Network Working Group
Internet-Draft
Intended status: Informational
Expires: January 9, 2020

A. Clemm
Futurewei
L. Ciavaglia
Nokia
L. Granville
Federal University of Rio Grande do Sul (UFRGS)
J. Tantsura
Apstra, Inc.

July 8, 2019

Intent-Based Networking - Concepts and Overview
draft-clemm-nmrg-dist-intent-02

Abstract

Intent and Intent-Based Networking are taking the industry by storm. At the same time, those terms are used loosely and often inconsistently, in many cases overlapping and confused with other concepts such as "policy". This document is intended to clarify the concept of "Intent" and provide an overview of functionality that associated with it. The goal is to contribute towards a common and shared understanding of terms, concepts, and functionality which can be used as foundation to guide further definition of associated research and engineering problems and their solutions.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Traditionally in the IETF, interest with regard to management and operations has focused on individual network and device features. Standardization emphasis has generally been put on management instrumentation that needed to be provided to a networking device. A prime example for this is SNMP-based management and the 200+ MIBs that have been defined by the IETF over the years. More recent examples include YANG data model definitions for aspects such as interface configuration, ACL configuration, or Syslog configuration.

There is a sense and reality that in modern network environments managing networks by configuring myriads of "nerd knobs" on a device-by-device basis is no longer sustainable. Big challenges arise with keeping device configurations not only consistent across a network, but consistent with the needs of services and service features they are supposed to enable. Adoptability to changes at scale is a fundamental property of a well designed IBN system, that requires abilty to consume and process analytics that are context/intent aware at near real time speeds. At the same time, operations need to be streamlined and automated wherever possible to not only lower operational expenses, but allow for rapid reconfiguration of networks at sub-second time scales and to ensure networks are delivering their functionality as expected.

Accordingly, IETF has begun to address end-to-end management aspects that go beyond the realm of individual devices in isolation. Examples include the definition of YANG models for network topology [RFC8345] or the introduction of service models used by service orchestration systems and controllers [RFC8309]. In addition, a lot of interest has been fueled by the discussion about how to manage autonomic networks as discussed in the ANIMA working group. Autonomic networks are driven by the desire to lower operational expenses and make management of the network as a whole exceptionally easy, putting it at odds with the need to manage the network one device and one feature at a time. However, while autonomic networks are intended to exhibit "self-management" properties, they still require input from an operator or outside system to provide operational guidance and information about the goals, purposes, and service instances that the network is to serve.

This vision has since caught on with the industry in a big way, leading to a significant number solutions that offer "intent-based management" that promise network providers to manage networks holistically at a higher level of abstraction and as a system that happens to consist of interconnected components, as opposed to a set of independent devices (that happen to be interconnected). Those offerings include IBN systems (offering full lifecycle of intent), SDN controllers (offering a single point of control and administration for a network) as well as network management and Operations Support Systems (OSS).

However, it has been recognized for a long time that comprehensive management solutions cannot operate only at the level of individual devices and low-level configurations. In this sense, the vision of "intent" is not entirely new. In the past, ITU-T’s model of a

Telecommunications Management Network, TMN, introduced a set of management layers that defined a management hierarchy, consisting of network element, network, service, and business management. High-level operational objectives would propagate in top-down fashion from upper to lower layers. The associated abstraction hierarchy was key to decompose management complexity into separate areas of concerns. This abstraction hierarchy was accompanied by an information hierarchy that concerned itself at the lowest level with device-specific information, but that would, at higher layers, include, for example, end-to-end service instances. Similarly, the concept of "policy-based management" has for a long time touted the ability to allow users to manage networks by specifying high-level management policies, with policy systems automatically "rendering" those policies, i.e. breaking them down into low-level configurations and control logic.

What has been missing, however, is putting these concepts into a more current context and updating it to account for current technology trends. This document attempts to clarify the concepts behind intent. It differentiates it from related concepts. It also provides an overview of first-order principles of Intent-Based Networking as well as associated functionality. In addition, a number of research challenges are highlighted. The goal is to contribute to a common and shared understanding that can be used as a foundation to articulate research and engineering problems in the area of Intent-Based Networking.

2. Key Words

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Definitions and Acronyms

ACL: Access Control List

Intent: An abstracted, declarative and vendor agnostic set of rules used to provide full lifecycle (Design/Build/Deploy/Validate) to a network and services it provides.

Policy: A rule, or set of rules, that governs the choices in behavior of a system.

SSoT: Single Source of Truth - A functional block in an IBN system that normalizes user' intent and serves as the single source of data for the lower layers.

IBA: Intent Based Analytics - Analytics that are defined and derived from user' intent and used to validate the intended state.

PDP: Policy Decision Point

PEP: Policy Enforcement Point

Service Model: A model that represents a service that is provided by a network to a user.

4. Introduction of Concepts

The following section provides an overview of the concept of intent respectively intent-based management. It also provides an overview of the related concepts of service models, and of policies respectively policy-based management, and explains how they relate to intent and intent-based management.

4.1. Intent and Intent-Based Management

In the context of Autonomic Networks, Intent is defined as "an abstract, high-level policy used to operate a network" [RFC7575]. According to this definition, an intent is a specific type of policy. However, to avoid using "intent" simply as a synonym for "policy, a clearer distinction needs to be introduced that distinguishes intent clearly from other types of policies.

For one, while Intent-Based Management clearly aims to lead towards networks that are dramatically simpler to manage and operate requiring only minimal outside intervention, the concept of "intent" is not limited to autonomic networks, but applies to any network. Networks, even when considered "autonomic", are not clairvoyant and have no way of automatically knowing particular operational goals nor what instances of networking services to support. In other words, they do not know what the "intent" of the network provider is that gives the network the purpose of its being. This still needs to be communicated by what informally constitutes "intent".

More specifically, intent is a declaration of operational goals that a network should meet and outcomes that the network is supposed to deliver, without specifying how to achieve them. Those goals and outcomes are defined in a manner that is purely declarative - they specify what to accomplish, not how to achieve it. "Intent" thus applies several important concepts simultaneously:

- It provides data abstraction: Users and operators do not need to be concerned with low-level device configuration and nerd knobs.

- It provides functional abstraction from particular management and control logic: Users and operators do not need to be concerned even with how to achieve a given intent. What is specified is a desired outcome, with the intent-based system automatically figuring out a course of action (e.g. a set of rules, an algorithm) for how to achieve the outcome.

In an autonomic network, intent should be rendered by the network itself, i.e. translated into device-specific rules and courses of action. Ideally, it should not even be orchestrated or broken down by a higher-level, centralized system, but by the network devices themselves using a combination of distributed algorithms and local device abstraction. Because intent holds for the network as a whole, not individual devices, it needs to be automatically disseminated across all devices in the network, which can themselves decide whether they need to act on it. This facilitates management even further, since it obviates the need for a higher-layer system to break down and decompose higher-level intent, and because there is no need to even discover and maintain an inventory of the network to be able to manage it.

Tentative definition for intent-based networks Networks configuring and adapting autonomously to the user or operator intentions (i.e., a desired state or behavior) without the need to specify every technical detail of the process and operations to achieve it (i.e., the "machines" will figure out on their own how to realize the user goal).

Other definitions of intent exist such as [TR523] and will be investigated in future revisions of this document. Likewise, some definitions of intent allow for the presence of a centralized function that renders the intent into lower-level policies or instructions and orchestrates them across the network. While to the end user the concept of "intent" appears the same regardless of its method of rendering, this interpretation opens a slippery slope of how to clearly distinguish "intent" from other higher-layer abstractions. Again, these notions will be further investigated in future revisions of this document and in collaboration with NMRG.

4.2. Related Concepts

4.2.1. Service Models

A service model is a model that represents a service that is provided by a network to a user. Per [RFC8309], a service model describes a service and its parameters in a portable/vendor agnostic way that can be used independent of the equipment and operating environment on which the service is realized. Two subcategories are distinguished: a "Customer Service Model" describes an instance of a service as provided to a customer, possibly associated with a service order. A "Service Delivery Model" describes how a service is instantiated over existing networking infrastructure.

An example of a service could be a Layer 3 VPN service [RFC8299], a Network Slice, or residential Internet access. Service models represent service instances as entities in their own right. Services have their own parameters, actions, and lifecycles. Typically, service instances can be bound to end users, who might be billed for the service.

Instantiating a service typically involves multiple aspects:

o A user (or northbound system) needs to define and/or request a service to be instantiated.

o Resources need to be allocated, such as IP addresses, AS numbers, VLAN or VxLAN pools, interfaces, bandwidth, or memory.

o How to map services to the resources needs to be defined. Multiple mappings are often possible, which to select may depend on context (such as which type of access is available to connect the end user with the service).

o [I-D.ietf-teas-te-service-mapping-yang] is an example of such mapping - a data model to map customer service models (e.g., the L3VPM Service Model) to Traffic Engineering (TE) models (e.g., the TE Tunnel or the Abstraction and Control of Traffic Engineered Networks Virtual Network model)

o Bindings need to be maintained between upper and lower-level objects.

o Once instantiated, the service needs to be validated and assured to ensure that the network indeed delivers the service as requested.

They involve a system, such as a controller, that provides provisioning logic. Orchestration itself is generally conducted using a "push" model, in which the controller/manager initiates the operations as required, pushing down the specific configurations to the device. (In addition to instantiating and creating new instances of a service, updating, modifying, and decommissioning services need to be also supported.) The device itself typically remains agnostic to the service or the fact that its resources or configurations are part of a service/concept at a higher layer.

Instantiated service models map to instantiated lower-layer network and device models. Examples include instances of paths, or instances of specific port configurations. The service model typically also models dependencies and layering of services over lower-layer networking resources that are used to provide services. This facilitates management by allowing to follow dependencies for troubleshooting activities, to perform impact analysis in which events in the network are assessed regarding their impact on services and customers. Services are typically orchestrated and provisioned top-to-bottom, which also facilitates keeping track of the assignment of network resources. Service models might also be associated with other data that does not concern the network but provides business context. This includes things such as customer data (such as billing information), service orders and service catalogues, tariffs, service contracts, and Service Level Agreements (SLAs) including contractual agreements regarding remediation actions.

Like intent, service models provide higher layers of abstraction. Service models are often also complemented with mappings that capture dependencies between service and device or network configurations. Unlike intent, service models do not allow to define a desired "outcome" that would be automatically maintained by the intent system. Instead, management of service models requires development of sophisticated algorithms and control logic by network providers or system integrators.

4.2.2. Policy and Policy-Based Management

Policy-based management (PBM) is a management paradigm that separates the rules that govern the behavior of a system from the functionality of the system. It promises to reduce maintenance costs of information and communication systems while improving flexibility and runtime adaptability. It is present today at the heart of a multitude of management architectures and paradigms including SLA-driven, Business-driven, autonomous, adaptive, and self-* management [Boutaba07]. The interested reader is asked to refer to the rich set of existing literature which includes this and many other references. In the following, we will only provide a much-abridged and distilled overview.

At the heart of policy-based management is the concept of a policy. Multiple definitions of policy exist: "Policies are rules governing the choices in behavior of a system" [Sloman94]. "Policy is a set of rules that are used to manage and control the changing and/or maintaining of the state of one or more managed objects" [Strassner03]. Common to most definitions is the definition of a policy as a "rule". Typically, the definition of a rule consists of an event (whose occurrence triggers a rule), a set of conditions (that get assessed and that must be true before any actions are actually "fired"), and finally a set of one or more actions that are carried out when the condition holds.

Policy-based management can be considered an imperative management paradigm: Policies specify precisely what needs to be done when and in which circumstance. Using policies, management can in effect be defined as a set of simple control loops. This makes policy-based management a suitable technology to implement autonomic behavior that can exhibit self-* management properties including self-configuration, self-healing, self-optimization, and self-protection. In effect, policies define management as a set of simple control loops.

Policies typically involve a certain degree of abstraction in order to cope with heterogeneity of networking devices. Rather than having a device-specific policy that defines events, conditions, and actions in terms of device-specific commands, parameters, and data models, policy is defined at a higher-level of abstraction involving a canonical model of systems and devices to which the policy is to be applied. A policy agent on a controller or the device subsequently "renders" the policy, i.e., translates the canonical model into a device-specific representation. This concept allows to apply the same policy across a wide range of devices without needing to define multiple variants. In other words - policy definition is de-coupled from policy instantiation and policy enforcement. This enables operational scale and allows network operators and authors of policies to think in higher terms of abstraction than device specifics and be able to reuse the same, high level definition defintion across different networking domains, WAN, DC or public cloud.

Policy-based management is typically "push-based": Policies are pushed onto devices where they are rendered and enforced. The push operations are conducted by a manager or controller, which is responsible for deploying policies across the network and monitor their proper operation. That said, other policy architectures are possible. For example, policy-based management can also include a pull-component in which the decision regarding which action to take is delegated to a so-called Policy Decision Point (PDP). This PDP can reside outside the managed device itself and has typically global visibility and context with which to make policy decisions. Whenever a network device observes an event that is associated with a policy, but lacks the full definition of the policy or the ability to reach a conclusion regarding the expected action, it reaches out to the PDP for a decision (reached, for example, by deciding on an action based on various conditions). Subsequently, the device carries out the decision as returned by the PDP - the device "enforces" the policy and hence acts as a PEP (Policy Enforcement Point). Either way, PBM architectures typically involve a central component from which policies are deployed across the network, and/or policy decisions served.

Like Intent, policies provide a higher layer of abstraction. Policy systems are also able to capture dynamic aspects of the system under management through specification of rules that allow to define various triggers for certain courses of actions. Unlike intent, the definition of those rules (and courses of actions) still needs to be articulated by users. Since the intent is unknown, conflict resolution within or between policies requires interactions with a user or some kind of logic that resides outside of PBM.

4.2.3. Distinguishing between Intent, Policy, and Service Models

What Intent, Policy, and Service Models all have in common is the fact that they involve a higher-layer of abstraction of a network that does not involve device-specifics, that generally transcends individual devices, and that makes the network easier to manage for applications and human users compared to having to manage the network one device at a time. Beyond that, differences emerge. Service models have less in common with policy and intent than policy and intent do with each other.

Summarized differences:

- o A service model is a data model that is used to describe instances of services that are provided to customers. A service model has dependencies on lower level models (device and network models) when describing how the service is mapped onto underlying network and IT infrastructure. Instantiating a service model requires orchestration by a system; the logic for how to orchestrate/manage/provide the service model, and how to map it onto underlying resources, is not included as part of the model itself.

- o Policy is a set of rules, typically modeled around a variation of events/conditions/actions, used to express simple control loops that can be rendered by devices themselves, without requiring intervention by outside system. Policy lets users define what to do under what circumstances, but it does not specify a desired outcome.

- Intent is a higher-level declarative policy that operates at the level of a network and services it provides, not individual devices. It is used to define outcomes and high-level operational goals, without the need to enumerate specific events, conditions, and actions. Which algorithm or rules to apply can be automatically "learned/derived from intent" by the intent system. In the context of autonomic networking, ideally, intent is rendered by the network itself; also the dissemination of intent across the network and any required coordination between nodes is resolved by the network itself without the need for outside systems.

One analogy to capture the difference between policy and intent systems is that of Expert Systems and Learning Systems in the field of Artificial Intelligence. Expert Systems operate on knowledge bases with rules that are supplied by users. They are able to make automatic inferences based on those rules, but are not able to "learn" on their own. Learning Systems (popularized by deep learning and neural networks), on the other hand, are able to learn without depending on user programming. However, they do require a learning or training phase and explanations of actions that the system actually takes provide a different set of challenges.

5. Principles

The following operating principles allow characterizing the intent-based/-driven/-defined nature of a system.

1. Single Source of Truth (SSoT) and Single Version/View of Truth (SVoT). The SSoT is an essential component of an intent-based system as it enables several important operations. The set of validated intent expressions is the system's SSoT. SSoT and the records of the operational states enable comparing the intented state and actual state of the system and determining drift between them. SsoT and the drift information provide the basis for corrective actions. If the intent-based is equipped with prediction capabilities or means, it can further develop strategies to anticipate, plan and pro-actively act on the diverging trends with the aim to minimize their impact. Beyond providing a means for consistent system operation, SSoT also allows for better traceability to validate if/how the initial intent and associated business goals have been properly met, to evaluate the impacts of changes in the intent parameters and impacts and effects of the events occurring in the system.

Single Version (or View) of Truth derives from the SSoT and can be used to perform other operations such as query, poll or filter the measured and correlated information to create so-called "views". These views can serve the operators and/or the users of the intent-based system. To create intents as single sources of truth, the intent-based system must follow well-specified and well-documented processes and models. In other contexts [Lenrow15], SSoT is also referred to as the invariance of the intent.

2. One touch but not one shot. In an ideal intent-based system, the user expresses its intents in one form or another and then the system takes over all subsequent operations (one touch). A zero-touch approach could also be imagined in case where the intent-based system has the capabilities or means to recognize intentions in any form of data. However, the zero- or one-touch approach should not be mistaken the fact that reaching the state of a well-formed and valid intent expression is not a one-shot process. On the contrary, the interfacing between the user and the intent-based system could be designed as an interactive and interactive process. Depending on the level of abstraction, the intent expressions will initially contain more or less implicit parts, and unprecise or unknown parameters and constraints. The role of the intent-based system is to parse, understand and refine the intent expression to reach a well-formed and valid intent expression that can be further used by the system for the fulfillment and assurance operations. An intent refinement process could use a combination of iterative steps involving the user to validate the proposed refined intent and to ask the user for clarifications in case some parameters or variables could not be deduced or learned by the means of the system itself. In addition, the Intent-Based System will need to moderate between conflicting intent, helping users to properly choose between intent alternatives that may have different ramifications.

3. Autonomy and Oversight. A desirable goal for an intent-based system is to offer a high degree of flexibility and freedom on both the user side and system side, e.g. by giving the user the ability to express intents using its own terms, by supporting different forms of expression of intents and being capable of refining the intent expressions to well-formed and exploitable expressions. The dual principle of autonomy and oversight allows to operate a system that will have the necessary levels of autonomy to conduct its tasks and operations without requiring intervention of the user and taking its own decisions (within its areas of concern and span of control) as how to perform and meet the user expiations in terms of performance and quality, while at the same time providing the proper level of oversight to satisfy

the user requirements for reporting and escalation of relevant information. to be added: description for feedback, reporting, guarantee scope (check points, guard rails, dynamically provisioned, context rich, regular operation vs. exception/ abnormal, information zoom in-out, and link to SVoT. Accountable for decisions and efficiency, late binding (leave it to the system where to place functionality, how to accomplish certain goals).

4. Learning. An intent-based system is a learning system. By contrast to imperative type of system, such as Event-Condition-Action policy rules, where the user define beforehand the expected behavior of the system to various event and conditions, in an intent-based system, the user only declare what the system should achieve and not how to achieve these goals. There is thus a transfer of reasoning/rationality from the human (domain knowledge) to the system. This transfer of cognitive capability implies also the availability in the intent-based system of capabilities or means for learning, reasoning and knowledge representation and management. The learning abilities of an intent-based systems can apply to different tasks such as optimization of the intent rendering or intent refinement processes. The fact that an intent-based system is a continuously evolving system creates the condition for continuous learning and optimization. Other cognitive capabilities such as planning can also be leveraged in an intent-based system to anticipate or forecast future system state and response to changes in intents or network conditions and thus elaboration of plans to accommodate the changes while preserving system stability and efficiency in a trade-off with cost and robustness of operations. Cope with unawareness of users (smart recommendations).

5. Explainability. Need expressive network capabilities, requirements and constraints to be able to compose/decompose intents, map user's expectation to system capabilities. capability exposure. not just automation of steps that need to be taken, but of bridging the semantic gap between "intent" and actionable levels of instructions Context: multi providers, need discovery and semantic descriptions Explainability: why is a network doing what it is doing

6. Abstraction - users do not need to be concerned with how intent is achieved

Additional principles will be described in future revision of this document addressing aspects such as: Target groups not individual devices, agnostic to implementation details, user-friendly, user vocabulary vs. language of the device/network, explainability, validation and troubleshooting, how to resolve and point out conflicts (between intents), reconcile the reality of what is possible with the fiction of what the user would want, "moderate", awareness of operating within system boundaries, outcome-driven ((what not how, for the user);(what and how/where, for the operator).not imperative/instruction based.).

The above principles will be further used to understand implications on the design of intent-based systems and their supporting architecture, and derive functional and operational requirements.

6. Lifecycle

```
          user related                    user data <-----<-+--------+
          data  +                          +                |        |
                |                          |                |        |
           +----v------+             +-----v-----+          |        |
           | recognize +---+   +-----+ generate  |          |        |
     user  +-----------+   |   |     +-----------+          |        |
     space                 |   |                            |        |
  +--------------------------------------------------------------------+
     system                |   |                            |        |
     space             +---v---v---+   +----------+   +-----+-----+  |
                       | translate <-->+ validate <---> recommend |  |
                       +-----+-----+   +----------+   +-----------+  |
                             |                                       |
                       +-----v-----+                                 |
                       | normalize |                                 |
                       +-----+-----+                                 |
                             |                                       |
                       +-----v-----+                                 |
                       | decompose |                                 |
                       +-----+-----+                                 |
                             |                                       |
                      +------v------+                                |
                      | communicate |                                |
                      +------+------+                                |
      preparation            |                                       |
      phase                  |                                       |
   +-------------------------------------------------------------------+
      operation              |                                       |
      phase            +-----v----+                                  |
                       | fullfill |                                  |
                       +-----+----+                                  |
                             |                                       |
                        +----v----+     +--------+                   |
                        | observe +-----> report +-------------------+
                        +----+----+     +--------+
                             |
                        +----v---+
                        | assure |
                        +--------+
```

Figure 1: Intent Lifecycle

The intent lifecycle is work in progress. Todo: Intent attributes, intent states. Distinguish flow from users to network, and from network to user.

Another version is depicted below. Some of the aspects worth highlighting:

- o There is a distinction between the traditional network operations realm on one hand (providing fulfillment and assurance functions), and the user realm on the other hand (who needs to give direction to the network and be given information and reports regarding how the network is doing. Intent-Based Systems provide the link between those two realms.

- o There is a genuine distinction between fulfillment operations, used to drive intent into the network, orchestrate configuration operations etc, aand assurance operations intended to gain a sense of whether the network is performing as intended.

```
          User Space   :       Translation / IBS        :   Network Ops
                       :            Space               :       Space
                       :                                :
         +---------+   :  +----------+   +-----------+  :   +---------+
 Fulfill |recognize| ---> |translate/|-->|learn/plan/| ---> | config/ |
         |intent   | <--- |  refine  |   |  render   |  :   |provision|
         +---------+   :  +----------+   +-----^-----+  :   +---------+
                       :                       |        :         |
 ..............................................|..................|.....
                       :                  +----+---+    :         v
                       :                  |validate|    :   +----------+
                       :                  +----^---+ <------| monitor/ |
 Assure   +-------+    :  +---------+    +-----+---+    :   | observe/ |
          |report | <---- |abstract |<---| analyze | <------|  assure  |
          +-------+    :  +---------+    |aggregate|    :   +----------+
                       :                 +---------+    :
```

Figure 2: Intent Lifecycle 2

7. Intent-Based Networking - Functionality

Intent-Based Networking involves a wide variety of functions which can be roughly divided into two categories:

- o Intent Fulfillment provides functions and interfaces that allow users to communicate intent to the network, and that orchestrates the intent, i.e. that breaks down intent abstractions into lower-level network and device abstractions and performs or coordinates the configuration operations across the network.

o Intent Assurance provides functions and interfaces that allow users to validate and monitor that the network is indeed adhering to and complying with intent. Control plane or lower-level management operations can cause behavior that inadvertently conflicts with intent which was orchestrated earlier. Accordingly, "intent drift" may occur. Network operators need to be able to detect when such drift occurs, or is about to occur, and be provided with the necessary functions to resolve such conflicts. This can occur by either bringing the network back into compliance, or by articulating modifications to the original intent to moderate between conflicting interests.

The following sections provide a more comprehensive overview of those functions.

7.1. Intent Fulfillment

RBD

7.2. Intent Assurance

Ability to reason about system' state by employing closed-loop validation in the presence of an inevitable change is a fundamental property of an Intent Assurance part of an IBN system. Since service expectations are created during intent consumption and modeling phase, closed-loop intent vaidation should start immidiatelly, with the service instantiation. Telemetry consumed could then be enriched with an additional context and must always be processed in context of the Intent it has been instantiated. Direct relationship between the Intent and telemetry gathered enables correlation between changes in states and the Intent and provides contextual base for reasoning about the changes.

8. Research Challenges

8.1. Intent Interfaces

One goal for intent-based systems is to have the system "infer" the intent of the user, rather than requiring the users to provide a precise and complete set of instructions. Instead of forcing users to speak the language of the system, the system should be able to adapt to the needs of the user.

This requires new ways of interacting with users. An intent interface may no longer necessarily involve an interface or API with a clearly defined syntax and set of parameters. Instead, it may apply alternative styles, for example of iterative interrogation- or interview-style interfaces in which the system requests additional information from the user as needed to provide clarification, to select between alternatives, to refine intent.

8.2. Explanation Component

In an Intent-Based System, some of the actions taken by the network or behavior observed may be difficult to understand, analogous to deep learning systems which may have difficulty explaining their actions. In a networking environment, this can create some problems of its own, such as ensuring that the system is indeed functioning correctly and not compromised, necessary to give network providers the confidence that the Intent-Based Systems can indeed be relied on in business-critical applications.

8.3. IBN Metrics to Guide Desired Outcomes

As Intent-Based Networks are driven by desired outcomes, how to assess the quality of expected outcomes becomes critical. Corresponding metrics and evaluation functions become the basis by which IBNs can choose between different alternatives, and assess their ability to "learn" and make progress.

9. Items for Discussion

Arguably, given the popularity of the term intent, its use could be broadened to encompass also known concepts ("intent-washing"). For example, it is conceivable to introduce intent-based terms for various concepts that, although already known, are related to the context of intent. Each of those terms could then designate an intent subcategory, for example:

- o Operational Intent: defines intent related to operational goals of an operator; corresponds to the original "intent" term.
- o Rule Intent: a synonym for policy rules regarding what to do when certain events occur.
- o Service intent: a synonym for customer service model [RFC8309].
- o Flow Intent: A synonym for a Service Level Objective for a given flow.

Whether to do so is an item for discussion by the Research Group.

10. IANA Considerations

Not applicable

11. Security Considerations

Not applicable

12. References

12.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

12.2. Informative References

[Boutaba07]
Boutaba, R. and I. Aib, "Policy-Based Management: A Historical perspective. Journal of Network and Systems Management (JNSM), Springer, Vol. 15 (4).", December 2007.

[eTOM] TMForum, "GB 921 Business Process Framework, Release 17.0.1.", February 2018.

[I-D.ietf-teas-te-service-mapping-yang]
Lee, Y., Dhody, D., Ceccarelli, D., Tantsura, J., Fioccola, G., and Q. Wu, "Traffic Engineering and Service Mapping Yang Model", draft-ietf-teas-te-service-mapping-yang-01 (work in progress), March 2019.

[Lenrow15]
Lenrow, D., "Intent As The Common Interface to Network Resources, Intent Based Network Summit 2015 ONF Boulder: IntentNBI", February 2015.

[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>.

[RFC8299] Wu, Q., Ed., Litkowski, S., Tomotaki, L., and K. Ogaki, "YANG Data Model for L3VPN Service Delivery", RFC 8299, DOI 10.17487/RFC8299, January 2018, <https://www.rfc-editor.org/info/rfc8299>.

[RFC8309] Wu, Q., Liu, W., and A. Farrel, "Service Models Explained", RFC 8309, DOI 10.17487/RFC8309, January 2018, <https://www.rfc-editor.org/info/rfc8309>.

[RFC8345] Clemm, A., Medved, J., Varga, R., Bahadur, N., Ananthakrishnan, H., and X. Liu, "A YANG Data Model for Network Topologies", RFC 8345, DOI 10.17487/RFC8345, March 2018, <https://www.rfc-editor.org/info/rfc8345>.

[Sloman94]
Sloman, M., "Policy Driven Management for Distributed Systems. Journal of Network and Systems Management (JNSM), Springer, Vol. 2 (4).", December 1994.

[Strassner03]
Strassner, J., "Policy-Based Network Management. Elsevier.", 2003.

[TR523] Foundation, O. N., "Intent NBI - Definition and Principles. ONF TR-523.", October 2016.

Authors' Addresses

Alexander Clemm
Futurewei
2330 Central Expressway
Santa Clara, CA 95050
USA

Email: ludwig@clemm.org

Laurent Ciavaglia
Nokia
Route de Villejust
Nozay 91460
FR

Email: laurent.ciavaglia@nokia.com

Lisandro Zambenedetti Granville
Federal University of Rio Grande do Sul (UFRGS)
Av. Bento Goncalves
Porto Alegre 9500
BR

Email: granville@inf.ufrgs.br

Jeff Tantsura
Apstra, Inc.

Email: jefftant.ietf@gmail.com

Network Management Research Group   J. Nobre
Internet-Draft   University of Vale do Rio dos Sinos
Intended status: Informational   L. Granville
Expires: May 19, 2018   Federal University of Rio Grande do Sul
                                                                  A. Clemm
                                                                    Huawei
                                                        A. Gonzalez Prieto
                                                                    VMware
                                                         November 15, 2017

Autonomic Networking Use Case for Distributed Detection of SLA Violations
draft-irtf-nmrg-autonomic-sla-violation-detection-13

Abstract

This document describes an experimental use case for autonomic networking concerning monitoring of Service Level Agreements (SLAs). The use case aims to detect violations of SLAs in a distributed fashion, striving to optimize and dynamically adapt the autonomic deployment of active measurement probes in a way that maximizes the likelihood of detecting service level violations with a given resource budget to perform active measurements, and is able to do so without any outside guidance or intervention.

This document is a product of the IRTF Network Management Research Group (NMRG). It is published for informational purposes.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The Internet has been growing dramatically in terms of size, capacity, and accessibility in the last years. Communication requirements of distributed services and applications running on top of the Internet have become increasingly demanding. Some examples are real-time interactive video or financial trading. Providing such services involves stringent requirements in terms of acceptable latency, loss, or jitter.

Performance requirements lead to the articulation of Service Level Objectives (SLOs) which must be met. Those SLOs are part of Service Level Agreements (SLAs) that define a contract between the provider and the consumer of a service. SLOs, in effect, constitute a service level guarantee that the consumer of the service can expect to receive (and often has to pay for). Likewise, the provider of a service needs to ensure that the service level guarantee and associated SLOs are met. Some examples of clauses that relate to service level objectives can be found in [RFC7297]).

Violations of SLOs can be associated with significant financial loss, which can by divided into two categories. For one, there is the loss that can be incurred by the user of a service when the agreed service levels are not provided. For example, a financial brokerage’s stock orders might suffer losses when it is unable to execute stock transactions in a timely manner. An electronic retailer may lose customers when their online presence is perceived by customers as sluggish. An online gaming provider may not be able to provide fair access to online players, resulting in frustrated players who are lost as customers. In each case, the failure of a service provider to meet promised service level guarantees can have a substantial financial impact on users of the service. By the same token, there is the loss that is incurred by the provider of a service who is unable to meet promised service level objectives. Those losses can take several forms, such as penalties for not meeting the service and, in many cases more important, loss of revenue due to reduced customer satisfaction. Hence, service level objectives are a key concern for the service provider. In order to ensure that SLOs are not being violated, service levels need to be continuously monitored at the network infrastructure layer in order to know, for example, when mitigating actions need to be taken. To that end, service level measurements must take place.

Network measurements can be performed using active or passive measurement techniques. In passive measurements, production traffic is observed and no monitoring traffic is created by the measurement process itself. That is, network conditions are checked in a non intrusive way. In the context of IP Flow Information eXport (IPFIX), several documents were produced that define how to export data associated with flow records, i.e. data that is collected as part of passive measurement mechanisms, generally applied against flows of production traffic (e.g., [RFC7011]). In addition, it would be possible to collect real data traffic (not just summarized flow records) with time-stamped packets, possibly sampled (e.g., per [RFC5474], as a means of measuring and inferring service levels. Active measurements, on the other hand, are more intrusive to the network in the sense that it involves injecting synthetic test traffic into the network to measure network service levels, as opposed to simply observing production traffic. The IP Performance Metrics (IPPM) WG produced documents that describe active measurement mechanisms, such as: One-Way Active Measurement Protocol (OWAMP) [RFC4656], Two-Way Active Measurement Protocol (TWAMP) [RFC5357], and

Cisco Service Level Assurance Protocol (SLA) [RFC6812]. In addition, there are some mechanisms that do not cleanly fit into either active or passive categories, such as Performance and Diagnostic Metrics Destination Option (PDM) techniques [RFC8250].

Active measurement mechanisms offer a high level of control of what and how to measure. They do not require inspecting production traffic. Because of this, active measurements usually offer better accuracy and privacy than passive measurement mechanisms. Traffic encryption and regulations that limit the amount of payload inspection that can occur are non-issues. Furthermore, active measurement mechanisms are able to detect end-to-end network performance problems in a fine-grained way (e.g., simulating the traffic that must be handled considering specific Service Level Objectives - SLOs). As a result, active measurements are often preferred over passive measurement for SLA monitoring. Measurement probes must be hosted in network devices and measurement sessions must be activated to compute the current network metrics (e.g., considering those described in [RFC4148]). This activation should be dynamic in order to follow changes in network conditions, such as those related with routes being added or new customer demands.

While offering many advantages, active measurements are expensive in terms of network resource consumption. Active measurements generally involve measurement probes that generate synthetic test traffic that is directed at a responder. The responder needs to timestamp test traffic it receives and reflect it back to the originating measurement probe. The measurement probe subsequently processes the returned packets along with time stamping information in order to compute service levels. Accordingly, active measurements consume substantial CPU cycles as well as memory of network devices to generate and process test traffic. In addition, synthetic traffic increases network load. Active measurements thus compete for resources with other functions, including routing and switching.

The resources required and traffic generated by the active measurement sessions are to a large part a function of the number of measured network destinations. (In addition, the amount of traffic generated for each measurement plays a role, which in turn influences the accuracy of the measurement.) The more destinations are being measured, the larger the amount of resources consumed and traffic needed to perform the measurements. Thus, to have a better monitoring coverage it is necessary to deploy more sessions which consequently increases consumed resources. Otherwise, enabling the observation of just a small subset of all network flows can lead to an insufficient coverage.

Furthermore, while some end-to-end service levels can be determined by adding up the service levels observed across different path segments, the same is not true for all service levels. For example, the end-to-end delay or packet loss from a node A to a node C routed via a node B can often be computed simply by adding delays (or loss) from A to B, and B to C. This allows to decompose a large set of end-to-end measurements into a much smaller set of segment measurements. However, end-to-end jitter and (for example) Mean Opinion Scores cannot be decomposed as easily and, for higher accuracy, must be measured end-to-end.

Hence, the decision how to place measurement probes becomes an important management activity. The goal is to obtain maximum benefits of service level monitoring with a limited amount of measurement overhead. Specifically, the goal is to maximize the number of service level violations that are detected with a limited amount of resources.

The use case and the solution approach described in this document address an important practical issue. They are intended to provide a basis for further experimentation to lead into solutions for wider deployment. This document represents the consensus of the IRTF’s Network Management Research Group (NMRG). It was discussed extensively and received three separate in-depth reviews.

2. Definitions and Acronyms

Active Measurements: Techniques to measure service levels that involve generating and observing synthetic test traffic

Passive Measurements: Techniques used to measure service levels based on observation of production traffic

AN: Autonomic Network; a network containing exclusively autonomic nodes, requiring no configuration and deriving all required information through self-knowledge, discovery, or intent.

Autonomic Service Agent (ASA): An agent implemented on an autonomic node that implements an autonomic function, either in part (in the case of a distributed function, as in the context of this document), or whole.

Measurement Session: A communications association between a Probe and a Responder used to send and reflect synthetic test traffic for active measurements

Probe: The source of synthetic test traffic in an active measurement

Responder: The destination for synthetic test traffic in an active measurement

SLA: Service Level Agreement

SLO: Service Level Objective

P2P: Peer-to-Peer

(Note: definitions of AN and ASA are borrowed from [RFC7575]).

3. Current Approaches

The current best practice in feasible deployments of active measurement solutions to distribute the available measurement sessions along the network consists in relying entirely on the human administrator expertise to infer which would be the best location to activate such sessions. This is done through several steps. First, it is necessary to collect traffic information in order to grasp the traffic matrix. Then, the administrator uses this information to infer which are the best destinations for measurement sessions. After that, the administrator activates sessions on the chosen subset of destinations considering the available resources. This practice, however, does not scale well because it is still labor intensive and error-prone for the administrator to determine which sessions should be activated given the set of critical flows that needs to be measured. Even worse, this practice completely fails in networks whose critical flows are too short in time and dynamic in terms of traversing network path, like in modern cloud environments. That is so because fast reactions are necessary to reconfigure the sessions and administrators are just not quick enough in computing and activating the new set of required sessions every time the network traffic pattern changes. Finally, the current active measurements practice usually covers only a fraction of the network flows that should be observed, which invariably leads to the damaging consequence of undetected SLA violations.

4. Use Case Description

The use case involves a service level provider who needs to monitor the network to detect service level violations using active service level measurements, and wants to be able to do so with minimal human intervention. The goal is to conduct the measurements in an effective manner maximizing the percentage of detected service level violations. The service level provider has a bounded resource budget with regards to measurements that can be performed, specifically, with regards to the number of measurements that can be conducted concurrently from any one network device, and possibly with regards to the total amount of measurement traffic on the network. However, while at any one point in time the number of measurements conducted is limited, it is possible for a device to change which destinations to measure over time. This can be exploited to achieve a balance of eventually covering all possible destinations using a reasonable amount of "sampling" where measurement coverage of a destination cannot be continuous. The solution needs to be dynamic and be able to cope with network conditions which may change over time. The solution should also be embeddable inside network devices that control the deployment of active measurement mechanisms.

The goal is to conduct the measurements in a smart manner that ensures that the network is broadly covered and the likelihood of detecting service level violations is maximized. In order to maximize that likelihood, it is reasonable to focus measurement resources on destinations that are more likely to incur a violation, while spending less resources on destinations that are more likely to be in compliance. In order to do so, there are various aspects that can be exploited, including past measurements (destinations close to a service level threshold requiring more focus than destinations further from it), complementation with passive measurements such as flow data (to identify network destinations that are currently popular and critical), and observations from other parts of the network. In addition, measurements can be coordinated among different network devices to avoid hitting the same destination at the same time and to be able to share results that may be useful in future probe placement.

Clearly, static solutions will have severe limitations. At the same time, human administrators cannot be in the loop for continuous dynamic measurement probe reconfigurations. Accordingly, an automated or, ideally, autonomic solution is needed in which network measurements are automatically orchestrated and dynamically reconfigured from within the network. This can be accomplished using an autonomic solution that is distributed, using Autonomic Service Agents that are implemented on nodes in the network.

5. A Distributed Autonomic Solution

The use of Autonomic Networking (AN) [RFC7575] can help such detection through an efficient activation of measurement sessions. Such an approach, along with a detailed assessment confirming its viability, has been described [P2PBNM-Nobre-2012]. The problem to be solved by AN in the present use case is how to steer the process of Measurement Session activation by a complete solution that sets all necessary parameters for this activation to operate efficiently, reliably and securely, with no required human intervention other than setting overall policy.

When a node first comes online, it has no information about which measurements are more critical than others. In the absence of information about past measurements and information from measurement peers, it may start with an initial set of measurement sessions, possibly randomly seeding a set of starter measurements, perhaps taking a round robin approach for subsequent measurement rounds. However, as measurements are collected, a node will gain increasing information that it can utilize to refine its strategy of selecting measurement targets going forward. For one, it may take note of which targets returned measurement results very close to service level thresholds that may therefore require closer scrutiny compared to others. Second, it may utilize observations that are made by its measurement peers in order to conclude which measurement targets may be more critical than others, and in order to ensure that proper overall measurement coverage is obtained (so that not every node incidentally measure the same targets, while other targets are not measured at all).

We advocate for embedding Peer-to-Peer (P2P) technology in network devices in order to conduct the Measurement Session activation decisions using autonomic control loops. Specifically, we advocate for network devices to implement an autonomic function to monitor service levels for violations of service level objectives, determining which Measurement Sessions to set up at any given point in time based on current and past observations of the node, and of other peer nodes.

By performing these functions locally and autonomically on the device itself, which measurements to conduct can be modified quickly based on local observations while taking local resource availability into account. This allows a solution to be more robust and react more dynamically to rapidly changing service levels than a solution that has to rely on central coordination. However, in order to optimize decisions which measurements to conduct, a node will need to communicate with other nodes. This allows a node to take into account other nodes’ observations in addition to its own in its decisions.

For example, remote destinations whose observed service levels are on the verge of violating stated objectives may require closer monitoring than remote destinations that are comfortably within a range of tolerance. It also allows nodes to coordinate their probing decisions to collectively achieve the best possible measurement coverage. As the amount of resources available for monitoring and for exchange of measurement data and coordination with other nodes are limited, a node may further be interested in identifying other nodes whose observations are most similar to and correlated with its own. This helps a node prioritize and guide with which other nodes to primarily coordinate and exchange data with. All of this requires the use of a P2P overlay.

A P2P overlay is essential for several reasons:

- o It makes it possible for nodes (respectively Autonomic Service Agents that are deployed on those nodes) in the network to autonomically set up Measurement Sessions, without having to rely on central management system or controller to perform configuration operations associated with configuring measurement probes and responders.

- o It facilitates the exchange of data between different nodes to share measurement results so that each node can refine its measurement strategy based not just its own observations, but observations from its peers.

- o It allows nodes to coordinate their measurements to obtain the best possible test coverage and avoid measurements that have a very low likelihood of detecting service level violations.

The provisioning of the P2P overlay should be transparent for the network administrator. An Autonomic Control Plane such as defined in [I-D.anima-autonomic-control-plane] provides an ideal candidate for the P2P overlay to run on.

An autonomic solution for the distributed detection of SLA violations provide several benefits. First, efficiency: this solution should optimize the resource consumption and avoid resource starvation on the network devices. A device that is "self-aware" of its available resources will be able to adjust measurement activities rapidly as needed, without requiring a separate control loop involving resource monitoring by an external system. Secondly, placing logic where to conduct measurements in the node enables rapid control loops in which devices are able to react instantly to observations and adjust their measurement strategy. For example, a device could decide to adjust the amount of synthetic test traffic being sent during the measurement itself depending on results observed so far on this and on other concurrent measurement sessions. As a result, the solution could decrease the time necessary to detect SLA violations. Adaptivity features of an autonomic loop could capture faster the network dynamics than an human administrator and even a central controller. Finally, the solution could help to reduce the workload of human administrator, or, at least, to avoid their need to perform operational tasks.

In practice, these factors combine to maximize the likelihood of SLA violations being detected while operating within a given resource budget, allowing to conduct a continuous measurement strategy that takes into account past measurement results, observations of other measures such as link utilization or flow data, sharing of measurement results between network devices, and coordinating future measurement activities among nodes. Combined this can result in efficient measurement decisions that achieve a golden balance between broad network coverage and honing in on service level "hot spots".

6. Intended User Experience

The autonomic solution should not require any human intervention in the distributed detection of SLA violations. By virtue of the solution being autonomic, human users will not have to plan which measurements to conduct in a network, often a very labor intensive task today that requires detailed analysis of traffic matrices and network topologies and is not prone to easy dynamic adjustment. Likewise, they will not have to configure measurement probes and responders.

There are some ways in which a human administrator may still interact with the solution. For one, the human administrator will of course be notified and obtain reports about service level violations that are observed. Second, a human administrator may set a policies regarding how closely to monitor the network for service level violations and how many resources to spend. For example, an administrator may set a resource budget that is assigned to network devices for measurement operations. With that given budget, the number of SLO violations that are detected will be maximized. Alternatively, an administrator may set a target for the percentage of SLO violations that must be detected, i.e. a target for the ratio between the number of detected SLO violations, and the number of total SLO violations that are actually occurring (some of which might go undetected). In that case, the solution will aim to minimize the resources spent (i.e. the amount of test traffic and Measurement Sessions) that are required to achieve that target.

7. Implementation Considerations

The active measurement model assumes that a typical infrastructure will have multiple network segments and Autonomous Systems (ASs), and a reasonably large number of routers. It also considers that multiple SLOs can be in place at a given time. Since interoperability in a heterogenous network is a goal, features found on different active measurement mechanisms (e.g. OWAMP, TWAMP, and IPSLA) and device programability interfaces (such as Juniper's Junos API or Cisco's Embedded Event Manager) could be used for the implementation. The autonomic solution should include and/or reference specific algorithms, protocols, metrics and technologies for the implementation of distributed detection of SLA violations as a whole.

Finally, it should be noted that there are multiple deployment scenarios, including deployment scenarios that involve physical devices hosting autonomic functions, or virtualized infrastructure hosting the same. Co-deployment in conjunction with Virtual Network Functions (VNF) is a possibility for further study.

7.1. Device Based Self-Knowledge and Decisions

Each device has self-knowledge about the local SLA monitoring. This could be in the form of historical measurement data and SLOs. Besides that, the devices would have algorithms that could decide which probes should be activated in a given time. The choice of which algorithm is better for a specific situation would be also autonomic.

7.2. Interaction with other devices

Network devices should share information about service level measurement results. This information can speed up the detection of SLA violations and increase the number of detected SLA violations. For example, if one device detects that a remote destination is in danger of violating an SLO, other devices may conduct additional measurements to the same destination or other destinations in its proximity. For any given network device, the exchange of data may be more important with some devices (for example, devices in the same network neighborhood, or devices that are "correlated" by some other means) than with others. The definition of network devices that exchange measurement data, i.e., management peers, creates a new topology. Different approaches could be used to define this topology (e.g., correlated peers [P2PBNM-Nobre-2012]). To bootstrap peer selection, each device should use its known endpoints neighbors (e.g., FIB and RIB tables) as the initial seed to get possible peers. It should be noted that a solution will benefit if topology information and network discovery functions are provided by the underlying autonomic framework. A solution will need to be able to discover measurement peers as well as measurement targets, specifically measurement targets that support active measurement responders and which will be able to respond to measurement requests and reflect measurement traffic as needed.

8. Comparison with current solutions

There is no standardized solution for distributed autonomic detection of SLA violations. Current solutions are restricted to ad hoc scripts running on a per node fashion to automate some administrator’s actions. There are some proposals for passive probe activation (e.g., DECON and CSAMP), but without the focus on autonomic features.

9. Related IETF Work

The following paragraphs discuss related IETF work and are provided for reference. This section is not exhaustive, rather it provides an overview of the various initiatives and how they relate to autonomic distributed detection of SLA violations.

1. [LMAP]: The Large-Scale Measurement of Broadband Performance Working Group aims at the standards for performance management. Since their mechanisms also consist in deploying measurement probes the autonomic solution could be relevant for LMAP specially considering SLA violation screening. Besides that, a solution to decrease the workload of human administrators in service providers is probably highly desirable.

2. [IPFIX]: IP Flow Information EXport (IPFIX) aims at the process of standardization of IP flows (i.e., netflows). IPFIX uses measurement probes (i.e., metering exporters) to gather flow data. In this context, the autonomic solution for the activation of active measurement probes could be possibly extended to address also passive measurement probes. Besides that, flow information could be used in the decision making of probe activation.

3. [ALTO]: The Application Layer Traffic Optimization Working Group aims to provide topological information at a higher abstraction layer, which can be based upon network policy, and with application-relevant service functions located in it. Their work could be leveraged for the definition of the topology regarding the network devices which exchange measurement data.

10. Acknowledgements

We wish to acknowledge the helpful contributions, comments, and suggestions that were received from Mohamed Boucadair, Brian Carpenter, Hanlin Fang, Bruno Klauser, Diego Lopez, Vincent Roca, and Eric Voit. In addition, we thank Diego Lopez, Vincent Roca, and Brian Carpenter for their detailed reviews.

11. IANA Considerations

This memo includes no request to IANA.

12. Security Considerations

Security of the solution hinges on the security of the network underlay, i.e. the Autonomic Control Plane. If the Autonomic Control Plane were to be compromised, an attacker could undermine the effectiveness of measurement coordination by reporting fraudulent measurement results to peers. This would cause measurement probes to be deployed in an ineffective manner that would increase the likelihood that violations of service level objectives go undetected.

Likewise, security of the solution hinges on the security of the deployment mechanism for autonomic functions, in this case, the autonomic function that conducts the service level measurements. If an attacker were able to hijack an autonomic function, it could try to exhaust or exceed the resources that should be spent on autonomic measurements in order to deplete network resources, including network bandwidth due to higher-than-necessary volumes of synthetic test traffic generated by measurement probes. Again, it could also lead to reporting of misleading results, among other things resulting in non-optimal selection of measurement targets and in turn an increase in the likelihood that service level violations go undetected.

13. Informative References

[draft-anima-boot]
Pritikin, M., Richardson, M., Behringer, M., Bjarnason, S., and K. Watsen, "draft-ietf-anima-bootstrapping-keyinfra", draft-ietf-anima-bootstrapping-keyinfra-08 (work in progress), October 2017.

[I-D.anima-autonomic-control-plane]
Behringer, M., Eckert, T., and S. Bjarnason, "An Autonomic Control Plane", draft-ietf-anima-autonomic-control-plane-12 (work in progress), October 2017.

[P2PBNM-Nobre-2012]
Nobre, J., Granville, L., Clemm, A., and A. Gonzalez Prieto, "Decentralized Detection of SLA Violations Using P2P Technology, 8th International Conference Network and Service Management (CNSM)", 2012, <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6379997>.

[RFC4148] Stephan, E., "IP Performance Metrics (IPPM) Metrics Registry", BCP 108, RFC 4148, DOI 10.17487/RFC4148, August 2005, <https://www.rfc-editor.org/info/rfc4148>.

[RFC4656] Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., and M. Zekauskas, "A One-way Active Measurement Protocol (OWAMP)", RFC 4656, DOI 10.17487/RFC4656, September 2006, <https://www.rfc-editor.org/info/rfc4656>.

[RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, DOI 10.17487/RFC5357, October 2008, <https://www.rfc-editor.org/info/rfc5357>.

[RFC5474] Duffield, N., Ed., Chiou, D., Claise, B., Greenberg, A., Grossglauser, M., and J. Rexford, "A Framework for Packet Selection and Reporting", RFC 5474, DOI 10.17487/RFC5474, March 2009, <https://www.rfc-editor.org/info/rfc5474>.

[RFC6812] Chiba, M., Clemm, A., Medley, S., Salowey, J., Thombare, S., and E. Yedavalli, "Cisco Service-Level Assurance Protocol", RFC 6812, DOI 10.17487/RFC6812, January 2013, <https://www.rfc-editor.org/info/rfc6812>.

[RFC7011] Claise, B., Ed., Trammell, B., Ed., and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information", STD 77, RFC 7011, DOI 10.17487/RFC7011, September 2013, <https://www.rfc-editor.org/info/rfc7011>.

[RFC7297] Boucadair, M., Jacquenet, C., and N. Wang, "IP Connectivity Provisioning Profile (CPP)", RFC 7297, DOI 10.17487/RFC7297, July 2014, <https://www.rfc-editor.org/info/rfc7297>.

[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>.

[RFC8250] Elkins, N., Hamilton, R., and M. Ackermann, "IPv6 Performance and Diagnostics Metrics (PDM) Destination Option", RFC 8250, October 2017.

Authors’ Addresses

Jeferson Campos Nobre
University of Vale do Rio dos Sinos
Porto Alegre
Brazil

Email: jcnobre@unisinos.br

Lisandro Zambenedetti Granvile
Federal University of Rio Grande do Sul
Porto Alegre
Brazil

Email: granville@inf.ufrgs.br

Alexander Clemm
Huawei
Santa Clara, California
USA

Email: ludwig@clemm.org

Alberto Gonzalez Prieto
VMware
Palo Alto, California
USA

Email: agonzalezpri@vmware.com

Network Management Research Group
Internet-Draft
Intended status: Informational
Expires: January 9, 2020

M-S. Kim
ETRI
Y-H. Han
KoreaTech
Y-G. Hong
ETRI

July 8, 2019

Intelligent Reinforcement-learning-based Network Management
draft-kim-nmrg-rl-05

Abstract

This document presents intelligent network management based on Artificial Intelligent (AI) such as reinforcement-learning approaches. In a heterogeneous network, intelligent management with Artificial Intelligent should usually provide real-time connectivity, the type of network management with the quality of real-time data, and transmission services generated by an application service. With that reason intelligent management system is needed to support real-time connection and protection through efficient management of interfering network traffic for high-quality network data transmission in the both cloud and IoE network systems. Reinforcement-learning is one of the machine learning algorithms that can intelligently and autonomously provide to management systems over a communication network. Reinforcement-learning has developed and expanded with deep learning technique based on model-driven or data-driven technical approaches so that these trendy techniques have been widely to intelligently attempt an adaptive networking models with effective strategies in environmental disturbances over variety of networking areas. For Network AI with the intelligent and effective strategies, intent-based network (IBN) can be also considered to continuously and automatically evaluate network status under required policy for dynamic network optimization. The key element for the intent-based network is that it provides a verification of whether the represented network intent is implementable or currently implemented in the network. Additionally, this approach need to provide to take action in real time if the desired network state and actual state are inconsistent.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Reinforcement-learning for intelligently autonomous network management, in general, is one of the challengeable methods in a dynamic complex and cluttered network environments. With the intelligent approach needs the development of computational systems in a single or large distributed networking nodes, where these environments involve limited and incomplete knowledge.

The reinforcement-learning can become a challenge-able and effective technique to transfer and share information via the global environment, as it does not require a priori-knowledge of the agent behavior or environment to accomplish its tasks [Megherbi]. Such a knowledge is usually acquired and learned repeatedly and autonomously by trial and error. The reinforcement-learning is also one of the machine learning techniques that will be adapted to the various networking environments for automatic networks [S.Jiang].

Deep-reinforcement-learning recently proposes has been extended from reinforcement-learning that can emerge as more powerful model-driven or data-driven model in a large state space, to overcome the classical behavior reinforcement-learning process. However, the classical reinforcement-learning slightly has a limitation to be adopted in networking areas, since the networking environments consist of significantly large and complex components in fields of routing configuration, optimization and system management, so that deep-reinforcement-learning can provide much more state information for learning process.[MS]

There are many different networking management problems to intelligently solve, such as connectivity, traffic management, fast Internet without latency and etc. Reinforcement-learning-based approaches can surely provide some of specific solutions with multiple cases against human operating capacities although it is a challengeable area due to a multitude of reasons such as large state space, complexity in the giving reward, difficulty in control actions, and difficulty in sharing and merging of the trained knowledge in a distributed memory node to be transferred over a communication network.[MS]

In addition, Intent-based network bridge to solve some of network problems and gaps between network business model and technical scheme. Intents should be applied to application service levels, security policies, compliance, operational processes, and other business needs. The network should constantly monitor and adjust to meet the intent in following the monitoring system. There are some of requirements to satisfy Intent-based network as following: (1) Transfer, (2) policy activation (automatically), (3) guarantee (Continuous monitoring and verification) [Cisco]. Through continuously monitoring with network data, we are able to collect network information and to analyze the collected information by artificial intelligent approach. If the analysis result shows that the new network configuration parameter needs to be changed or reconfigured by deriving the optimized value.

2. Conventions and Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Theoretical Approaches

3.1. Reinforcement-learning

Reinforcement-learning is an area of machine learning concerned with how software agents should take actions in an environment so as to maximize some notion of cumulative reward.[Wikipedia] The reinforcement-learning is normally used with a reward from centralized node (the global brain), and capable of autonomous acquirement and incorporation of knowledge. It is continuously self-improving and becoming more efficient as the learning process from an agent experience to optimize management performance for autonomous learning process.[Sutton][Madera]

3.2. Deep-reinforcement-learning

Some of advanced techniques using reinforcement-learning encounter and combine with deep-learning in neural networks that has made it possible to extract high-level features from raw data in compute vision [A Krizhevsky]. There are many challenges under the deep-learning models such as convolution neural network, recurrent neural network and etc., on the reinforcement-learning approach. The benefit of the deep learning applications is that lots of networking models, but the problematic issue is complex and cluttered networking structures used with large amounts of labelled training data.

Recently, the advances in training deep neural networks to develop a novel artificial agent, termed a deep Q-network (deep-reinforcement-learning network), can be used to learn successful policies directly from high-dimensional sensory inputs using end-to-end reinforcement learning [V.Mnih].

The deep-reinforcement-learning (deep Q-network) can provide more extended and powerful scenarios to build networking models with optimized action controls, huge system states and real-time-based reward function. Moreover, the technique has a significant advantage to set highly sequential data in a large model state space. [MS] In particular, the data distribution in reinforcement-learning is able to change as learning behaviors, that is a problem for deep learning approaches assumed by a fixed underlying distribution [V. Mnih].

3.3. Advantage Actor Critic (A2C)

Advantage Actor Critic is one of the intelligent reinforcement-learning models based on policy gradient model. The intelligent approach can optimize deep neural network controller in terms of reinforcement-learning algorithms, and show that parallel actor-learners have a stabilizing effect on training and they can be allowing all of the methods to successfully train neural network controllers [Volodymyr Mnih]. Even if the prior deep-reinforcement-learning algorithm with experience replay memory tremendously has performance in challenging of the control service domains, it still needs to use more memory and computational power due to off-policy learning methods. To make up for this algorithms, a new algorithm has appeared.

The Advantage Actor Critic (consisting of actor and critic) method would implement generalized policy iteration alternating between a policy evaluation and a policy improvement step. Actor is a policy-based method that can improve the current policy for available the best next action. Critic in the value-based approach can evaluate the current policy and reduce the variance by a bootstrapping method. It is more stable and effective algorithm than the pure policy-based gradient methods.[MS]

3.4. Asynchronously Advantage Actor Critic (A3C)

Asynchronously Advantage Actor Critic is the updated algorithm based on Advantage Actor Critic. The main algorithm concept is to run multiple environments in parallel to run the agent asynchronously instead of experience replay. The parallel environment reduces the correlation of agent's data and induces each agent to experience various states so that the learning process can become a stationary process. This algorithm is a beneficial and practical point of view since it allows learning performance even with a general multi-core CPU. In addition, it can be applied to continuous space as well as discrete action space, and also has the advantages of learning both feedforward and recurrent agent.[MS]

A3C algorithm is possibly a number of complementary improvement to the neural network architecture and it has been shown to accurately produce and estimate of Q-values by including separate streams for the state value and advantage in the network to improve both value-based and policy-based methods by making it easier for the network to represent feature coordinates [Volodymyr Mnih].

3.5. Intent-based Network (IBN)

ntent-based Network is a new technical approach that can adapt the network flexibly through configuration parameters derived from data analysis for network machine learning. Software-defined Networking (SDN) is a similar concept with Intent-based Network, however, Software-defined Networking has not yet tipped in the sector that relies on network automation. With the approach, network machine learning is integrated with network analysis, routing, wireless communications, and resource management. However, unlike the field of computer vision, which can easily acquire sufficient data, it is difficult to obtain data over a real network. Therefore, there are limitations to apply machine learning technique to network field with the data. Reinforcement Learning (RL) can diminish much attention and the importance of securing high-quality data, so that both concepts of reinforcement learning and intent-based network might solve the limitation and integrate a gap between network machine learning and network technique.

Intent-based network is also describing how to apply the setting values for network management/operation in a procedural way. For that reason, the approach is also the core of Intent processing that automatically interprets it and declares it declaratively. Even if the basic concepts of intent-based network reflects and to be announced regarding intent, there is no standardized form of Intent processing technology. While intent-based network has the advantage of providing a higher level of abstraction in network management/operation and providing ease of use, a more specific and clear definition of the technology is likely to be needed.

4. Reinforcement-learning-based process scenario

With a single agent or multiple agents trained for intelligent network management, a variety of training scenarios are possible, depending on how agents are interacted and how many models are linked to the agents. The followings are possible RL training scenarios for network management.

4.1. Single-agent with Single-model

This is the traditional scenario of training a single agent who tries to achieve one goal related to network management. It receives all of information and rewards from a network (or a simulated network), and decides its appropriate action for the current network status.

4.2. Multi-agents Sharing Single-model

In this scenario, multiple agents share a single model and a single goal linked to the model. But, each of them is connected to an independent part of network or an independent whole network, so that they receive different information and rewards from such an independent one. The multiple agents experience differently on their connected networks. However, it does not mean their training behavior for network management will diverge. Each of their experience is used to train the single model. This scenario is a kind of parallelized version of the traditional 'Single-Agent with Single-Model' scenario, which can speed-up the RL training process and stabilize the single model's behavior.

4.3. Adversarial Self-Play with Single-model

This scenario contains two interacting agents with inverse reward functions linked to a single model. This scenario makes an agent have the perfectly matched opposing agent: itself, and trains the agent to become increasingly more skilled for network management. Inverse rewards are used to punish the opposing agent when an agent receives as positive reward, and vice versa. The two agents are linked to a single model for network management, and the model are trained and stabilized while both agents interact in a conflicting manner.

4.4. Cooperative Multi-agents with Multiple-models

In this scenario, two or more interacting agents share a common reward function linked to multiple different models for network management. In this scenario, a common goal is set up and all agents are trained to achieve the goal together that is hard to be achieved alone. Usually, each agent has access only to partial information of network status and determines an appropriate action by using its own model. Each of actions will be independently taken in order to accomplish a management task and collaboratively achieve the common goal.

4.5. Competitive Multi-agents with Multiple-models

This scenario contains two or more interacting agents with diverse reward function linked to multiple different models. In this scenario, agents will compete with one another to obtain some limited set of network resources and try to achieve their own goal. In a network, there will be tasks that have different management objectives. This leads multi-objective optimization problems, which are generally difficult to solve analytically. This scenario is suitable for solving such a multi-objective optimization problem related to network management by allowing each agent solve a single-objective problem, but complete with each other.

5. Use Cases

5.1. Intelligent Edge-computing for Traffic Control using Deep-reinforcement-learning

Edge computing is a concept that allows data from a variety of devices to be directly analyzed at the site or near the data, rather than being sent to a centralized data center such as the cloud. As such, edge computing will support data flow acceleration by processing data with low latency in real-time. In addition, by supporting efficient data processing on large amounts of data that can be processed around the source, and internet bandwidth usage will be also reduced.

Deep-reinforcement-learning would be useful technique to improve system performance in an intelligent edge-controlled service system for fast response time, reliability and security. Deep-reinforcement-learning is model-free approach so that many algorithms such as DQN, A2C and A3C can be adopted to resolve network problems in time-sensitive systems.

5.2. Edge computing system in a field of Construction-site using Reinforcement-learning

In a construction site, there are many dangerous elements such as noisy, gas leak and vibration needed by alerts, so that real-time monitoring system to detect the alerts using machine learning techniques can provide more effective solution and approach to recognize dangerous construction elements.

Representatively, to monitor these elements CCTV (closed-circuit television) should be locally and continuously broadcasting in a situation of construction site. At that time, it is in-effective and wasteful even if the CCTV is constantly broadcasting unchangeable scenes in high definition. However, the streaming should be converted to high quality streaming data to rapidly show and defect the dangerous situation, when any alert should be detected due to the dangerous elements. To approach technically deep-reinforcement-learning can provide a solution to automatically detect these kinds of dangerous situations with prediction in an advance. It can also provide the transform data including with the high-rate streaming video and quickly prevent the other risks. Deep-reinforcement-learning is an important role to efficiently manage and monitor with the given dataset in real-time.

5.3. Deep-reinforcement-learning-based remote Control system over a software-defined network

With the nonlinear control system such as cyber physical system provides an unstable system environment with initial control state due to its nonlinear nature. In order to stably control the unstable initial state, the prior-complex mathematical control methods (Linear Quadratic Regulator, Proportional Integral Differential) are used for successful control and management, but these approaches are needed with difficult mathematical process and high-rate effort. Therefore, using deep-reinforcement-learning can surely provide more effective technical approach without difficult initial set of control states to be compared with the other methods.

The ultimate purpose of the reinforcement-learning is to interact with the environment and maximize the target reward value. Observing the state in the step and the action by the policy are performed, and the reward judge a value through the compensation given in the environment. Deep-reinforcement-learning using Convolutional Neural Network (CNN) can provide more performing learning process to make stable control and management.

As part of the system, it shows how the physical environment and the cyber environment interact with the reinforcement-learning module over a network. The actions to control the physical environment, delivered to the Enhanced Learning model based on DQN, transfer to data to the physical environment using networking communication tools as below.

```
      +-----Environment-----+            +---Control and Management---+
      .                     .            .                            .
      . +-----------------+ .  Network   +--------------+             .
      . . Physical System . .----------->. Cyber Module .             .
      . .                 . .<-----------.              .             .
      . +-----------------+ .            +--------------+             .
      .                     .            .      .          +--------+ .
      +---------------------+            .      .----------.RL Agent. .
                                         .                 +--------+ .
                                         +............................+
```

Figure 1: DRL-based Cyber Physical Management Control System

With the use-case, the reinforcement learning agent interacts with the physical remote device while exchanging network packets. The Software-defined network controller can manage the network traffic transmission, so that the system is naturally composed of a cyber environment and physical environment, and two environments closely and synchronously.[Ju-Bong]

For the intelligent traffic management in the system, software-defined networking for automation (basic concept for IBN) should be used to control and manage of connection between the cyber physical system and edge computing module. The intelligent approach consists of software that intelligently controls the network and technique that allows software to set up and control the network. The concept of can be centralized to control of network operation by software programming, centralizes switch/router control function based on existing hardware. It is possible to manage the network according to the requirements without the detailed network configuration.

In addition, software-defined networking switch is able to enable the network traffic control to be controlled and managed by software-based controllers. This approach is really similar with intent-based networking since both approaches can share the similar principle using software to run the network, however, intent-based networking offers an abstraction layer under the implemented policy and instruction across all the physical hardware within the infrastructure for automated networking. To achieve superior intent-based networking over a real network, the physical control system will be implemented to automatically manage and provide IoE edge smart traffic control service for high quality real time connection.

6. IANA Considerations

There are no IANA considerations related to this document.

7. Security Considerations

[TBD]

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

8.2. Informative References

[I-D.jiang-nmlrg-network-machine-learning]
Jiang, S., "Network Machine Learning", ID draft-jiang-nmlrg-network-machine-learning-02, October 2016.

[Megherbi]
"Megherbi, D. B., Kim, Minsuk, Madera, Manual., A Study of Collaborative Distributed Multi-Goal and Multi-agent based Systems for Large Critical Key Infrastructures and Resources (CKIR) Dynamic Monitoring and Surveillance, IEEE International Conference on Technologies for Homeland Security", 2013.

[Teiralbar]
"Megherbi, D. B., Teiralbar, A. Boulenouar, J., A Time-varying Environment Machine Learning Technique for Autonomous Agent Shortest Path Planning, Proceedings of SPIE International Conference on Signal and Image Processing, Orlando, Florida", 2001.

[Nasim] "Nasim ArianpooEmail, Victor C.M. Leung, How network monitoring and reinforcement learning can improve tcp fairness in wireless multi-hop networks, EURASIP Journal on Wireless Communications and Networking", 2016.

[Minsuk] "Dalila B. Megherbi and Minsuk Kim, A Hybrid P2P and Master-Slave Cooperative Distributed Multi-Agent Reinforcement Learning System with Asynchronously Triggered Exploratory Trials and Clutter-index-based Selected Sub goals, IEEE CIG Conference", 2016.

[April] "April Yu, Raphael Palefsky-Smith, Rishi Bedi, Deep Reinforcement Learning for Simulated Autonomous Vehicle Control, Stanford University", 2016.

[Markus] "Markus Kuderer, Shilpa Gulati, Wolfram Burgard, Learning Driving Styles for Autonomous Vehicles from Demonstration, Robotics and Automation (ICRA)", 2015.

[Ann] "Ann Nowe, Peter Vrancx, Yann De Hauwere, Game Theory and Multi-agent Reinforcement Learning, In book: Reinforcement Learning: State of the Art, Edition: Adaptation, Learning, and Optimization Volume 12", 2012.

[Kok-Lim] "Kok-Lim Alvin Yau, Hock Guan Goh, David Chieng, Kae Hsiang Kwong, Application of Reinforcement Learning to wireless sensor networks: models and algorithms, Published in Journal Computing archive Volume 97 Issue 11, Pages 1045-1075", November 2015.

[Sutton] "Sutton, R. S., Barto, A. G., Reinforcement Learning: an Introduction, MIT Press", 1998.

[Madera] "Madera, M., Megherbi, D. B., An Interconnected Dynamical System Composed of Dynamics-based Reinforcement Learning Agents in a Distributed Environment: A Case Study, Proceedings IEEE International Conference on Computational Intelligence for Measurement Systems and Applications, Italy", 2012.

[Al-Dayaa]
"Al-Dayaa, H. S., Megherbi, D. B., Towards A Multiple-Lookahead-Levels Reinforcement-Learning Technique and Its Implementation in Integrated Circuits, Journal of Artificial Intelligence, Journal of Supercomputing. Vol. 62, issue 1, pp. 588-61", 2012.

[Chowdappa]
"Chowdappa, Aswini., Skjellum, Anthony., Doss, Nathan, Thread-Safe Message Passing with P4 and MPI, Technical Report TR-CS-941025, Computer Science Department and NSF Engineering Research Center, Mississippi State University", 1994.

[Mnih] "V.Mnih and et al., Human-level Control Through Deep Reinforcement Learning, Nature 518.7540", 2015.

[Stampa]   "G Stamp, M Arias, etc., A Deep-reinforcement Learning Approach for Software-defined Networking Routing Optimization, cs.NI", 2017.

[Krizhevsky]
"A Krizhevsky, I Sutskever, and G Hinton, Imagenet classification with deep con- volutional neural networks, In Advances in Neural Information Processing Systems, 1106-1114", 2012.

[Volodymyr]
"Volodymyr Mnih and et al., Asynchronous Methods for Deep Reinforcement Learning, ICML, arXiv:1602.01783", 2016.

[MS]   "Intelligent Network Management using Reinforcement-learning, draft-kim-nmrg-rl-03", 2018.

[Ju-Bong]   "Deep Q-Network Based Rotary Inverted Pendulum System and Its Monitoring on the EdgeX Platform, International Conference on Artificial Intelligence in Information and Communication (ICAIIC)", 2019.

Authors' Addresses

Min-Suk Kim
Etri
161 Gajeong-Dong Yuseung-Gu
Daejeon 305-700
Korea

Phone: +82 42 860 5930
Email: mskim16@etri.re.kr

Youn-Hee Han
KoreaTech
Byeongcheon-myeon Gajeon-ri, Dongnam-gu
Choenan-si, Chungcheongnam-do
330-708
Korea

Phone: +82 41 560 1486
Email: yhhan@koreatech.ac.kr

Yong-Geun Hong
ETRI
161 Gajeong-Dong Yuseung-Gu
Daejeon 305-700
Korea

Phone: +82 42 860 6557
Email: yghong@etri.re.kr

Internet Research Task Force K. Sivakumar
Internet-Draft M. Chandramouli
Intended status: Informational Cisco Systems, Inc.
Expires: May 1, 2018 October 28, 2017

Concepts of Network Intent
draft-moulchan-nmrg-network-intent-concepts-00

Abstract

This document presents an overview of the concepts of Network Intent and provides definitions for some of the nomenclature. Some potential use cases are presented.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Recently, there have been deployments of networks of Service Provider, enterprise and data centres in a very large scale. From a network management perspective, the manageability of networks of such scale poses new challenges. The increasing complexity of network configuration is an additional challenge for the network administrators. To an extent, for device-level configurations, there has been standardization efforts underway in technologies such as YANG [RFC6020], and NETCONF [RFC6241]. However, the challenge still remains at the network level configuration, orchestration and management. The complexity of the network can lead to potential mis-configurations and furthermore, it may be difficult to troubleshoot the network failure conditions.

From a management perspective, it is of paramount importance for the network administrator to reduce the complexity of the network management. There are several measures and approaches that have been under consideration towards that objective. One aspect that has gained attention is Network Programmability APIs in the management plane. Programmability allows the capabilities of network functionality to be modified or extended. Programmability promises to enable the development of a whole new wave of applications that provide additional management intelligence. Programmability enables the development of applications whose purpose is to make the networks easier to manage, and those applications can be embedded and tightly coupled with the network. The application developers can use the Network Programmability APIs that can allow them to add new features that can facilitate ease of network management, efficiency and the effectiveness with which the network can be provisioned, administrated and managed. Programmability, as provided through SDN, provides exciting new opportunities to increase manageability by facilitating the development of corresponding applications. Software defined networking (SDN) is an umbrella term for a programmatic approach to managing network devices, using software controls to replace manual configuration. Initial motivations for SDN were to overcome the the lack of network programmability, and manageability in networks.

SDN technologies allow network-wide visibility and the possibility of feedback actions across the network. The desire to implement higher layers of management abstraction such as policy-based management, or the desire to extend an application’s capabilities with application-specific pre-processing that can be delegated to the network.

Leveraging the Network Programmability APIs opens the possibility to introduce an abstraction for the network, which can be used to synthesise the overall system behaviour. In the networking parlance, there have been several concepts that have been have been considered to simplify the network management - Network Policy, Autonomic Networking, Service Models, and Network Configuration. We introduce the concept of Intent Based Networking, by which the network administrator can articulate a desired outcome to the network. The Network Intent is translated to appropriate network policies and/or network configurations. With this approach to Network Intent, the focus is more on "what" the network should do and less on "how" i.e., the intermediate steps that should be executed. This level of abstraction can be referred to as "Network Intent". The implicit assumption is that for "Network Intent" there might be some prerequisite steps that may need to be performed, such as the network elements are discovered and controlled, and device capabilities and features are identified.

While there has been investigations of Network Intent, there are some still ambiguities in terms of the terminology used. This initial proposal is an attempt to clarify some of the terms and provides a brief outline of the goals or the vision intended. Some use cases are presented to illustrate the concepts introduced in this document.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

3. Hierarchy of Manageability

There is a certain non-physical, logical hierarchy in a network management environment, as described in the figure below. The user at the top of the hierarchy can be represented by a "real" user or a system that performs actions on behalf of a user, such as a management station.

The "user" establishes an "intent" to be taken on the network as a whole and pushes that intent to the second layer of the management hierarchy, which consists of the intent engine.

The next layer of the hierarchy consumes the "intent" and translates the intent to desired actions based on the meaning of the intent.

The bottom layer of the hierarchy consists of the devices on the network that consume the configurations and actions issued to them by the intent engine. These devices sit directly on the network and are responsible for traffic flowing through the network.

```
                            +---------+
                            |  user   |
                            +----+----+
                                 |
                                 | Intent  API
                                 |
                          +------+-------+             +------+-------+
                          |  SDN         |             |              |
                          +              +-------------+Intent Engine +
                          |  Controller  |             |              |
                          +------+-------+             +------+-------+
                                 |
                                 | Instruct
                                 | network
              +------------------+------------------+
              |                  |                  |
         +----+----+        +----+----+        +----+----+
         | Device  |        | Device  |        |  Device |
         +---------+        +---------+        +---------+
```

Figure 1

4. Network Configuration

Network configurations are the most basic atomic operations that can be performed on a network device. A particular feature of the network software can be enabled by one or many lines of network configurations. Often the network devices are configured by experts with _domain expertise_ and based on the functionality the network device has to perform. Often, network configuration is performed on a device by device basis and this is a manual process. Automation of this process is very important step, which can save time and reduce the possible number of mis-configurations.

5. Network Policy

Policy based network management has been widely discussed in the literature [JNSM]. Several proposals for the semantics and structure for expressing network policy have been considered. There are some particular implementations and deployments of network policies such as Performance Forwarding, QoS profiles etc.

A network policy can be viewed as a set of rules a network administer can use to manage the network resources; for example to provide differential treatment for traffic. Policies can be at a network level and can provide a way of consistently managing multiple network devices. The administrator can define policies and specify how the network devices should deal with different types of traffic. Policies can be defined to be conditional, in the sense, if there is a condition A is observed, then a set a network policy can be implemented on some network devices. Policies can be a group of network configurations which perform a specific function that can be applied to network devices. In the SDN paradigm, network policies can be pushed to the network devices using NETCONF [RFC6020] and RESTCONF [RFC8040].

6. Network Intent

Network Intent can be considered as a declarative paradigm by which the network administrator articulates a desited outcome or the state of the network. In abstraction, the network enables a set of services that can be consumed. In particular, Network Intent is a desirable functionality that can be enabled from an SDN Controller. There are potential benefits of ease-of-use and operational simplicity and the capability of programming the entire network.

Network Intent need not be prescriptive or expressed explicitly in terms of specific actions. The following are the intended design considerations of network intent.

o First, there may be several alternative approaches to realise a specific Network Intent.

- o Second, it is conceivable that it may not be possible to realise some of the Network Intents due to non-availability of network resources or the network may not have functionality.

- o Third, some new Network Intent can be in conflict with the current state of the network or can disrupt the Network Intents expressed previously. It is assumed such a feedback regarding the conflicts is provided back to the administrator the originator of the Network Intent. Based on the feedback, it should be possible for the network administrator to refine the new Network Intent.

This proposal or definition of Network Intent can be viewed as analogous to the promise theory framework proposed [Promise]. In order to realise the Network Intent, it may be useful consider a logical functional block - the Intent Engine - that can resolve the network intent and render the Network Intent appropriately on to the network.

The simplistic method to realise network intent is to consider linear one-to-one mapping of Network Intents to actual network policies or network configurations. In a more general framework of Network Intent, it should be possible to consider a more general approach leveraging artificial intelligence based techniques so that the Network Intent can be accurately realised and appropriately rendered on the network. Translating the intent requests to rendering actions would require the modelling of network devices and the functionalities and configurations.

In order to realise a Network Intent eventually that should consist of network configuration blocks that can be implemented in one or more network devices.

There is a general confusion between policy based network management and intent based network management. An analogy can be drawn between intent based network management and the automotive industry. Though cliched, this analogy provides the closest match. Many cars, if not all, have cruise control as a function today. Cruise control is a very simplistic functionality that keeps the car going at a specific speed. It monitors the speed and adjusts it up or down. This can be considered as a policy, to keep the car driving at certain speed, until the operator disengages the policy manually.

An car that can handle intent would, on the other hand, accept a request such as "take me from San Francisco to Los Angeles within 6 hours," plot the appropriate path based on historical data on which roads are the best ones to take to achieve the constraint of reaching within 6 hours and plots the direction to go in. Then it would constantly monitor the traffic on the path and provide feedback to the operator about whether the path chosen will still achieve the constraint. If the constraint cannot be achieved, then it either re-plots the path or lets the operator know that the constraint cannot be achieved and requests a new constraint. The operator is removed from making the decision about which exact path to take and is instead just providing the constraints that need to be achieved.

7. Use Cases

This section lists certain use cases that showcase the value of intent based network management. There are a variety of use cases where intent based network management is of value but the highest value is present in scenarios where a network needs to be reprogrammed in a significant manner in the shortest of time frames. Such a network reconfiguration should not result in misconfiguration that could result in the loss of communication capabilities for the users of the network.

We provide two scenarios where such a reconfiguration of the network is required. There are obviously many more day-to-day scenarios where the intent of change or monitoring of a network can be of a much lower scale.

7.1. A simple example

The network administrator articulates the Network Intent, "Route traffic from Node A to Node B with minimum bandwidth of K mbps". The Intent Engine then resolves the intent. This step involves understanding the intent expressed and the second step to resolving that intent would require performing routing calculations between Node A and Node B. This is a key step involved in this proposal.

Once the intent has been resolved, routing calculations are well-known and there are standard techniques taking into account the network topology between Node A and Node B; the current utilisations with minimum guaranteed bandwidth of K Mbps between Node A and Node B. Once the path is determined, that routing and next hop configurations are communicated to the respective network nodes.

7.2. Disaster Management

Planning for disaster management and sudden reconfiguration of infrastructure is common in the "physical" world - ie roads, water supply, electricity, etc. Similar reconfigurations of communication networks also is important during a disaster. During a disaster management / recovery, it is important to ensure that emergency communication traffic (such as 911 in the USA, 999 in UK and similar in other countries) gets more bandwidth and resources than non-emergency communication. It is also important to allow people to communicate with their family members inside and outside the disaster area, to help in recovery efforts. For this reason, voice communication, including VoIP, should be prioritized over streaming video services.

Such a disaster management is geographically bounded, therefore the network changes need to also be appropriately geographically bounded. This is very often hard to apply manually in a very large network at the moment that the change is needed. Intent based networks can provide an abstraction that use the underlying knowledge of the network and policies to achieve an action to provide this ability in a finer grained manner.

As the disaster scenario subsides the applied intent should automatically subside as well. This requires not only action to be taken based on policies, but also requires constant monitoring of the operational state network. Such monitoring presents significant amounts of data and it is quite hard to build rules and conditions to operate on such data while minimizing mistakes. Machine learning based monitoring can provide a mechanism to make applying an intent easier, especially in very large networks. Such machine learning based mechanisms can be integrated with physical world monitoring to identify when a disaster hits a certain geography and to automatically trigger a pre-set intent for that scenario. With such machine learning mechanisms and multiple pre-set intents, it would be possible for a management system to automatically trigger a specific intent when it detects a particular scenario. Similar combination of operational monitoring and intent based networking mechanism can be used to withdraw an intent when the disaster like scenario recedes.

8. Issues with Intent based networking

Intent based network management is about creating an abstraction to handle the management of a network. Naturally issues related to any abstraction mechanism applies here as well. Specifically, an abstraction like this removes the direct interaction of a user with the network for operations management. While the original creators of this intent, and the associated policies, would have understood the reasoning behind this intent, and more importantly the fine distinction between when to apply and when NOT to apply such an intent, later users of the system may not have that clear distinction and may apply this intent needlessly. This problem exists in any abstraction mechanism.

9. Security Considerations

This draft currently does not impose any security considerations.

10. IANA Considerations

This memo has no actions for IANA.

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

11.2. Informative References

[JNSM] Boutaba, R. and I. Aib, "Policy-Based Management: A Historical perspective, Journal of Network and Systems Management 15 (4), 447-480", 2007.

[Promise] Borril, P., Burgess, M., Craw, T., and M. Dvorkin, "A Promise Theory Perspective on Data Networks, CoRR, abs/1405.2627", September 2014.

Authors’ Addresses

Kaarthik Sivakumar
Cisco Systems, Inc.
Sarjapur Outer Ring Road
Bangalore 560103
India

Phone: +91 80 4429 2264
Email: kasivaku@cisco.com
URI: http://www.cisco.com/

Mouli Chandramouli
Cisco Systems, Inc.
Sarjapur Outer Ring Road
Bangalore 560103
India

Phone: +91 80 4429 2409
Email: moulchan@cisco.com
URI: http://www.cisco.com/

Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: December 23, 2017
June 21, 2017

K. Wehmuth
A. Ziviani
LNCC

A Reference Model for Representing SDN Environments
draft-wehmuth-nmrg-sdn-model-00

Abstract

Software-Defined Networks (SDNs) are multilayer systems. In this context, this draft defines a graph-based reference model capable of properly representing such complex multilayer networks. The defined reference model thus eases the management and planning of SDN environments.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Software-Defined Networks (SDNs) are inherently multilayer systems. In addition to the traditional layers associated with the separated data and control planes, other layers can be considered to support structures, such as hierarchical controllers, structured interaction between applications, use of Network Functions Virtualization (NFV) on SDN environments, among others. It is important to properly represent such a complex structure in a convenient way that allows modeling and analysis of a SDN environment with a single object.

In this context, we propose the use of a theoretical graph framework [Wehmuth2016], capable of modeling multilayer complex networks, for representing SDN environments. This framework is capable of representing complex networks containing an arbitrary (finite) number of layers, thus allowing the representation of SDN systems with any number of associated layers. In this framework, if desired, the usual SDN layers can be divided into sub-layers allowing the creation of more detailed and structurally rich SDN reference models. Therefore, this framework is capable of modeling various distinct SDN architectures, such as ForCES [RFC3746], SDN systems adherent to [RFC7426], [draft-irtf-sdnrg-pop-00], or any other layered networking architecture. Further, the considered framework has the property of guaranteeing that any model created in it is necessarily equivalent (isomorphic) to a directed graph. Therefore, all knowledge available for directed graph analysis can be directly applied to representation based on this framework. Additionally, the graph theoretical knowledge can be extended in the framework in order to allow for results based on advanced aggregation of layers that are present on the represented models.

Since SDN reference models created using the proposed framework are guaranteed to be equivalent to directed graphs, they can be represented in their canonical compact form, or by means of matrices usually employed for graph representation (e.g., adjacency matrices). Further, well-known graph algorithms can be applied directly to the representation based on the considered framework, allowing for the straightforward computing of distances among objects on a SDN system, the evaluation of the flow capacity of any given path on the system, the finding of structurally relevant objects or edges in the system (i.e. centrality evaluation), the construction of flow matrices, or any other operation possible for directed graphs.

The proposed SDN reference model fully reflects the complexity of SDN systems, while also allowing the straightforward usage of the model as a directed graph. Moreover, the fact that the whole network structure can be represented by a single mathematical object greatly contributes to the consistency of the obtained results. Therefore, the proposed framework can be useful either in an offline environment, where it can be used for system design and simulation of what-if scenarios, or as an online environment deployed, for instance, in the SDN controller(s), allowing for real-time evaluation of the structural properties of the whole system network. The proposed reference model for representing SDN environments thus contributes to the management and planning of these environments.

2. Why modeling SDNs as multilayer networks

Since SDNs are intrinsically layered systems, it is natural to model it as a multilayer network. Moreover, the usage of such a model has the advantage of clearly exposing the SDN layered structure. In a multilayer model, not only the natural layers visible on a SDN are clearly represented, but also, if desired, it is possible to divide each SDN layer into a set of sub-layers. In this way, structures such as hierarchical distributed control architectures, where multiple controllers with distinct hierarchy can be allocated to distinct control sub-layers. In this manner, not only the topological structure of the controllers is clearly modeled, but also, their hierarchical structure. Further, structures that may sometimes be attached to a SDN system, such as NFVs, can be modeled in layers specifically reserved for them, making the whole structure clear.

Moreover, by modeling a SDN as a multilayer network, it becomes possible to take advantage from the body of knowledge already established in graph theory for analyzing the SDN structure.

3. How to model a SDN as a multilayer network

3.1. Introduction to MultiAspect Graphs

A MultiAspect Graph (MAG) is a graph generalization introduced in [Wehmuth2016] that is shown to be equivalent to a directed graph. In this generalization, the set of vertices, layers, time instants, or any other independent features are considered as an aspect of the MAG. For instance, a MAG is able to represent multilayer or time-varying networks, while both concepts can also be combined to represent a multilayer time-varying network and even other higher-order networks. Since the MAG structure admits an arbitrary (finite) number of aspects, it hence introduces a powerful modeling abstraction for networked complex systems.

3.2. Multilayer graph (MLG) definition

We propose to model SDN systems by using a Multilayer Graph (MLG) model, that is a particular case of a MultiAspect Graph˜(MAG) [Wehmuth2016], in which the vertices and layers are the key features (i.e., aspects) to be represented by the model. Formally, a MAG can be defined as an object H=(A,E), where E is a set of edges and A is a finite list of sets, each of which is called an aspect. In our case, for modeling a MLG, we have two aspects, namely vertices and layers, i.e. |A|=2. For the sake of simplicity, this 2-aspect MAG can be regarded as representing a MLG with an object H = (V, E, L), where V is the set of vertices, L is the set of layers, and E is a subset of (V X L X V X L), that is the set of edges. As a matter of notation, we denote V(H) as the set of all vertices in H, E(H) the set of all edges in H, and L(H) the set of all layers in H.

An edge e in E(H) is defined as an ordered quadruple e = (u, la, v, lb), where u,v in V(H) are the origin and destination vertices, while la, lb in L(H) are the origin and destination layers, respectively. Therefore, e = (u, l_a, v, l_b) should be understood as a directed edge from vertex u at layer la to vertex v at layer lb. If one needs to represent an undirected edge in the MLG, both (u, l_a, v, l_b) and (v, l_b, u, l_a) should be in E(H).

An edge e= (u, la, v, lb) in our model may be classified into four classes depending on its characteristic:

o Intralayer edges connect two vertices in a same layer, e is in the form of e =(u, la, v, la)$, where u and v are distinct;

o Interlayer edges connect the same vertex in two distinct layers, e is in the form of e=(u, la, u, lb), where la and lb are distinct;

- Mixed edges connect distinct vertices in distinct layers, e is in the form of e=(u, la, v, lb)$, where u and v are distinct and $la and lb$ are distinct;

- Intralayer self-loop edges connect the same vertex in the same layer, e is in the form of e=(u, la, u, la).

Further, we define a composite vertex as an ordered pair (u, la), where u in V(H)$ and l_a in L(H). The set VL(H) of all composite vertices in a MLG H is given by the Cartesian product of the set of vertices and the set of layers, i.e. VL(H) = V(H) X L(H)$. As a notation note, a composite vertex is represented by the ordered pair that defines it, e.g. (u, l_a), where u in V(H) and la in L(H).

3.3. Algebraic representations and structures

In this section, we discuss ways to properly represent a MLG using our proposed model. Similarly to static graphs, a MLG can be fully represented by an algebraic structure, like the MAG structure from which our MLG model is derived. In this work, we adopt matrix-based representations, in particular the adjacency matrix.

In order to illustrate such representations, we use the MLG W presented in Figure 1.

3.4. MLG adjacency matrix

Since every MAG has a directed graph that is equivalent to it, the same holds for our MLG model, since it is a particular specialized case of a MAG. Consequently, it follows that the MLG can be represented by an adjacency matrix. For the sake of standardization and without loss of generality, we define that in a MLG the first aspect represents the vertices (i.e. the objects that compose the SDN system) and the second aspect represents the layers of the represented system.

In the more general environment represented by a MAG, a companion tuple is used in order to properly identify and position each composite vertex of the equivalent graph in the adjacency matrix. Since the case we present in this work is restricted to MAGs with 2 aspects, it follows that the companion tuple is reduced to a pair, which in the first entry has the number of vertices and the second entry has the number of layers. For instance, considering the MLG example of Figure 1, the companion tuple associated with its adjacency matrix is (10,3), since there are 10 vertices and 3 layers. The function of the companion tuple is only to ensure that the order by which the composite vertices are placed in the adjacency matrix is the one shown in Figure 2. Since in the case where the number of aspects is restricted to 2 this placement can be easily achieved, in this work we do not further mention the companion tuple.

To get the MLG adjacency matrix, we only need to consider that each composite vertex (u,la) can be thought of as a vertex in a directed graph. This directed graph has |V| * |L| vertices and, as a consequence, its adjacency matrix has |V| * |L| * |V| * |L| = |V|^2 * |L|^2 entries. Since the non-zero entries of this matrix correspond to the edges of the MLG, further analysis show that this matrix is usually sparse and can therefore be stored in an efficient way.

```
       +---+     +---+                     +---+
       | A |     | A |                     | A |
       | 1 +-----+ 2 |                     | 3 |
       |   |     |   |                     |   |
       +-+-+     +-+-+                     +-+-+      Application Layer
   ......|.........|.........................|..........................
         |         |                         |
    +----+---------+----+           +--------+---------+
    |                   |           |                  |
    |        C1         +-----------+        C2        |
    |                   |           |                  |
    +-+--+---------+----+           +---+----------+---+
      |  |         |                    |          |  Control Layer
   .../..|.........|....................|..........|....................
     /   |         |                    |          |  Data Layer
    /  +-+-+     +-+-+                +-+-+      +-+-+
    |  | D |     | D |                | D |      | D |
    |  | 1 +-----+ 3 |                | 4 +------+ 5 |
    |  |   |     |   |                |   |      |   |
    |  +-+-+     +-+-+                +---+      +---+
    |    |         |
    |    |         |
    |    |  +---+  |
    |    +--+ D |  |
    +-------+ 2 +--+
            |   |
            +---+
```

Figure 1: SDN Example

Figure 2 shows the adjacency matrix obtained for the illustrative MLG W shown in Figure 1. From Figure 2, we highlight that the adjacency matrix form of the MLG has interesting structural properties.

```
   +-                                                           -+
   |0 1 1 0 0 0 0 0 0 0|0 0 0 0 0 1 0 0 0 0|0 0 0 0 0 0 0 0 0 0|D1
   |1 0 1 0 0 0 0 0 0 0|0 0 0 0 0 1 0 0 0 0|0 0 0 0 0 0 0 0 0 0|D2
   |1 1 0 1 0 0 0 0 0 0|0 0 0 0 0 1 0 0 0 0|0 0 0 0 0 0 0 0 0 0|D3
   |0 0 1 0 1 0 0 0 0 0|0 0 0 0 0 0 1 0 0 0|0 0 0 0 0 0 0 0 0 0|D4
   |0 0 0 1 0 0 0 0 0 0|0 0 0 0 0 0 1 0 0 0|0 0 0 0 0 0 0 0 0 0|D5 Data
   |0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|C1 Layer
   |0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|C2
   |0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|A1
   |0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|A2
   |0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|A3
   |...................|...................|...................|
   |0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|D1
   |0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|D2
   |0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|D3
   |0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|D4
   |0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|D5 Ctrl
   |1 1 1 0 0 0 0 0 0 0|0 0 0 0 0 0 1 0 0 0|0 0 0 0 0 0 0 1 1 0|C1 Layer
   |0 0 0 1 1 0 0 0 0 0|0 0 0 0 0 1 0 0 0 0|0 0 0 0 0 0 0 0 0 1|C2
   |0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|A1
   |0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|A2
   |0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|A3
   |...................|...................|...................|
   |0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|D1
   |0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|D2
   |0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|D3
   |0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|D4
   |0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|D5 Apps
   |0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|C1 Layer
   |0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 0 0 0 0|C2
   |0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 1 0 0 0 0|0 0 0 0 0 0 0 0 1 0|A1
   |0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 1 0 0 0 0|0 0 0 0 0 0 0 1 0 0|A2
   |0 0 0 0 0 0 0 0 0 0|0 0 0 0 0 0 1 0 0 0|0 0 0 0 0 0 0 0 0 0|A3
   +-                                                           -+
    D D D D D C C A A A D D D D D C C A A A D D D D D C C A A A
    1 2 3 4 5 1 2 1 2 3 1 2 3 4 5 1 2 1 2 3 1 2 3 4 5 1 2 1 2 3

          Data                Control             Apps
          Layer               Layer               Layer
```

Figure 2: SDN Matrix Example

First, each one of the ten vertices (identified as D1, D2, D3, D4, D5, C1, C2, A1, A2 and A3) of the MLG W clearly appears as a separate entity in each of the three layers (l0 - Data, l1 - Control, and l2 - Applications) that compose the MLG W. Second, the main block diagonal contains the entries corresponding to the intralayer edges of each layer. In these blocks, the entries corresponding to the intralayer edges of the MLG carry value 1. Finally, the entries at the off-diagonal blocks correspond to the interlayer edges. The eight interlayer edges present at the MLG W are indicated by the value 1 on the off-diagonal blocks. Further, we remark that all these structural properties derive from the order adopted for representing the vertices and layers present in the MLG and can be readily verified in the matrix form in a quite convenient way.

3.5. SDN reference model

From the MLG definition, it follows that a MLG can represent multilayer networks with an arbitrary (finite) number of layers. At a first glance, this would be enough to represent a multilayer system, such as a SDN. However, additional definitions can be made in order to provide a clear description of a SDN. For instance, a SDN reference model could benefit from an adequate name structure for its layers.

We start by naming the four basic layers considered in this work as Ld for the data layer, Lc for the control layer, La for the application layer, and Ln for the NFV layer. Further, each basic layer can be defined in a number of sub-layers, yielding Ld1 to Ldj for data plain layers, Lc1 to Lck for control plan layers, La1 to Lam for application layers and Ln1 to Lni for NFV layers. In this way, the total number of layers in the SDN model is given by |L| = j + k + m + i. Note that not all layers need to be necessarily represented. For instance, a simple SDN with 1 data plan layer, 1 control plan layer, 1 application layer, and no NFV layer, can be modeled by a 3 layer MLG, where j = k = m = 1 and i = 0.

We remark that since a MLG is equivalent to a directed graph, all extensions usually applied to graphs, such as edge weights and vertices weights can be directly applied to MLGs, and also, all algorithms known for directed graphs can be directly applied to MLG.

In addition to the traditional directed graph algorithms, it is possible to construct algorithms that use the full information present on the MLG and deliver aggregated results (e.g. results for vertices; disregarding layers). By using these algorithms, the results do not consider the artifacts generated by the traditional aggregation operation. This means, for instance, that aggregated paths are calculated using only paths that are actually present on the MLG.

4. Conclusion

In this work, we presented a SDN reference model based on MLGs, which are a special case of a MultiAspect Graph (MAG). In particular, a MLG is a MAG with exactly 2 aspects, named vertices and layers. Since the MLG has a fix number of aspects, it can be constructed with a simpler structure than a MAG.

We show that a MLG can properly represent a SDN system and that since the MLG inherits the basic properties of a MAG, in particular, the equivalence (isomorphism) to directed graphs, the knowledge present in the theory of directed graphs can be applied to our proposed reference model for representing SDN environments. This makes our model a convenient way of representing a SDN, by both expressing it as a multilayer system, while also providing a well established theoretical ground and available algorithms to build analytics.

5. IANA Considerations

This memo includes no request to IANA.

6. Security Considerations

Similarly to [RFC7426], this document does not propose a new network architecture or protocol and therefore does not have any impact on the security of the Internet. However, security in SDN environments is discussed in the literature, e.g. in [SDNSec], [SDNSecSrv], and [SDNSecOF].

7. Informative References

[Wehmuth2016]
Wehmuth, K., Fleury, E., and A. Ziviani, "On MultiAspect graphs", Theoretical Computer Science Vol. 651, pp. 50-61, DOI 10.1016/j.tcs.2016.08.017, October 2016.

[SDNSecOF]
Kloti, R., Kotronis, V., and P. Smith, "OpenFlow: A Security Analysis", 21st IEEE International Conference on Network Protocols (ICNP) pp. 1-6, October 2013.

[SDNSecSrv]
Scott-Hayward, S., O’Callaghan, G., and S. Sezer, "SDN Security: A Survey", In IEEE SDN for Future Networks and Services (SDN4FNS), pp. 1-7, 2013.

[SDNSec]
Kreutz, D., Ramos, F., and P. Verissimo, "Towards Secure and Dependable Software-Defined Networks", In Proceedings of the second ACM SIGCOMM workshop on Hot Topics in Software Defined Networking, pp. 55-60, 2013.

[I-D.irtf-sdnrg-pop]
Tian, Y., "Programming Model for Protocol Oblivious Forwarding SDN Networks", draft-irtf-sdnrg-pop-00 (work in progress), January 2017.

[RFC3746] Yang, L., Dantu, R., Anderson, T., and R. Gopal, "Forwarding and Control Element Separation (ForCES) Framework", RFC 3746, DOI 10.17487/RFC3746, April 2004, <http://www.rfc-editor.org/info/rfc3746>.

[RFC7426] Haleplidis, E., Ed., Pentikousis, K., Ed., Denazis, S., Hadi Salim, J., Meyer, D., and O. Koufopavlou, "Software-Defined Networking (SDN): Layers and Architecture Terminology", RFC 7426, DOI 10.17487/RFC7426, January 2015, <http://www.rfc-editor.org/info/rfc7426>.

Authors' Addresses

Klaus Wehmuth
LNCC
Avenida Getulio Vargas, 333
Petropolis, RJ 25651-075
Brazil

Phone: +55 24 2233-6000
Email: klaus@lncc.br

Artur Ziviani
LNCC
Avenida Getulio Vargas, 333
Petropolis, RJ 25651-075
Brazil

Phone: +55 24 2233-6199
Email: ziviani@lncc.br

none
Internet-Draft
Intended status: Informational
Expires: May 3, 2018

S. Yan
Huawei
P. Martinez-Julia
NICT/Japan
A. Cabellos-Aparicio
Technical University of Catalonia
October 30, 2017

A General Considerations of Intelligence Driven Network
draft-yan-idn-consideration-00

Abstract

This document aims to pinpoint the work scope of Intelligence Driven Network (IDN) and mine the potential standardization work. Firstly, the problems and new requirements for the existing methods are analyzed. Numbers of high value use-cases are proposed as examples to instantiate them. A benchmark framework design is proposed, which is important during the machine learning and inference process. Finally, a reference model of IDN is proposed, based on which the potential standardization work is analyzed.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119] when they appear in ALL CAPS. When these words are not in ALL CAPS (such as "should" or "Should"), they have their usual English meanings, and are not to be interpreted as [RFC2119] key words.

Status of This Memo

This Internet-Draft will expire on May 3, 2018.

Copyright Notice

Table of Contents

1. Introduction

Recently, AI technology has made a great achievement and become more and more popular. The combination of AI and network is also a hot topic. The concept of Intelligence Driven Network (IDN) has been proposed. This concept is intended to describe the schemes that introducing AI into network and provide new solutions for the current and future network problems. There has been quite a lot of discussions about the AI application in the network in both academic and industrial area. However, the detail works, especially the potential standard points are still not clear.

In this document, we want to summerize the valuable content in the idnet maillist and make clear about the following.

- o What are the requirements? In network area, what problems need AI to solve? It always makes misunderstanding that AI is almighty. But it is factual that AI has both advantages and disadvantages. The work scope and scenarios, which AI may be useful and perform well, will be discussed and analyzed.

- o What are the gap when combining AI and network? The modern AI algorithms are proposed by image processing area but not network. Most of the algorithms cannot be migrated and used directly. Take the data format as an example. The input and output of the AI algorithm may be just numerical matrix or vector. The network data are not entirely formatted and regular. They need to be translated or converted before and after the algorithm. The gaps, like the data format, data orchestration and etc., will be analyzed.

- o What are the potential and new standard points? The intruduction of AI will bring new requirements for the current network. For example, the AI engine may need high frequency and high accuracy data to feed. Moreover, these data needs to be captured and transmitted in real-time and continuously. What improvements should be accomplished for the existing protocols? Whether there are new protocol requirements? What communication processes are universal and what kinds of data format that can be utilized in most of the scenarios?

This document aims to become the blueprint for the future work. The structure is organized as following. Section 2 describes the work scope of idnet and summerize the use cases. Section 3 indicates the analysis of measurement and data format. Section 4 discusses about the benchmark of data. Section 5 abstracts the IDN architecture and gives a brief analysis of potential standard points. Section 6 points out the new security challenge which AI brings to the network. Section 7 to 9 are IANA, Acknowledgements and References.

TBD

2. Scope and use cases

TBD

2.1. Scope

A general description about what should be focused during the IETF work and what should not. Clarify the work boundary. TBD

2.2. High Value Use Cases

There are numbers of use cases, which have been discussed in the idnet mail list. Describe the scenarios that may be useful and valuable. A details analysis may be helpful for the data and protocol design.

2.2.1. Traffic Prediction

Collect the history traffic data and external data which may influence the traffic. Predict the traffic in short/long/specific term. Avoid the congestion or risk in previously.

The process, data format and message needs are:

Process: 1. Data collection (e.g. traffic sample of physical/logical port); 2. Training Model; 3. Real-time data capture and input; 4. Predication output; 5. Fix error and go back to 3.

Data Format:

Time : [Start, End, Unit, Number of Value, Sampling Period]

Position: [Device ID, Port ID]

Direction: IN / OUT

Route : [R1, R2, ..., RN] (might be useful for some scenarios)

Service : [Service ID, Priority, ...] (Not clear how to use it but seems useful)

Traffic: [T0, T1, T2, ..., TN]

Message :

Request: ask for the data

Reply: Data

Notice: For notification or others

Policy: Control policy

2.2.2. QoS management

It is worthy to predict the traffic change for avoiding the congestion and ensuring QoS. As the following figure shown, the AI system continuously collects link status data from the network. This AI system is responsible for two things. One is monitoring and predicting the traffic on each link and the other one is calculating the usable route for any pair of nodes according to the prediction and current link status. Assume that there is a VPN named VPN_S_D from node S to D which pass through S-A-B-C-D. According to the prediction, there will be a huge traffic flow from node A to C in the future 10 min. The traffic will increase the end-to-end delay from S to D so that the QoS will not be ensured.

```
              x     x
        _ A ---- B ---- C._      link status   +----------+
      ,'    \       /      '.   =============>|IDN Engine|
    -'        \    /         '-                +----------+
  S ------I ---- J ----  K ----  D
    .         /   \          ,'
     '.     /       \      ,'
       '  O ---- P ----  Q '
```

There are at least two solutions. one is modifying the object’s configuration to avoid the potential congestion. For example, we modify the VPN_S_D route from S-A-B-C-D to S-I-J-K-D. The other one is restricting non-object’s transmission so that to protect the object’s QoS. For example, we increase the reserved bandwidth of VPN_S_D or modify the route of non-object flows from S-A-B-C-D to S-I-J-K-D therefore most of the traffic will not affect VPN_S_D.

Here we may have some challenges. Challenge 1 is the AI prediction and autonomic decision should be a quick response. The whole process must be finished before the congestion happens meanwhile the AI system is meaningless. The question is how to implement such quick response? Challenge 2 is whether there is existing protocols which can support high frequency measurement? Because AI system needs to be fed with continuous link status data. And the real-time data need to be captured frequently otherwise the route change will be worthless. I think the protocols that support high frequency measurement and data collection may become one of our focus point.

The process, data format and message needs are:

Process: 1. Data capture (e.g. traffic sample of physical/logical port); 2. Training Model; 3. Real-time data capture and input; 4. Output percentages; 5. Fix error and go back to 3.

Data Format:

Time : [Timestamp, Value type (Delay/Packet Loss/...), Unit, Number of Value, Sampling Period]

Position: [Link ID, Device ID]

Value: [V0, V1, V2, ..., VN]

Message :

Request: ask for the data

Reply: Data

Notice: For notification or others

Policy: Control policy

2.2.3. Deep Reinforcement-Learning Control of the Network

Recently important breakthroughs have been achieved in the area Deep-Reinforcement Learning (DRL) [REF1] architectures where agents can be trained online to operate complex environments and achieve quasi-optimal configurations. In this context, a DRL can be used to control the routing of the network and achieve the target policy set by the administrators (e.g., [REF2, REF3, REF4]).

The following figure describes a common architecture of a DRL operating a network. The agent acts upon the network (action) by changing the configuration, this results in the network changing its fundamental state (e.g, different per-link utilization and a different traffic load). Finally, the reward function is defined by the operator and represents the target performance (e.g., load-balance the traffic in the network). The agent will learn how to act upon the network to maximize the expected reward function.

```
                             +---------------+
          +------------------>               |
          |                  |     Agent     +---------------------+
          |  +--------------->               |                     |
          |  |               +---------------+                     |
          |  |                                                     |
    State |  |                                                     |
          |  |  Reward Function (Policy)                    Action |
          |  |                                                     |
          |  |                                                     |
          |  |                                                     |
          |  |    +------------------------------------+           |
          |  +----+                                    |           |
          +-------+              Network               <-----------+
                  |                                    |
                  +------------------------------------+
```

The main operational advantages of DRL agents with respect to existing optimization techniques are:

1. DRL are able to learn and generalize from past experience to provide solutions to unseen scenarios. This is not possible using existing optimization techniques that do not learn from the past.

2. Once trained, either offline or online, DRL agents can optimize in one single step. On the contrary, existing optimization techniques require to run iteratively each time a new scenario is found, for instance when a link goes down or the traffic changes in a significant way. It is worth noting that a common practice is to run such techniques in advance of common scenarios and store their resulting configurations, however it is very complex to consider all the potential scenarios.

3. DRL agents see the network as a black-box and do no need any prior assumption about the system. However heuristics, very commonly used in optimization strategies, are tailored for the problem they are trying to optimize. However, an operator only needs to change the reward function to implement a different target network policy.

In what follows we describe the process, data format and messages needed assuming a DRL agent that seeks to load-balance the traffic of the network that is, to minimize the maximum loaded link. This is a very common optimization strategy.

Process: 1.- Act upon the network by changing the routing configuration, for instance using a standard mechanism. 2.- Receive the state of the network, this is the per-link delay and the current traffic load. 3.- Compute the reward function as a function of the state. 4.- Deep Reinforcement Learning training. 5.- Go back to step 1.

Data Format

(state) Per-Link Utilization: [link id, utilization, averaging time]

(action) Change on the routing configuration. This can be done through the SDN controller and/or other standard mechanisms.

(reward) This is an algorithm that has as input the state and as output a value that represents how close we are to the target policy set by the operator. More about this can be found in the next section.

Messages:

State: Measure the per-link utilization

Action: Change the routing configuration

2.2.3.1. The Reward Function as the Network Policy

The agent seek to maximize the expected reward function and it represents the target policy that the agent will aim to achieve and configure on the network. In this context the reward function is the mathematical representation of the target network policy. However, the entire architecture includes a set of different pieces that may come from different vendors but must interoperate, the pieces are: the agent itself, the reward function and the state. This requires the following standardization efforts:

1. The reward function and its translation from the human-readable target network policy. The operators may want to use different vendor DRL agents that need to understand the reward function. Please note that the reward function depends on the representation of the state.

2. The state includes monitoring information about the network, such as the per-link utilization or the traffic load. Since the state is an input of the agent and is used in the reward function, there is a need for standard representation so that the different pieces can interoperate.

2.2.4. QoE Management via Supervised Learning

Networks can measure low-level metrics, such as delay, jitter and losses. However users perceive the performance of the network based on QoE metrics, such as Mean Opinion Scores. Unfortunately, QoE metrics cannot be typically directly measured over the wire and as such, need the subjective views of the users. The challenge is then to operate the network based on low-level metrics while fulfilling non-measurable QoE metrics. One of the main reason behind this challenge is that the relationship between the low-level and the QoE metrics are very complex, i.e. multi-dimensional and non-lineal.

```
        +-------------+            +---------------------+
        |  Supervised |  Extract   |Relation between QoE |
        |  Learning   +-Knowledge-->and low-level network+-------+
        |             |            |metrics              |       |
        +------^------+            +---------------------+       |
               +                                                 |
             Learn                                               |
               |                                     Install Knowledge
               |                                                 |
    +----------+--------------+                +-----------------v-----+
    |    Network Analytics    |                |                       |
    | (including Ground Truth)|                |  Network Management   |
    |                         |                |                       |
    +----------+--------------+                +-----------------------+
               ^                                                 |
               |                                                 |
               |                   +-------------+               |
               |                   |             |               |
               +-----Monitor-------+   Network   <----Operate----+
                                   |             |
                                   +-------------+
```

For this a well-established technique (e.g., see [REF5] and the references therein) is to follow the architecture depicted in the following figure. First the network low-level metrics are measured using telemetry, this information is stored in the Network Analytics platform. In addition to this users and or applications are polled to obtain QoE metrics of the network. The data-set containing both the low-level metrics and the QoE metrics is considered the ground truth.

By means of supervised learning (e.g., deep neural networks) we aim to learn the relation between the low-level and the QoE metrics. As an example we aim to learn the relation between the amounts of losses in different wireless links, the SNR and the utilization with the perceived MoS. Typically it has been shown that such relationship is non-lineal and multi-dimensional and as such, can be understood by a neural network. This relationship is the knowledge that we extract from the ground truth and it is used by the Network Management (NM) module. By means of this knowledge, the NM can understand how to operate the network based on low-level metrics (e.g., keep losses below a certain threshold) to fulfill QoE requirements.

2.2.5. TBD

3. Measurement and Data Format

TBD

3.1. Measurement Tools and Methods

The modern AI algorithms are mostly based on data-driven, which means that the AI engine needs quite plenty of data to feed and upgrade. In other words, higher frequency and accuracy data is required. The high scalability requirement needs distributed measurement tools to provide such abilities. The traditional methods and improvements may hardly support.

Firstly, the current measurement methods mostly orient to the service. For example, the voice service requires the end to end delay and jitter in a low level. Besides that, the AI engine may need more data from both network and other sources. For example, the QoE and identity information may influence the AI engine to make different decisions. The current measurement tools and data model cannot support this ability. Thus, the potential usable tools and methods, such as high frequency, high precision, new KPIs and so on, may need to develop.

Secondly, the current measurement methods mostly cannot support high frequency measurement. Even though it can, the data feedback scheme is commonly closed. The word "closed" means that the measured data is commonly sent to the device which launches the measure action rather than the data demander (AI Engine). The future measurement tools require more programmability, especially in the data feedback scheme.

TBD.

3.2. Data Format Analysis

There is huge gap between the current network data and algorithm data. The network data, such as IP address, delay, link utilization and etc., is mostly semantic. It means that each data actually describe a specific physical or logical entity. For example, one IP address means a certain location or a certain host in the network. However, the input and output data of an algorithm is usually non-semantic, which means it is not responding to a specific concept/action/device that can be found in the network. This depends on the fundamental design of AI algorithm and is hardly changed in the short term.

Another issue is that the AI engine potentially needs to obtain data from external sources. For the data that can be provided one-off, it is easily solved according to the application. For the data that needs to be provided continuously (e.g. the real-time external data), it is required to define the data format that satisfy the algorithm. Similarly, the output of algorithm may need to be translated into specific format that the next step devices can run and execute. Otherwise, it is hard to build up the full autonomic close loop of the network management. In other words, the data aggregation process is important and it is valuable to build the bridge between the network data and algorithm data.

TBD.

4. Benchmarking Framework

A standard benchmarking framework is required to assess the quality of an AI mechanism when it is used to resolve a specific problem in the network management and control area. It comprises a reference set of procedures, methods, models, and boundary values that *must* be enforced to the benchmarked mechanism, so that its operation can be comparable to other mechanisms and users can easily understand what to expect from each one.

Moreover, both the metrics included as a reference within the benchmarking framework and the results obtained from its application to a new mechanism must follow a standard format. Therefore, the standard formats must be enforced to all data, either being introduced to the benchmarking application or system (consumed), or obtained from its application (produced).

A common and decentralized "data market" can (and would) arise from the inclusion, dependency, and the general relation of all data, considering it is represented using the same concepts (ontology) and the standard format mentioned here. As a reference, it is worth to mention that a similar approach has been already applied to genome and protein data to build standardized and easily transferable data banks [PMJ1][PMJ2] [PMJ3], and they have demonstrated to be key enablers in their respective work areas.

The initial scope of input/output data would be the datasets, but also the new knowledge items that are stated as a result of applying the benchmarking procedures defined by the framework, which can be collected together to build a database of benchmark results, or just contrasted with other existing entries in the database to know the position of the solution just evaluated. This increases the usefulness of IDNET.

5. References Model and Potential Standardization Points

5.1. References Model

A three layers reference model of IDN has been proposed as follow. This architecture can cover, explain and support most of the current use cases and scenarios.

```
            +-----------+                          +----------+
            |Open       |------------------------->|          |
            |Application|    +---------------------+3rd Party |-+
            |Interface  |    | IDN Engine          |Algorithm | |
            +-----------+    | +---------+ +-----+ |Interface | |
            +------------+   | |Algorithm| |Model| |          | |
            |Data Refiner+-->| +---------+ +-----+ +----------- |
            +------------+   +----------------------------------+
                  ^          |    Training   |    Inference     |
    Intelligent   |          +----------------------------------+
    Layer         +-----------------+                  |
                  |                 |                  v
            +-------------+  +-------------+     +-------------+
            |External Data|  |Internal Data|     |  Policy     |
            |Interface    |  |Interface    |     |  Generator  |
            +-------------+  +-------------+     +-------------+
                  ^                 ^                  |
                  |                 |                  v
              +----------+   +-------------+    +----------------+
    Control   |3rd Party |   |Aggregating  |--->|Control Function|
    Layer     |Dataset   |   |Dataset      |    +----------------+
              +----------+   +-------------+    |   Inference    |
                  ^                  ^          +----------------+
                  |                  |                 |
                  |                  |                 |
                  |                  |                 v
            +-------------+    +-----------+      +------------+
    Infras- |Terminal/User|    |Measurement|      | Network    |
    tructure|Device       |--->|Function   |<-----| Function   |
    Layer   +-------------+    +-----------+      +------------+
```

The under layer is Infrastructure layer, which contains network function, measurement function and terminal/user device. The network function stands for the traditional routers, switches and other network devices, which are responsible for constructing the network foundations and forwarding data. The Measurement function stands for devices that can collect information from the network and various devices. A popular option are probe system, which is deployed distributed among the network. Besides that, some of the network devices integrate the measure function and play two roles. The information may involve but not limited the content listed in following table. The Terminal/User Device stands for the device that produces and consumes data, which may include PC, smart phone, datacenter, content storage server, cloud and etc. Some of the data produced by terminal/user devices is measurable. This type of data will be captured by the measurement function. Other types of data that cannot be measured directly by network measurement functions is represented as 3rd party datasets, which hopefully can be utilized in the future via 3rd party integration at the intelligence layer.

Type	Content
Network Data	Delay, Jitter, Packet Lose Rate, Link Utilization, ...
Device Data	Device Configuration, VPN Configuration, Slicing Configuration, ...
User Data	QoE Feedback, User Information, ...
Data Packet	Packet Sample, Packet Character, ...
Other Type	TBD

The middle layer is Control Layer, which contains Control Function, Dataset Aggregation (Function) and 3rd Party Dataset. The control function stands for entities that can control, configure and operate devices, especially network devices. In SDN, controller and orchestrator are control functions. Classical network devices such as routers integrate the forwarding and control functions (although as of today not with many instances of intelligent control functions). Classical routers therefore include functions from two layers. We foresee that the control function will most likely only perform intelligent inference, but not learn. For example, to execute neural networks, but do not train them. This is only an assumption at this time though and may prove to be wrong in the future when training becomes something easier defined into the control layer.

The aggregated dataset function owns the ability to gather and tidy the data. The database or database cluster is the typical example. Some of the control devices, such as SDN controller, integrate this function. Distributed instances aggregate data have also been defined. The network data can be directly sent back to the control function in support of network policies. For example, the controller can adjust the flow table according to the local cache which collects the network data periodically from the devices in its controlled area. The 3rd party dataset involves the data that may be provided by all kinds of applications or services. For example, the content provider may own social contact data and the map service provider may own the geographic data. This information does not belong to the network but could be very helpful for intelligent analytics and decision making in the network - which is why we device in the architecture the ability to communicate it between 3rd parties and the network.

The high layer, which is also the main body of IDN, is the Intelligence Layer. This layer is commonly deployed in the datacenter, or large scale computing centre that can support massive storage and computing resources. To the south direction, there are two interfaces which provides external data (3rd party data oriented) and internal data (network data oriented) access. We define a data refiner component to emphasize the need to adopt format and structure of various types of collected information to the needs of the IDN Engine.

The core of the IDN Engine are algorithm and model. The IDN Engine can be built based on the result of the large body of research and platform development work that already exists (albeit mostly developed for and deployed with non-network data). The platform should be agile extensible for future services, therefore we define a 3rd party Algorithm Interface to provide an adaptive developing ability. The user (or a 3rd party) may develop his/her own algorithms and upload then onto the IDN Engine via a northbound Open Application Interface. Additional Northbound Open Application interfaces can also be used to connect other software platforms to the IDN Engine to create a cooperation between multiple systems (not shown).

The output of IDN Engine is transmitted to the Policy Generator. Since the policy language might be machine readable or unreadable, the Policy Generator is responsible for generating the executable commands and connect to the control devices. This process refers to the interactions of northbound interface of control devices - which is what often gets standardized. Therefore, some of the potential standardization points will be mentioned in the following.

5.2. Measurement

In IDN, the intelligent system (or database) needs frequent and repeat measurement to obtain the link information. A fast measure and feedback protocol is needed to meet the requirement of measurement and data collecting. It may be based on SNMP or an absolutely new protocol. The intelligent system needs massive data to feed and support to formulate the policy and decision. Therefore, the measurement must be satisfy the data requirement of IDN. Firstly, there may be higher-level requirement for the existing measuring technology. The high timeliness is one of the potential point. The IDN’s control function needs accurate, global and highly real-time network data support. The current measure technology can only satisfy at least two characters of the three. Secondly, the IDN may need more kinds of data type to measure. Not only the delay, jitter and packet loss rate, but also the link utilization and other necessary parameters.

5.3. Data representation, transport and aggregation

The data representation is significant. Most of the current AI algorithms were born in the pattern recognition area, especially the image processing. The advantage of these algorithms is that they are very good at dealing with complex problems, especially mining and modeling the hidden relationship among the non-semantic data. One of the disadvantages is that almost all the algorithms require the training data has a high concordance. Fortunately, the image file instinctively owns this character. All the images can be expressed as uniform binary vectors or can be easily transformed into uniform format. But this condition is hardly satisfied in network area.

A uniform data format is required, which can implement the justification, correlation and affiliation of the data. Which may obtain the best performance of AI algorithm to mine the valid pattern hidden in the data. Since the intelligent system is data-driven, and the data resources are from different kind of vendors and device types, the data representation SHALL be consistent so that the intelligent system could merge the data and do the analysis/learning. Also, the data collection interface might also need to be standardized so that the interface is able to get the data the intelligent system needs.

Moreover, it is significant to standard the policy representation. Since there may multiply SDN controller system, a readable and uniform policy representation is valuable to improve the policy deploying efficiency and simplify the communication between controllers on the East-West direction.

5.4. Legacy Device Route control

Similar with IPv4/IPv6 transition, the IDN potentially faces to the legacy problem, which means that the new devices and functions will co-work with the legacy devices. Therefore, it is potentially required to design the control protocols to solve the transition problems.

5.5. TBD

TBD

6. Security Considerations

When security relevant decisions are made based on the use of intelligent analytics or automated intelligent decision making, care must be taken to understand the new security challenges. When for example more intelligent decisions are enabled through the collection of ever more data, it needs to be analyzed how that potentially enables attackers to easier feed data that derails the intelligent system ability to distinguish good from bad behavior.

TBD

7. IANA Considerations

There is no IANA action required by this document.

8. Acknowledgements

TBD

9. References

9.1. Normative References

[ISO_IEC10589]
"Intermediate system to Intermediate system intra-domain routeing information exchange protocol for use in conjunction with the protocol for providing the connectionless-mode Network Service (ISO 8473), ISO/IEC 10589:2002, Second Edition.", Nov 2002.

[RFC1195] Callon, R., "Use of OSI IS-IS for routing in TCP/IP and dual environments", RFC 1195, DOI 10.17487/RFC1195, December 1990, <https://www.rfc-editor.org/info/rfc1195>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC5301] McPherson, D. and N. Shen, "Dynamic Hostname Exchange Mechanism for IS-IS", RFC 5301, DOI 10.17487/RFC5301, October 2008, <https://www.rfc-editor.org/info/rfc5301>.

[RFC5304] Li, T. and R. Atkinson, "IS-IS Cryptographic Authentication", RFC 5304, DOI 10.17487/RFC5304, October 2008, <https://www.rfc-editor.org/info/rfc5304>.

[RFC5305] Li, T. and H. Smit, "IS-IS Extensions for Traffic Engineering", RFC 5305, DOI 10.17487/RFC5305, October 2008, <https://www.rfc-editor.org/info/rfc5305>.

[RFC5308] Hopps, C., "Routing IPv6 with IS-IS", RFC 5308, DOI 10.17487/RFC5308, October 2008, <https://www.rfc-editor.org/info/rfc5308>.

9.2. Informative References

[PMJ1] , <https://www.ncbi.nlm.nih.gov/genome/>.

[PMJ2] , <https://www.ncbi.nlm.nih.gov/genbank/>.

[PMJ3] , <https://www.rcsb.org/pdb/home/home.do>.

[REF1] "Human-level control through deep reinforcement learning. Nature, 518(7540), pp.529-533.", 2015.

[REF2] "A Deep-Reinforcement Learning Approach for Software-Defined Networking Routing Optimization. arXiv preprint arXiv:1709.07080.", September 2017.

[REF3] "A roadmap for traffic engineering in SDN-OpenFlow networks. Computer Networks, 71(C):1&#150;30", October 2014.

[REF4] "Packet routing in dynamically changing networks: A reinforcement learning approach. In Advances in neural information processing systems, pages 671&#150;678,", 1994.

[REF5] "A machine learning approach to classifying YouTube QoE based on encrypted network traffic. Multimedia Tools and Applications", January 2017.

Authors’ Addresses

Shen Yan
Huawei
Beiqing
Beijing, Haidian 100095
China

Email: yanshen@huawei.com

Pedro Martinez-Julia
NICT/Japan

Email: pedro@nict.go.jp

Albert Cabellos-Aparicio
Technical University of Catalonia

Email: albert.cabellos@gmail.com