Dynamic LAG Application (DLA) for VOLTHA Whitebox OLTs

ONF Broadband Community Meetup 2023

Presenters:
Dogukan Gulyasar
Burak Gurdag
OUTLINE

- Motivation
- LACP Protocol Overview
- Previous Work
- DLA Design Overview
- DLA Components
- Development Phases
- Items to be Discussed
- Q&A
WHAT IS LINK AGGREGATION? WHY DO WE NEED IT?

Link Aggregation is the combination of multiple network connections in parallel.

- Connects routers, switches, hosts, firewalls etc…
- We called this combination of physical ports as a link aggregation group (LAG).

- It increases total throughput.
- Provides redundancy. If a port fails, the other port will resume as a primary link and continue to operate.
- Divides traffic between links via various hashing mechanisms.
- Implementation followed vendor-independent standards such as Link Aggregation Control Protocol (LACP). In that way it works with multiple vendors.
Two protocols:
- LACP
- PAGP

LACP is a Open Standard protocol
→ Defined in 802.3ad specification.
→ Control the bundling of several physical ports together
to form a single logical link
→ Automatic bundling of links by sending LACP packets

PAGP is a Cisco-proprietary protocol
→ Can be used only on Cisco switches
→ Invented in early 1990s
→ Dynamically group similarly configured ports
PREVIOUS WORK IN SEBA

● Static LAG support on the OpenOLT Agent level
  ○ OLT level (i.e. not visible to VOLTHA)
  ○ Developed as a PoC
  ○ Not upstreamed to the community
    ■ Dynamic LAG was required
      • Added to the wishlist

● LAG work on the community side
  ○ Discussions in the brigade meetings
  ○ LAG folder under ONF Drive
  ○ Notes of the “Multi-NNI/LAG Discussion” Meeting with ONF
● **DLA Core**: DLA daemon starts its execution from here. It holds LACP stack as well.

● **DevCLI**: CLI of DLA for testing/debugging

● **REST Server**: NBI of DLA

● **LAG Manager**: LAG Management Entity in the control plane (position is TBD)

● **BAL**: Broadband Abstraction Layer (BAL) for PON and switch subsystems.

● **OpenOLT Agent**: VOLTHA driver for whitebox OLTs

● **OpenOLT Adapter**: VOLTHA adapter for whitebox OLTs
Components:
- LACP Daemon (i.e. LACP Stack)
  - Based on LibreSwitch LACP Daemon
- Customized OVSDB Interface Implementation for Dynamic LAG
- BAL Wrapper for LAG management
  - Based on BAL 3.10.4.4

BAL WRAPPER
- BAL API calls gathered in bal_wrapper lib
  - DLA Core initialized as a BAL Host Application (like OpenOLT Agent)

- It connects the DLA process (LACPd) to the BAL Core Service and configure the corresponding BAL objects:
  - ACL to trap incoming PDUs
  - Flow to packet out outgoing PDUs
  - LAG Interfaces

DEV CLI
- CLI for debugging and testing DLA
- Connected via telnet
- Functionalities:
  - Create/Delete LAG
  - Add/Remove LAG Members
  - Dump/Write LAG Config
  - Dump LAG states

REST SERVER
- Northbound Interface of DLA
- Based on Edgecore Public PSME REST Server Implementation
  - PSME REST API
- API Functionalities:
  - Create/Delete/Modify LAG Interfaces
  - Get LAG Status
  - Get LAG Statistics
  - Subscribe to LAG-related Events
    - LAG UP/DOWN/DEGRADED
    - NNI Member Events
DLA CORE - LACP STACK

LACP Stack Initialization:

- LACP Stack (protocol queues and event listeners)
- Rest Server
- DevCLI
- LACPd as the host application attached to the BAL Core

Stack Initialization

- Initializes all interfaces & all LAGs lists
- Configures the OLT switch with the LAG global parameters.
- Install ACL to trap LACP PDU packets to the host.
- Configures the System ID and priority.
- Creates all LAG groups from the config.yaml file.
- Configures all NNI from the config.yaml file as LAG members.

Adding member

Add NNI to LAG
- Install flow and ACL to send and receive PDUs.

Removing ACL from NNI and insert it in the LAG interface.
- Partner detection & aggregation agreement.
DEVELOPMENT PHASES

• Phase 1
  - LAG Configuration based on DLA config file
    ▪ Control plane is not LAG-aware
  - Tested on
    ▪ Zyxel SDA3016SS
    ▪ Edgecore ASGvOLT64
  - Completed

• Phase 2
  - LAG Configuration through NBI
  - LAG Manager and its placement should be clarified
    ▪ Level of LAG-awareness in the control plane should be determined
ITEMS TO BE DISCUSSED (IN TST MEETINGS?)

- VOLTHA assumes single OLT uplink: NNI-0
  - Should SEBA/VOLTHA be LAG-aware?
- Who will be the LAG Manager?
  - Device Manager?
  - A new entity?
- Issues with BAL 3.10.4.4
Thank You

Q&A

dogukan.gulyasar@netsia.com
burak.gurdag@netsia.com