - Existing service instances (1-30)
  - Slave (1) Component
    - Existing service instances (1-25, 31)
  - Slave (2) Component
    - Existing service instances (1-26, 32)
  - Slave (3) Component
    - Existing service instances (1-23, 33)

  **Post-failover auditing procedure**
  - Incomplete instances 24-30 identified
    - Release incomplete instances 24-30 and inform all associated neighbors
      - Existing service instances (1-23)
  - Orphan 31 identified
    - Release orphan 31 and inform all associated neighbors
      - Existing service instances (1-23)
  - Orphan 32 identified
    - Release orphan 32 and inform all associated neighbors
      - Existing service instances (1-23)
  - Orphan 33 identified
    - Release orphan 33 and inform all associated neighbors
      - Existing service instances (1-23)

- **External cleanup**
  - System Services
    - Existing service instances (1-30)
  - External Auditing Procedure
    - Orphans (26-30) identified
    - Terminate instances (26-30)
    - Existing service instances (1-25)
  - External entity
    - Existing service instances (1-25)
HA framework – procedures overview

- Standby Module
  - Service instances (may be packed) saved in replication database
  - Service instances restoration
  - Service instances activation
  - Standby Module takes the active role
  - Restored service instances internal auditing and cleanup
  - Restored service instances external auditing and cleanup
  - New active service module is now clean and functional

- Active Module
  - Service instances established
  - Service instances replication
  - Active module failure occurs
  - Failover trigger

- External Entity
  - New service requests are blocked during this period
  - New services are now accepted on the new active module
  - Out-of-synch external service instance cleanup
System Operation Component

- **HA framework functions**
  - Error condition reporting interface
  - Local application process operation status monitoring
    - Heartbeat (Application ↔ system operation agent)
  - HA data replication interface
  - HA control
    - Heartbeat (Active ↔ standby nodes)
    - Process restart on failure
    - node restart / reboot
    - Failover control
    - Standby role relay
  - Replication data transportation and storage

- **A distributed component**
  - Application interface (part of each application process)
  - System Operation Agent (process running in active nodes)
  - System Operation Master (process running in standby node)

- **Provide generic HA services to applications via APIs**
System Operation Component

Local failure monitoring, reporting and handling etc. HA procedure

Inter nodes failure monitoring, reporting and handling etc. HA procedure
System Operation Application Interface

- **Generic HA interface class**
  - Provide all HA APIs to application processes
  - Implementation
    - Base class of subSystem class or
    - A class member of subSystem class
  - FSM core
    - Interface with local System Operation Agent to monitor application operation status
    - Transport application operation status reports and replication data
    - Execute HA control commands
System Operation Application Interface

Functions

- Aggregates local operation status of the monitored application process and reports to the controlling system operation agent.
- Provides heartbeat mechanism between application process and its controlling system operation agent process, responses to requests from the controlling agent.
- Replicates application process data to standby node via its controlling agent.
- Reacts to various commands from the controlling agent, such as
  - Application process stop. This command will force the application process to exit.
  - Application process restart. This command will force the application process to restart.
  - System operation interface role setting. This command will set the active/standby role of the system operation interface.
  - Switchover. This command can be triggered by either a manual switchover request or a failure caused failover, and will suspend all application process data replication in an active node and start the switchover procedure in the standby node.
System Operation Application Interface

## Application Interface FSM

### States definitions
- **SO_IF_ST_NULL**: system operation interface NULL state.
- **SO_IF_ST_CONNECTED**: IPC connection to the controlling system operation agent established.
- **SO_IF_ST_ASSOCIATED**: system operation master-agent association establishment notification received.
- **SO_IF_ST_ACTIVE**: role setting request (ACTIVE) received from the agent.
- **SO_IF_ST_STANDBY**: system operation interface is in full functional state but plays the standby role.
- **SO_IF_ST_SWITCHING**: in process of switching from standby role to active role.
- **SO_IF_ST_SWITCHED**: switch-over done, waiting for passing standby role on.

### Events definitions
- **SO_IF_EV_AGT_CONNECT**: system operation agent IPC connection event.
- **SO_IF_EV_AGT_DISCONNECT**: system operation agent IPC disconnection event.
- **SO_IF_EV_AGT_ASSOC_UP**: system operation agent-master association establishment notification event.
- **SO_IF_EV_AGT_ASSOC_DOWN**: system operation agent-master association breakdown notification event.
- **SO_IF_EV_AGT_STATUS_REQ**: application process operation status request sent from system operation agent.
- **SO_IF_EV_AGT_STOP_REQ**: application process stop request received from system operation agent. This event will cause the application process to exit.
- **SO_IF_EV_AGT_RESTART_REQ**: application process operation restart request received from system operation agent. This event will cause the application process to restart.
- **SO_IF_EV_AGT_REPLICATION_RSP**: replication response received from the system operation agent.
- **SO_IF_EV_AGT_ROLE_REQ**: role setting request received from the system operation agent.
- **SO_IF_EV_AGT_SWITCHOVER_UREQ**: unacknowledged failover/switchover request received from system operation agent (can be of restoration, activation, auditing and cleanup etc. different action types).
- **SO_IF_EV_AGT_SWITCHOVER_DONE**: failover/switchover done notification received from system operation agent.
- **SO_IF_EV_REPLICATION_TO**: application process replication timeout event.
- **SO_IF_EV_ERROR_TO**: application process error report timeout event.
- **SO_IF_EV_HEARTBEAT_TO**: application process heartbeat timeout event.
System Operation Agent

- A process running in every node
  - Both active and standby
  - Aggregate local operation status and report to system operation master
  - Deliver application replication data to replication repertory
  - Pass HA related commands from system operation master to application processes

- Implementation
  - Dual FSMs – to simplify the control logic
  - Application side FSM interfaces with application interface FSM
  - Master side FSM interfaces with system operation master
System Operation Agent

System Operation Agent Application Side FSM

- **States definitions**
  - SO_AGT_APP_ST_NULL: the very initial state
  - SO_AGT_APP_ST_ACTIVE: the system operation agent application side state-machine instance has established the IPC connection with the corresponding application process

- **Events definitions**
  - SO_AGT_APP_EV_CONNECT: IPC connection to managed application process established.
  - SO_AGT_APP_EV_DISCONNECT: IPC connection to managed application process torn down.
  - SO_AGT_APP_EV_START_IND: start indication received from application process
  - SO_AGT_APP_EV_STATUS: status update received from application process. This message also serves as the heartbeat message.
  - SO_AGT_APP_EV_ERROR: error report received from application process
  - SO_AGT_APP_EV_STATUS_CFM: status confirmation reply received from application process.
  - SO_AGT_APP_EV_STOP_UIND: unacknowledged stop indication received from application process
  - SO_AGT_APP_EV_RESTART_UIND: unacknowledged restart indication received from application process
  - SO_AGT_APP_EV_STOP_CFM: stop confirmation reply received from application process
  - SO_AGT_APP_EV_RESTART_CFM: restart confirmation reply received from application process
  - SO_AGT_APP_EV_ROLE_CFM: role set confirmation received from application process
  - SO_AGT_APP_EV_REPLICATION_IND: replication request received from application process
  - SO_AGT_APP_EV_REQUEST: request sent to application process
  - SO_AGT_APP_EV_REQUEST_TO: request timeout event
  - SO_AGT_APP_EV_HEARTBEAT_TO: heartbeat timeout event
System Operation Agent

- **System Operation Agent Master Side FSM**
  - **States definitions**
    - SO_AGT_MST_ST_NULL: the very initial state
    - SO_AGT_MST_ST_ASSOCIATED: IPC connection to system operation master established
    - SO_AGT_MST_ST_ACTIVE: “active” role setting command received from system operation master
    - SO_AGT_MST_ST_STANDBY: “standby” role setting command received from system operation master
    - SO_AGT_MST_ST_SWITCHING: system operation agent is in failover/switchover process
    - SO_AGT_MST_ST_SWITCHED: switchover done, waiting for pass standby role on to new standby unit
  - **Events definitions**
    - SO_AGT_MST_EV_CONNECT: IPC connection to system operation master established.
    - SO_AGT_MST_EV_DISCONNECT: IPC connection to system operation master broken.
    - SO_AGT_MST_EV_STATUS_REQ: operation status request from system operation master
    - SO_AGT_MST_EV_STOP_REQ: stop request received from system operation master. This event will cause the node to shutdown.
    - SO_AGT_MST_EV_RESTART_REQ: restart request received from system operation master. This event will cause the node to reboot/restart.
    - SO_AGT_MST_EV_REPLICATION_RSP: replication response received from system operation master
    - SO_AGT_MST_EV_BATCHREP_REQ: batch replication start request received from system operation master
    - SO_AGT_MST_EV_ROLE_REQ: role setting request received from the system operation master.
    - SO_AGT_APP_EV_SWITCHOVER_UREQ: unacknowledged switchover request from system operation master.
    - SO_AGT_APP_EV_SWITCHOVER_DONE: failover/switchover done event
    - SO_AGT_APP_EV_REPLICATION_IND: replication indication received from the application side
    - SO_AGT_MST_EV_HEARTBEAT_TO: heartbeat timeout event
A process running on standby node

- Monitors the operation status and healthiness of all active nodes within the redundancy cluster
- Aggregates service context replication data from all active node and save into the replication repertory
- Manages replication repertory functions, including
  - Replication data record addition, deletion, modification etc.
  - Replication data compression or decompression if necessary
- Initiates the failover procedure on active node failure event
- Terminates and handles administrative commands from network administrator
  - Shutdown a node within the redundancy cluster
  - Restart a node within the redundancy cluster
  - Trigger a manual switchover between the standby node and any one of the active nodes within the redundancy cluster
- Executes and controls failover/switchover procedure
  - Restoration
  - Activation
  - Auditing
  - Cleanup
- Passes the standby role to new standby node after failover/switchover
System Operation Master

- **An FSM core**
  - Interface with all system operation agents in the HA cluster
System Operation Master

- System Operation Master FSM
  - States definitions
    - SO_MST_ST_Null: the very initial state
    - SO_MST_ST_ACTIVE: system operation master instance has connected with the corresponding agent and is ready to act all designed functions.
    - SO_MST_ST_SWITCHING: in process of failover/switchover
    - SO_MST_ST SWITCHED: the post-failover/switchover state, waiting for passing control to the new system operation master in the new standby node when available
System Operation Master

- **System Operation Master FSM**
  - **Events definitions**
    - `SO_MST_EV_CONNECT`: IPC connection to agent established
    - `SO_MST_EV_DISCONNECT`: IPC connection to agent down.
    - `SO_MST_EV_START_IND`, start indication received from the agent
    - `SO_MST_EV_STATUS`: status update received from the agent, this is the heartbeat message from the agent
    - `SO_MST_EV_ERROR`: error report received from the agent
    - `SO_MST_EV_STATUS_CFM`: status confirmation received from the agent, this is the reply to a master issued status request.
    - `SO_MST_EV_STOP_UIND`: unacknowledged agent stop indication received. This is an indication from the agent on the shutting down.
    - `SO_MST_EV_RESTART_UIND`: unacknowledged agent restart indication received. This is an indication from the agent on the rebooting.
    - `SO_MST_EV_STOP_CFM`: agent stop confirmation received. This is the reply to a master issued stop request.
    - `SO_MST_EV_RESTART_CFM`: agent restart confirmation received. This is the reply to a master issued restart request.
    - `SO_MST_EV_AGT_ROLE_CFM`: setting role confirmation received from system operation agent. This is the agent reply to a master issued role setting request.
    - `SO_MST_EV_REPLICATION_IND`: agent replication indication received. This is the message containing the replication data.
    - `SO_MST_EV_STOP_AGT_REQ`: high level request to stop an agent. This request will cause the targeted node to shut down.
    - `SO_MST_EV_RESTART_AGT_REQ`: high level request to restart an agent. This request will cause the targeted node to reboot/restart.
    - `SO_MST_EV_SWITCHOVER_REQ`: high level switchover request. This request will trigger a failure caused failover or a manual switchover.
    - `SO_MST_EV_SWITCHOVER_DONE`: switchover done event.
    - `SO_MST_EV_HEARTBEAT_TO`: heartbeat timeout event
    - `SO_MST_EV_REQUEST_TO`: request timeout event
System Operation Component Interaction Overview

- System Operation Component Startup Procedure
System Operation Component Interaction Overview

- **Switchover/failover procedure**

![Diagram of system operation component interaction]

- Failure detected on active TCS unit or manual switchover command received from network administrator
- Service context data restoration, activation, auditing and cleanup
- Switchover done
- Acting as the protected active TCS unit
- Waiting for new standby TCS unit and relay the standby role on
- Stop application services
- Pending for actions form network administrator

**Notes:**
- Master
- Agent master side
- Agent application side
- Application interface
- Active Node
- Agent master side
- Agent application side
- Application interface
Standby role relay procedure

Previous standby and now active Node

Master
Agent master side
Agent application side
Application interface

IPC Connect
Role set (standby)
IPC Disconnect
Association down
IPC Connect
Association up
Role set (active)
Role set (active)

Standby role relayed to the new TCS unit, batch and incremental service context data replication started

Newly booted up Node (will act as the new standby node)

Master
Agent master side
Agent application side
Application interface

Null
Null
Null

IPC Connect
ASSOCIATED
ACTIVE
CONNECTED

Association notify
Spawn out a master
Association down
ASSOCIATED
ACTIVE
ASSOCIATED

Role set (standby)
Role set (standby)
STANDBY
STANDBY

Waiting for new standby TCS unit and relay the standby role on
Free the master
System Application Contexts

- **Application Specific Contexts need to be preserved for HA**
  - SIP subscriber contexts
  - SIP call control contexts
    - Regular calls
    - Emergency calls
  - SMS contexts (??)
  - CMonitor and CDR contexts
  - Diameter Base Node Contexts
    - Stateful sessions: session state context
    - Stateless sessions: pending request queue
  - SS7 VLR contexts
    - Request queue
    - Subscriber ID map
    - Device ID map

- **Referenced Context replication**
  - Handlings of reference pointers
  - Handlings of reference indexes
  - Handlings of embedded contexts
  - Separate from the replication of referencing contexts

- **Applications to make decisions on**
  - What are to be replicated
  - Definition of replication data structure

- **Replication coordination and synchronization**
System Replication Checkpoints

- **General roles are**
  - Add replication when entering “stable” state(s)
  - Delete replication when leaving ‘stable” state(s)
  - Modify replication only when necessary (optional)

- **Stable states definitions are very component specific**

- **Special replication handling**
  - Emergency call context replication

- **System application specific checkpoints**
  - Subscriber: stable state “registered”
  - Call control: stable state “active”
  - Diameter Base Node
    - Stateless session: request queue
    - Stateful session: session state
  - VLR
Multiple Hash tables

- Node ID and application ID are used to index to the hash table
- Application data type and data key are used as hash key
HA APIs

- Implemented in System operation application interface class
- For applications to access HA framework services
- APIs
  - HA registration API. Used for application to register with the HA framework for HA services
  - HA replication API. A generic API called at application checking points to deliver application specific replication data to HA framework for replication
    - Addition, Deletion, Modification
  - HA error report API. An API to be called by application to reported the encountered error condition to the HA framework
  - HA command handler API. An API for HA framework to deliver command to applications.
  - HA failover/switchover API: an API used by HA framework to trigger the application specific failover/switchover procedures on the standby node.