Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 20, 2019

B. Liu
Huawei Technologies
B. Carpenter
Univ. of Auckland
October 17, 2018

Roadmap to a Networkless World
draft-liu-nmrg-networkless-roadmap-01

Abstract

This document aims to illustrate possible approaches to make network management and operations more autonomic in several aspects. The ultimate goal is that the network could run all by itself, so that users and administrators may feel that there is no network to take care of at all (a.k.a. "Networkless"). The approaches are described in a form of different levels (inspired by the Self-Driven Car levels). The higher the level is, the more autonomic management capabilities the network would have.

Although some specific technologies are categorized into different levels, it is not the document's intent to rank them; rather, this document is more about discussing the possible next stage and the ultimate vision. Hopefully, this document could collect people's consensus in the industry and provide guidance for future technology developments.

Table of Contents

1. Introduction

As the network is evolving rapidly, the system is becoming more and more complex; thus managing a network is more and more challenging. It has been a common feeling in the industry that the operational expense of running networks is becoming a vital pain point. To address the management complexity challenges, there are new technologies emerging. For example, Autonomic Networking [RFC7575], which is under standardization in IETF Anima working group [Anima],

is following an approach to allow the network elements do more management related operations by themselves. SDN techniques have significantly improved the efficiency of network service delivery in some scenarios. Network function virtualization, network slicing, and the related orchestration techniques are expected to do the same. In future, the intent-based network concept, which focuses more on the operational simplicity perspective, should allow users or administrators to control the network system in a radically simple way (that is, driven by abstract intent, rather than by detailed configurations).

This document is not proposing a new technology, rather, it collects available tecnologies and illustrates possible future technologies and the final effect on network users or administrators. The ultimate goal is that the network could run all by itself, so that users oradministrators may feel like there no network to take care of at all (a.k.a. "Networkless").

In Section 2, network management is divided into several aspects for discussion, from an administrator's perspective. In each aspect there are automation and autonomicity levels to illustrate past (Level 0), current state of art (Level 1) and possible future technologies (Level 2-4). Section 3 focuses on some common and vital capabilities the network system needs to have, in order to support the goals described in Section 2.

2. Level-by-Level approach to Networkless

2.1. Self-Organization Levels

Self-organization represents the ability of network elements to autonomically connect with each other, form domains, or even decide the topology, hierarchy or architecture.

- Level 1: LAN auto-connection
  - E.g. current Ethernets can connected with each other without any configurations once the cables are connected.
- Level 2: IP auto-routing & NE auto-connection to NMS
  - IGP and BGP protocols allow the routers to connect with each other autonomically, assuming prefixes are assigned to links.
  - NEs automatically get connected with the NMS, current solutions includes DCN [Q: What is DCN?], Anima ACP [I-D.ietf-anima-autonomic-control-plane] etc. [Q: Mention Netconf "call home"?]

- Level 3: Network Areas Self-Division and Key NEs election
  - E.g. IGP Area self-division; controller election
- Level 4: Network Architecture and NE roles Self-identification
  - E.g. autonomically identify topology characteristics and divide network layers; autonomically identify roles such as access gateway, aggregation gateway, core gateway etc.

  [Note] More detailed technical discussion regarding to Level-3 and Level-4 please refer to Section 4.1 .
- Level 5: Self-Construction of Network Topologies
  - E.g. for wireless network or overlay virtual networks

2.2. Self-Configuration Levels

- Level 1: CLI
  - remote log-in, do configs one by one
- Level 2: NE Configs Auto-delivery
  - Administrators design detailed configurations of each NE, using NMS/Controller automatically deliver the configurations
- Level 3/4: NE Configs Auto-Compiling
  - Administrators design network architecture and solutions, the network autonomically compiles detailed NE configurations (centrally or in a distributed manner).
  - All detailed configurations are created by software.
  - More and more machine-native configurations rather than human interfaces.

  [Note] More detailed technical discussion please refer to Section 4.2 .
- Level 5: Network Self-Orchestration
  - Administrators/Apps only input highly abstracted service requests (e.g., build a wireless backhaul network), then the network would deduce all configurations.

2.3. Self-Optimization and Levels

This sub-section focuses on traffic forwarding performance of the network, mainly include path selection and QoS related issues.

- Level 1: Static Traffic Engineering
- Level 2: Auto Traffic Load Balance
  - Controller dynamically adjust paths to achieve balanced traffic load and congestion, according to specific algorithms;
  - NE can achieve port-based load balancing locally
- Level 3/4: Comprehensive SLA/QoS Self-Optimization
  - The network autonomically optimizes delay, bandwidth etc. according to Administrator's or App's requirements;
  - The network autonomically achieves measurement according to the optimization goal.
- Level 5: Autonomous Optimization
  - The network generates optimization policies by itself, and keeps the performance at the best level;
  - Meanwhile, achieves balance between performance and cost.

2.4. Self-Diagnostic Levels

This sub-section focuses on network fault diagnostic.

- Level 1: NMS-assisted manual diagnostic
  - Administrators use tools like ping/tracroute for mannual diagnostics
- Level 2: Automatic Data Analysis
  - Software collects data around the whole network, and use data mining or machine learning and decision tree to aggregate alarms and analyze the cause.
- Level 3/4: Precise Fault Location
  - Precise alarms to report the exact fault events.

- Precise location to reveal the real root cause.

o Level 5: Fault Prediction

- Network uses current data (e.g. bit error rates, temperature alarms) to predict failures.

2.5. Self-Healing Levels

o Level 1: NMS-assisted manual healing

- Administrators use NMS to manually recover the configurations or do the adjustment.

o Level 2: Protocol-based Healing

- Fixed healing functions built into NEs, such as BFD, and FRR etc.

o Level 3: Programmable Healing

- Administrators can set specific healing policies based on a set of general and abstracted rules of dealing with fault.

- Automatic call-out of technician.

o Level 4/5: Fault Avoidance

- According to the prediction, avoid the fault by backup, adjust traffic, early call-out of technician, etc.

3. Key Capablities to Achieve Networkless

3.1. Network Perception

o Level 1: NE-based Statistics and Probe

- E.g. NE port statistics; end to end probe. Based on known fixed topology.

o Level 2: Network Visualization

- Telemetry, logs/event analysis etc.

- Display current toplogy.

o Level 3: Real-time Holographic Network Data

- Network Digital Twin;
- NE deeply sense local traffic and fault etc.

o Level 4: Network Modeling and Pattern Recognition

- Comprehensive modeling for complex network problems;
- Pattern recognition to identify current network status

o Level 5: Network Event/Traffic Trend Prediction

- Based on ML trained on past observation of similar networks.

3.2. Decision and Reasoning

o Level 1: Fixed Control Loops

- The control loop functions are embedded in specific protocols/ modules, such as IGP, DHCP, Anima BRSKI [I-D.ietf-anima-bootstrapping-keyinfra] , and Anima ACP [I-D.ietf-anima-autonomic-control-plane] etc.

o Level 2: Programmable Control Loops

- Algorithms (in Controller or Autonomic Service Agent) for specific functions and scenarios
- might embed some Machine Learning capabilities, or outsource ML to a central resource.

o Level 3: Machine Learning [Q: Maybe this should be Level 2/3?]

- General control loops, driven by specific Intents (e.g. Intent provides the Reward definition of the reinforcement learning)

o Level 4: Machine Inference [Q: Maybe this should be Level 4?]

- Configuration, optimization, diagnostic, healing policies inference

o Level 5: (To be filled)

3.3. Operation Interface

o Level 1: CLI

- Manual management oriented interface; batch processing within a machine (e.g. Shell)

o Level 2: NE-level Primitive API

- Controller oriented NE-level API containing detailed configurations. (E.g. Openflow, Netconf/YANG)

o Level 3: NE-level Declarative API

- Orchestrator oriented NE-level declarative API

- Orchestrator doesn't need to care about detailed NE specific configurations

o Level 4: Network-level Declarative API

- User/Administrator oriented declarative API, to make the network be called as a service.

o Level 5: Machine-native Autonomous API

- The machines would autonomously construct the content of the APIs to fulfill the need of collaboration between modules.

- This level would likely be based on ML trained on similar networks with similar applications.

4. Possible next-step technologies

This section discusses some possible next-steps for the technologies described in Section 2. Basically, the next-steps are Level-3 or Level-4 for each aspect.

4.1. Self-Organization

Current technologies (such as the Level-2 Self-organization) can decently deal with the problem of how a device can get connected to the NOC and then get managed. After that, it still relies on human planning to properly configure the basic network connectivity, such as IP addresses, IGP etc. (This part of basic configurations is always called "underlay configurations" comparing to the overlay services.)

Thus, to simplify human work, it is expected that the system can do some "planning" work. Some critical aspects of network planning are as following, which are pre-conditions for both underlay configurations and overlay configurations.

- Routing domain division: the system can divide the devices into groups, according to some mechanisms.

- Network hierarchy recognition: the system can learn there is hierarchy in the network; and it can even recognize which are higher hierarchy and which are lower.

- Roles recognition: some device roles are directly related to the topology, such as access gateway, aggregation gateway, core gateway.

If the system can figure out the above things, then it would be much easier to create the specific configurations. The IP addresses could be assigned in a good order (e.g. from higher hierarchy to lower, keep the addresses in a certain prefix for a specific domain); the IGP could be inheritably configured according to the routing domain divisions.

4.2. Self-Configuration

This section is mostly regarding service configurations (e.g. VPN configurations).

The following figure shows a typical architecture of how current state-of-the-art technologies do configurations for services.

(preamble)

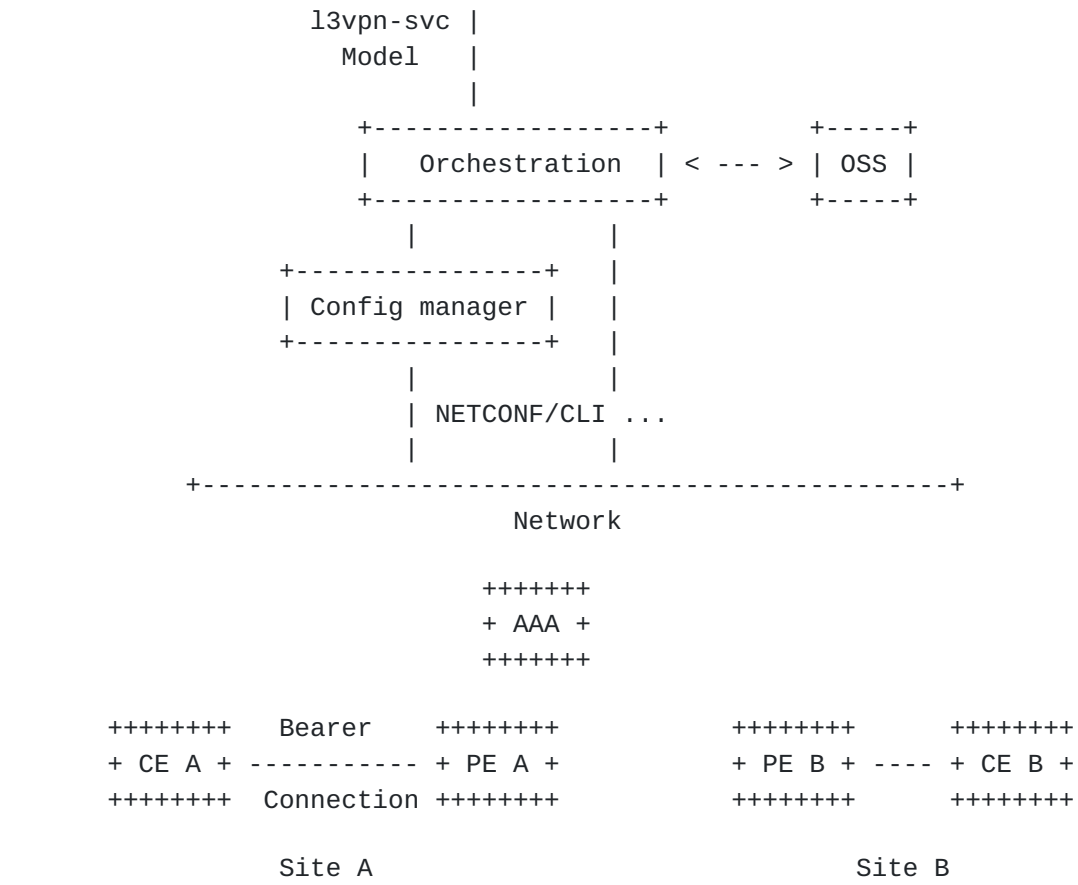

Figure 1: L3VPN Service Configuration Architecture (from RFC8299)

For this approach, there are several issues:

1. Too much details in currently defined service models, which implies

   - Cost a lot of human labor

   - The more details, the harder to achieve a unified and correct model

2. Orchestrator/Controller is hard to scale

   - Binding to specific service and underlay models; need to develop new instance when service/underlay varies

   - Need to compile each single model in each device

3. Southbound data models are hard to be unified

- Each vendor's capabilities are different
- Each operator's needs are different
- A long-term puzzle from the SNMP era, not fixed by Netconf/ YANG

To address Issue-1, we'll need some easy expression of the network service, this surely fits into the Intent-based network field. (TBD.)

To address Issue-2 we might need a intermediate common layer to separate the binding between specific service-level models and device-level configurations. (TBD.)

4.3. Self-Optimization

TBD.

4.4. Self-Diagnostic/Healing

TBD.

5. Security Considerations

Security mechanisms such as firewall placement, firewall or route filtering rules, authorization to join the network, key distribution, VPN encryption policy etc. are potentially subject to all of the above. However, raising security management to Levels 3 or 4 requires great confidence that the autonomic mechanisms are themselves foolproof. It is to be expected that security management remains at Level 0, 1 or 2 longer than most other aspects. An exception is threat or DoS detection, where ML techniques should be applicable in the short term.

6. IANA Considerations

No IANA assignment is needed.

7. Acknowledgements

The initial idea of this work and the "networkless" concept were from Xiaofei Xu.

8. Informative References

[Anima] "https://datatracker.ietf.org/wg/anima/about/".

[I-D.ietf-anima-autonomic-control-plane]
Eckert, T., Behringer, M., and S. Bjarnason, "An Autonomic Control Plane (ACP)", draft-ietf-anima-autonomic-control-plane-18 (work in progress), August 2018.

[I-D.ietf-anima-bootstrapping-keyinfra]
Pritikin, M., Richardson, M., Behringer, M., Bjarnason, S., and K. Watsen, "Bootstrapping Remote Secure Key Infrastructures (BRSKI)", draft-ietf-anima-bootstrapping-keyinfra-16 (work in progress), June 2018.

[I-D.ietf-anima-reference-model]
Behringer, M., Carpenter, B., Eckert, T., Ciavaglia, L., and J. Nobre, "A Reference Model for Autonomic Networking", draft-ietf-anima-reference-model-08 (work in progress), October 2018.

[RFC7575] Behringer, M., Pritikin, M., Bjarnason, S., Clemm, A., Carpenter, B., Jiang, S., and L. Ciavaglia, "Autonomic Networking: Definitions and Design Goals", RFC 7575, DOI 10.17487/RFC7575, June 2015, <https://www.rfc-editor.org/info/rfc7575>.

Authors' Addresses

Bing Liu
Huawei Technologies
Q14, Huawei Campus
No.156 Beiqing Road
Hai-Dian District, Beijing 100095
P.R. China

Email: leo.liubing@huawei.com

Brian Carpenter
Department of Computer Science
University of Auckland
PB 92019
Auckland 1142
New Zealand

Email: brian.e.carpenter@gmail.com