A day in the life of an autonomic function

P. Peloso, L. Ciavaglia

IETF 94th
Motivations

Autonomics can improve network operations.
Operators need unified management functions to use autonomics and gain confidence in it.

Common management functions of AF bring:
- trust in Autonomic Functions behavior
- capacity to control Autonomic Functions
- conflict avoidance mechanisms
Management of Autonomic Functions

3 phases

Installation
- To dynamically install ASA to Nodes.

Instantiation
- To allocate the network resources to be managed by ASA.
- To organize ASA in ASA domains.

Operation
- To control the running of ASA.
- To avoid conflicts between ASAs.
- To share knowledge between ASAs.

Focus of today
ASA life-cycle

3 states + 2 transit ones

- **Undeployed**
  - Advertizes blank Manifest

- **Installed**
  - Receives a Mandate
  - Advertizes its Manifest

- **Un-Instantiating**
  - Receives revoke Mandate

- **Instantiating**
  - Receives a Set-up cmd

- **Instantiated**
  - Receives a Set-down cmd

- **Operational**

- **Undeployed**
ASA Interactions

The Life-Cycle shows that:

**Entities** pertaining to the **Control of Autonomic Function** are interacting with the **ASAs** and serving **all ASAs**

- **AF Mgt Function**
- **Coordination Function**
- **Info sharing Function**
What is needed in the ANIMA ecosystem?
Current ANIMA picture

- GRASP preferably uses ACP
- ASA use GRASP signaling in-between them
- ASA monitor the equipment and modify its state directly using either NetConf, SNMP, call to Basic OS API...

Legend

Protocol engine
Simplest option to control ASAs:

- **Same as before**
- Plus use GRASP signaling between ASAs and AF Control Agents (Coordination, AF Mgt, Info Distribution)
- Hence multiple type of GRASP clients
The functions controlling autonomic functions NEED NOT being instantiated in each Node

Actually there even likely being instantiated in servers part of the ACP but not on network equipments like routers or switches.
Minimal control of ASA

Control when it runs
(and how it runs)

Know what it does to the network

Decide which equipments are under the ASA control
(Or vice-versa which ASAs control an equipment)
Control when an ASA runs

NEED
- On request Start and Stop the execution of ASA

SOLUTION
- Send a START command
- Send a STOP command
Control when an ASA runs

NEED
- On request Start and Stop the execution of ASA

SOLUTION
- Send a START command
- Send a STOP command

IN ANIMA
- Add to GRASP imperative commands type of message
Know what an ASA does to the network

NEED

- Know which network resources are modified by ASA control loop
- Know which network resources are monitored by ASA control loop

SOLUTION

- Disclose an ASA Manifest at ASA bootstrap time
Know what an ASA does to the network

**NEED**
- Know which network resources are modified by ASA control loop
- Know which network resources are monitored by ASA control loop

**SOLUTION**
- Disclose an ASA Manifest at ASA bootstrap time

**IN ANIMA**
- Disclose Manifest with GRASP Discovery messages
Decide which ASA control which equipment

NEED
- Give instructions to ASA during bootstrapping

SOLUTION
- Send a Mandate to ASA before end of bootstrap
Decides which ASA control which equipment

NEED
- Give instructions to ASA during bootstrapping

SOLUTION
- Send a Mandate to ASA before end of bootstrap

IN ANIMA
- Specify Intent formats compatible with Mandate and identify proper message in GRASP to convey Intent
Conclusion

Control when it runs

Minimal control of ASA

Start/Stop

Know what it does to the network

Manifest

Decide which equipments are under the ASA control

Mandate
Conclusion

Can we design a solution that oversees the operators trust in it?
Appendix
Autonomic Networking Infrastructure

The toolbox

Other option
- Segmenting between ASA-ASA and ASA-ANI function

Legend
- Protocol engine
- ANI function (serving ASAs)

Autonomic Node

ANI

ASA

ASA

ASA

GRASP Eng

AF Mgt Func

Coordination

Info distib

AFMP Eng

ACP Engine

Basic OS

20 | Autonomic Functions | IETF 94th
In case GRASP should absolutely not care about semantics of what is carried and roles of its clients
Deployment examples of AF