Network Working Group
Internet Draft
Intended status: Standard Track
Expires: January 4, 2016

M. Klyus
NetCracker
J. Strassner
Huawei Technologies

July 4, 2015

SUPA Value Proposition
draft-klyus-supa-proposition-02

Abstract

The rapid growth in the variety and importance of traffic flowing over increasingly complex enterprise and service provider network architectures makes the task of network operations and management applications and deploying new services much more difficult. Simplified Use of Policy Abstractions (SUPA) defines an interface to a network management function that takes high-level, possibly network-wide policies as input and creates element configuration snippets as output. SUPA expresses policies using a generic policy information model, and outputs generic YANG data models.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

Copyright Notice

Table of Contents

1. Introduction

The rapid growth in the variety and importance of traffic flowing over increasingly complex enterprise and service provider network architectures makes the task of network operations and management applications and deploying new services much more difficult. In addition, network operators want to deploy new services quickly and efficiently. Two possible mechanisms for dealing with this growing difficulty are the use of software abstractions to simplify the design and configuration of monitoring and control operations and the use of programmatic control over the configuration and operation of such networks. Policy-based management can be used to combine these two mechanisms into an extensible framework.

Policy statements can be used to express high-level network operator requirements directly, or from a set of management applications, to a network management or element system. The network management or element system can then interpret those requirements to control the configuration of network elements.

The key benefit of policy management is that it enables different network elements and services to be instructed to behave the same way, even if they are programmed differently.

Simplified Use of Policy Abstractions (SUPA) will define a generic policy information model (GPIM) for use in network operations and management applications. The GPIM represents different types of policies for controlling the configuration of network elements throughout the service development and deployment lifecycle. The GPIM will be translated into corresponding YANG data models to define interoperable implementations that can exchange and modify generic policies using protocols such as NETCONF/RESTCONF.

Management applications will benefit from using policy rules that enable scalable and consistent programmatic control over the configuration of network elements.

1.1. Problem Statement

Network operators are faced with networks of increasing size and complexity while trying to improve their quality and availability, as more and more business services depend on them.

Currently, different technologies and network elements require different forms of the same policy that governs the production of network configuration snippets. The power of policy management is its applicability to many different types of systems. This provides significant improvements in configuration agility, error detection, and uptime for operators.

Many different types of actors can be identified that can use a policy management system, including applications, end-users, developers, network administrators, and operators. Each of these actors typically has different skills and uses different concepts and terminologies. For example, an operator may want to express that only Platinum and Gold users can use streaming and interactive multimedia applications. As a second example, an operator may want to define a more concrete policy rule that looks at the number of dropped packets. If, for example, this number exceeds a certain threshold value, then the applied queuing, dropping and scheduling algorithms could be changed in order to reduce the number of dropped packets.

1.2. Proposed Solution

SUPA enables network operators to express policies to control network configuration data models. SUPA provides a generic infrastructure that defines policies to control the configuration of network elements. The configuration process is independent of domain or type of application, and results in configuration according to YANG data models.

Both of the above examples can be referred to as "policy rules", but they take very different forms, since they are at different levels of abstraction and likely authored by different actors. The first example described a very abstract policy rule, and did not contain any technology-specific terms, while the second example included a more concrete policy rule and likely used technical terms of a general (e.g., IP address range and port numbers) as well as vendor-specific nature (e.g., specific algorithms implemented in a particular device). Furthermore, these two policy rules could affect each other. For example, Gold and Platinum users might need different device configurations to give the proper QoS markings to their streaming multimedia traffic. This is very difficult to do if a

common policy framework does not exist.

Note that SUPA is not limited to any one type of technology. While the above two policies could be considered "QoS" policies, other examples include:

- network elements must not accept passwords for logins
- all SNMP agents in this network must drop all SNMP traffic unless it is originating from, or targeting, the management network
- Periodically perform workload consolidation if average CPU utilization falls below X%

The above three examples are not QoS related, and will be explained more in Sections 4.1 and 4.2. This emphasizes the utility of the SUPA approach in being able to provide policies to control different types of network element configuration snippets.

There are many types of policies. SUPA differentiates between "management policies" and "embedded policies". Management policies are used to control the configuration of network elements. Management policies can be interpreted externally to network elements, and the interpretation typically results in configuration changes of collections of network elements. In contrast, "embedded policies" are policies that are embedded in the configuration of network elements, and are usually interpreted on network elements in isolation. Since embedded policies are interpreted in the network device, they are typically composed in a very specific fashion to run at near-realtime timescales.

1.3. Value of the SUPA Approach

SUPA will achieve an optimization and reduction in the amount of work required to define and implement policy-based data models in the IETF. Part of this is due to the generic and extensible framework of SUPA, which models concepts common to any type of policy as well as provides two information models (ECA and declarative), along with the associated YANG data models.

SUPA defines policy independent of where it is located. Other WGs are working on embedding policy in the configuration of a network element; SUPA is working on defining policies that can be interpreted external to network elements. Hence, SUPA policies can be used to define the behavior of and interaction between embedded policies.

SUPA can also be used to derive a (more abstract) information model from a (more specific) data model. This extracts data that is part of a particular technology and/or application and makes it reusable, so that these data can be applied to multiple technologies and/or domains.

The SUPA policy framework defines a set of consistent, flexible, and scalable mechanisms for monitoring and controlling resources and services. It may be used to create a management and operations interface that can enable existing IETF data models, such as those from I2RS and L3SM, to be managed in a unified way that is independent of application domain, technology and vendor. Resource and service management become more effective, because policy defines the context that different operations, such as configuration, are applied to.

2. Framework for Generic Policy-based Management

This section briefly describes the design and operation of the SUPA policy-based management framework.

2.1. Overview

Figure 1 shows a simplified functional architecture of how SUPA is used to define policies for creating network element configuration snippets. SUPA uses the Generic Policy Information Model (GPIM) to define a consensual vocabulary that different actors can use to interact with network elements. The GPIM defines a generic structure for imperative and declarative policies. This is converted to generic YANG data models. The IETF produces the models, and IANA is used to register the model and changes.

In the preferred approach, SUPA generic policy data models are then used to create vendor- and technology-specific data models. These define the specific elements that will be controlled by policies. The Policy Interface uses this information to create appropriate input mechanisms for the operator to define policies (e.g., a web form or a script) for creating and managing the network configuration. The operator interacts with the interface, which is then translated to configuration snippets. Note that the policy interface is NOT being designed in SUPA.

In one of possibly several alternate approaches (shown with asterisks in Figure 1), the SUPA generic policy YANG data models contain enough information for the Policy Interface to create appropriate input mechanisms for the operator to define policies. This transfers the work of building vendor- and technology-specific data models to the SUPA Data Model-Specific Translation

Function.

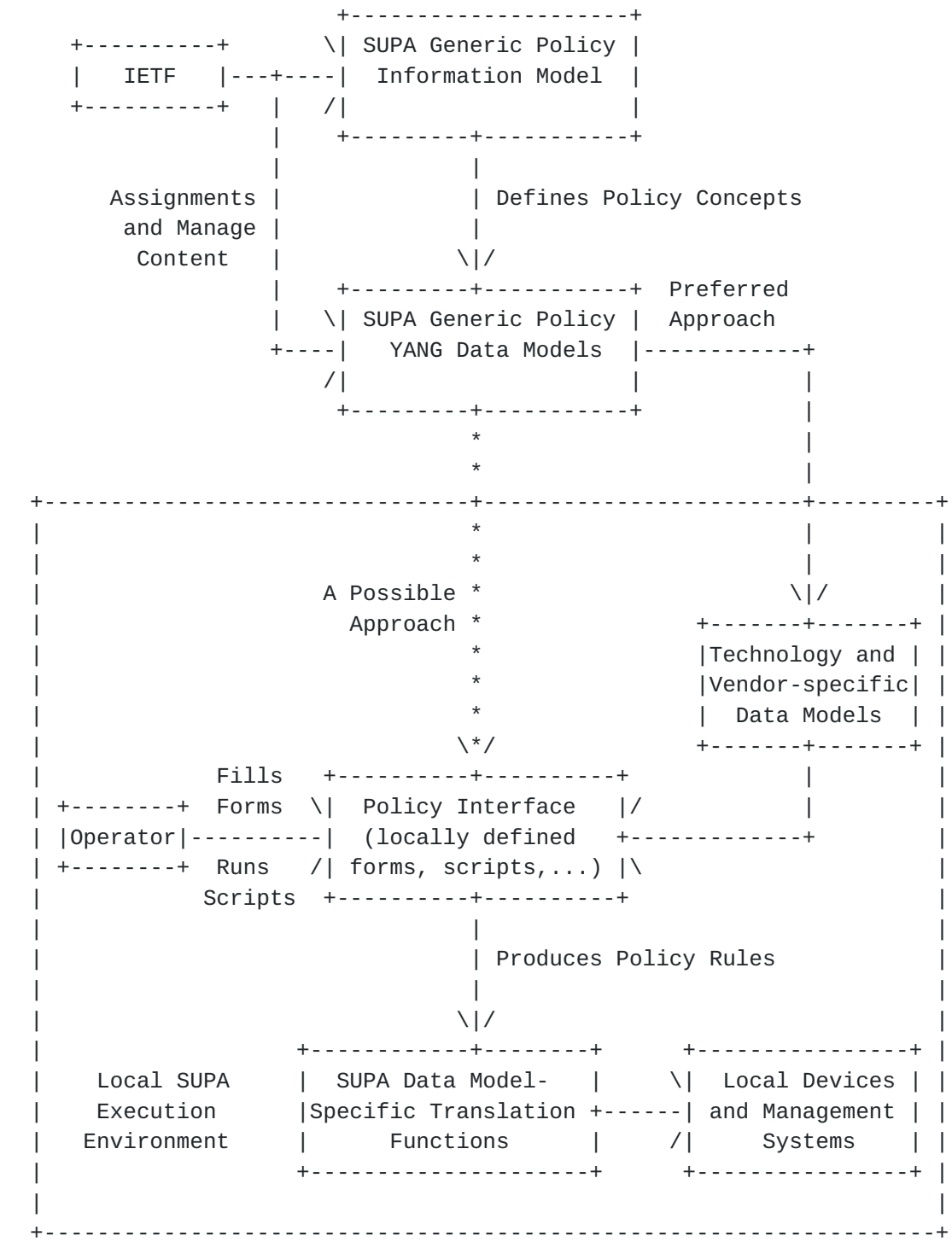

Figure 1. SUPA Framework

Figure 1 is meant to be exemplary. The Operator actor shown in Figure 1 can interact with SUPA in other ways not shown in the Figure. In addition, other actors that can interact with SUPA were not shown for simplicity. For example, an application developer could build an application that uses the SUPA information and data

models to directly output configuration snippets. In addition, other actors can use the SUPA framework.

SUPA defines an Event-Condition-Action (ECA) policy as an example of imperative policies; it also defines two forms of declarative policies using simple Propositional Logic and First Order Logic. An ECA policy rule is activated when its event clause is true; the condition clause is then evaluated and, if true, signals the execution of one or more actions in the action clause.

In contrast, a declarative policy defines what actions to take, but not how to execute them. Declarative policies in SUPA take the form of a set of statements that present facts, and a conclusion of those facts.

2.2. Operation

SUPA can be used to define various types of policies, including policies that affect services and/or the configuration of individual or groups of network elements. SUPA can be used by a centralized and/or distributed set of entities that for creating, managing, interacting with, and retiring policy rules. The Policy Interface and SUPA Translation Function are two entities that make up the Policy Management (PM) function.

The duties of the PM function depend on the type and nature of policies being used. For example, imperative (e.g., ECA) policies require conflict detection and resolution, while declarative policies do not. A short exemplary list of functions that are common to both types of policies include:

- policy creation, update, delete, and view functions (typically in conjunction with policy repositories)
- policy storage, search, and retrieval (typically uses distributed repositories that the PM communicates with)
- policy distribution (typically uses a message bus; note that this involves requesting and responding to requests for policy decisions as well as distributing policies and inforing interested entities of policy results)
- making policy decisions (this SHOULD include more than the simple Policy Decision Point functions defined in [RFC3198])
- executing policy decisions (this SHOULD include more than the simple Policy Enforcement Point functions defined in [RFC3198])
- validating that the execution of the policy produced what was expected (this is NOT defined in [RFC3198]).

An exemplary architecture that illustrates these concepts is shown in [TR235].

The SUPA scope is limited to policy information and data models. SUPA will not define network resource data models, which is out of scope. Similarly, SUPA will not define network service data models, which is also out of scope. Instead, SUPA will make use of network resource data models defined by other WGs or SDOs.

2.3. Generic Policy Information Model

The GPIM provides a common vocabulary for representing concepts that are common to expressing different types of policy, but which are independent of language, protocol, repository, and level of abstraction.

This enables different policies at different levels of abstraction to form a continuum, where more abstract policies can be translated into more concrete policies, and vice-versa. For example, the information model can be extended by generalizing concepts from an existing data model into the GPIM; the GPIM extensions can then be used by other data models.

SUPA will develop an information model for expressing policy at different levels of abstraction. Specifically, three information model fragments are envisioned: (i) a generic policy information model (GPIM) that defines concepts needed by policy management independent of the form and content of the policy, (ii) a more specific information model that refines the GPIM to specify how to build policy rules of the event-condition-action paradigm, and (iii) a more specific information model that refines the GPIM to specify how to build policy rules that declaratively specify what goals to achieve (but not how to achieve those goals); this is often called "intent-based" policy. These are all contained in the Generic Policy Information Model block in Figure 1.

2.4. Refinement of the GPIM

An information model is abstract. As such, it cannot be directly instantiated (i.e., objects cannot be created directly from it). Therefore, SUPA translates its information model to two different data models (which can be instantiated).

SUPA will translate the GPIM into concrete YANG data models that define how to manage and communicate policies between systems. Any number of imperative and/or declarative policy YANG data models may be instantiated from the GPIM, and may be used separately or in combination. This is enabled by the SUPA GPIM.

The two data models differ in how they represent policies. However, they share common characteristics and behavior. Therefore, it is easier to define a set of three information models to represent the common, ECA, and declarative parts of a policy. These three information models are then translated into either a YANG ECA data model or a YANG declarative data model. Note that because they share a common information model, they can be used separately or together (e.g., a declarative policy could call an ECA policy). This provides two different types of abstractions that serve different use cases. It also helps prove the genericity of the GPIM.

2.4.1. Event-Condition-Action Policy Information Model

The SUPA ECA Policy Rule Information Model (EPRIM) represents a policy rule as a statement that consists of an event clause, a condition clause, and an action clause. An ECA policy rule is activated when its event clause is true; the condition clause is then evaluated and, if true, signals the execution of one or more actions in the action clause. This type of Policy Rule explicitly defines the current and desired states of the system being managed.

2.4.2. Declarative Policy Information Model

The SUPA Logic Statement Information Model (LSIM) is a set of (logic-based) propositions that form a (single) conclusion. A proposition is a type of statement that is either TRUE or FALSE. A proposition can be created from simpler propositions. This version of the LSIM defines two forms of SUPA Logic Statements: one using propositional logic, and one using first order logic.

3. Application of Generic Policy-based Management

This section provides examples of how SUPA can be used to define different types of policies. Examples applied to various domains, including system management, operations management, access control, routing, and service function chaining, are also included.

3.1. Declarative Examples

Declarative policies are policies that describe what to do, but not how to do it. Declarative policies can apply to services and/or resources. Here are some simple examples:

System and Operations Management Examples

All routers and switches must have password login disabled.

The above policy first resolves 'routers and switches' to a set of network elements, and then pushes the appropriate configuration to those network elements.

All SNMP agents must enable SNMPv3 and must disable all other versions of SNMP.

The above policy can be mapped to the leafs v1, v2c, and v3 in the ietf-snmp YANG data model (RFC 7407).

All SNMP traffic is dropped unless it originates from, or is directed to, an interface of a management system.

The above policy first resolves a management system interface to a list of IP addresses, and then creates a set of suitable ACL rules that are configured on all network elements.

Access to source code servers is limited to authorized Intranet users.

The above policy assumes that the user is authenticated and authorized to access the code server. It places an additional constraint of requiring Intranet acces before granting access to the resource. Note that this rule is not limited to any one specific user or type of application.

Periodically perform workload consolidation if average CPU utilization falls below X%.

This policy moves workloads on a set of source VMs to a common target VM if the average CPU utilization for the CPUs on the source VM is less than a predefined threshold. Note that the policy did not specify which particular VM to move the workload on the source VM to; that is part of the search and optimization algorithms that are implied, but not specified, by this policy.

Service Management Examples

Proactively monitor Gold Service users to ensure their SLAs are not violated.

Gold Service is an aggregation of different traffic types, each with different constraints. The policy will dynamically create a service function chain based on the current context to ensure that the customer's SLA is not violated.

Gold and Platinum Service Users must have WAN optimization applied to multimedia applications.

The above policy applies only to multimedia applications for users whose SLA types are either Gold or Platinum. It installs a service chain that performs WAN optimization (and likely content caching and other services) to ensure that the SLAs of these users are not violated.

3.2. ECA Examples

ECA policies are statements that consist of an event clause, a condition clause, and an action clause.

Network Service Management Example

Event:	too many interface alarms received from an L3VPN service
Condition:	alarms resolve to the same interface within a specified time period
Action:	if error rate exceeds x% then put L3VPN service to Error State and migrate users to one or more new L3VPNs

Security Management Example

Event:	anomalous traffic detected in network
Condition:	determine the severity of the traffic
Action:	apply one or more actions to affected NEs based on the type of the traffic detected (along with other factors, such as the type of resource being attacked if the traffic is determined to be an attack)

Traffic Management Examples

Event:	edge link close to being overloaded by incoming traffic
Condition:	if link utilization exceeds Y% or if link utilization average is increasing over a specified time period
Action:	change routing configuration to other peers that have better metrics

```
         Event:     edge link close to be overloaded by
                    outgoing traffic
         Condition: if link utilization exceeds Z% or if link
                    utilization average is increasing over a
                    specified time period
         Action:    reconfigure affected nodes to use source-based
                    routing to balance traffic across multiple links
```

Service Management Examples

```
         Event:     alarm received or periodic time period check
         Condition: CPU utilization level comparison
         Action:    no violation: no action
                    violation:
                      1) determine workload profile in time interval
                      2) determine complementary workloads (e.g.,
                         whose peaks are at different times in day)
                      3) combine workloads (e.g., using integer
                         programming)

         Event:     alarm received or periodic time check
         Condition: if DSCP == AFxy and
                    throughput < T% or packet loss > P%
         Action:    no: no action
                    yes: remark to AFx'y'; reconfigure queuing;
                    configure shaping to S pps; ...
```

Note: it is possible to construct an ECA policy rule that is directly tied to configuration parameters; this is in general not possible for declarative policy. The value of declarative policy is in expression of the goal of the policy, and the freedom in implementing that goal. The value of ECA is in more clearly specifying what needs to be done.

3.3. ECA plus Declarative Example

The fundamental reason that SUPA defines two different types of policy rules is to enable different actors to express policy in a manner conducive to their roles. The SGPIM defines concepts that are common to both the EPRIM and the SLSIM. This enables these two types of policies to be used together to provide a more powerful definition of the goals of the policy as well as how to implement those goals.

For example, compare the ECA and declarative forms of the SLA Service Management Policy:

```
Declarative form:
  Proactively monitor Gold Service users to ensure their SLAs
  are not violated.

ECA form:
    Event:     alarm received or periodic time check
    Condition: if DSCP == AFxy and
               throughput < T% or packet loss > P%
    Action:    no: no action
               yes: remark to AFx'y'; reconfigure queuing;
               configure shaping to S pps; ...
```

The declarative policy is more abstract than its ECA counterpart, since the declarative version expresses intent without defining which specific network elements are affected and how the configuration of those network elements should be changed. The above ECA policy rule is written in a high-level form, but note that it still is specifying how to monitor the Gold Service, how to determine if the SLA is being violated, and which actions to take.

The execution of the declarative example could result in one or more ECA policy rules being triggered, such as the one above. Similarly, an ECA policy rule could trigger additional ECA policy rules to be evaluated. For example, the above ECA rule could be rewritten so that if the condition was satisfied, then each of the actions shown could be their own policy rules. This provides additional flexibility through reusing policy rules and the components of policy rules.

4. Related Work

4.1. Related Work within the IETF

4.1.1. I2RS Working Group

I2RS defines an interface that interacts with the routing system using a collection of protocol-based control or management interfaces. Users of I2RS interfaces are typically management applications and controllers. SUPA does not directly interface to the routing system. Rather, SUPA uses data produced by I2RS (e.g., topological information) to construct its policies.

4.1.2. L3SM Working Group

L3SM defines an L3 VPN service model that can be used for communication between customers and network operators. This model enables an orchestration application or customers to request network services provided by L3 VPN technologies. The implementation of network services is often guided by specific policies, and SUPA provides a tool that can help with the mapping of L3 VPN service requests to L3 VPN configurations of network elements.

4.1.3. ALTO Working Group

The ALTO working group defined an architecture for exposing topology information, more specifically the cost of paths through an infrastructure, as defined in [RFC7285]. ALTO services are able to provide network maps defined as groups of endpoints, and can therefore represent any granularity of network, from the physical to groups of networks following similar paths or restraints. Although this model can represent different levels of granularities, it is not clear if it could be adapted easily for other purposes than providing cost maps in the context of ALTO. The ALTO model is meant to be used outside of the trust domain of an ISP by external clients.

SUPA does not generate data that is similar to ALTO. Rather, SUPA could use ALTO data as part of its policies to configure services and/or resources.

4.1.4. TEAS Working Group

The Traffic Engineering Architecture and Signaling (TEAS) working group is responsible for defining MPLS- and GMPLS-based Traffic Engineering architectures that enable operators to control how specific traffic flows are treated within their networks. It covers YANG models for a traffic engineering database. In coordination with other working groups (I2RS) providing YANG models for network topologies.

Both TEAS and SUPA use YANG data models. SUPA does not generate traffic engineering (TE) data. However, SUPA could use TE data as part of its policies for configuring resources and/or services. SUPA could also define policies that define which service, path, and link properties to use for a given customer, and consequently, which protocol extensions to use. TEAS data could also be used to enable operators to define how particular traffic flows are treated in a more abstract (but still consistent) manner.

4.1.5. BESS Working Group

The BGP Enabled Services (BESS) working group defines and extends network services that are based on BGP. This includes BGP/MPLS IP provider-provisioned L3VPNs, L2VPNs, BGP-enabled VPN solutions for use in data center networking, and extensions to BGP-enabled solutions to construct virtual topologies in support of services such as Service Function Chaining. The working group is also chartered to work on BGP extensions to YANG models and data models for BGP-enabled services.

Both BESS and SUPA use YANG data models. SUPA could generate BGP configurations by using data defined by BESS as part of its policies for configuring resources and/or services.

SUPA could also define policies that govern different aspects of services defined by BESS.

4.1.6. SFC Working Group

The Service Function Chaining (SFC) working group defines a mechanism where traffic is classified; that classification is then use to select an ordered set of services to pass the traffic through.

Both SFC and SUPA use YANG data models. SUPA could define policies that augment the functionality of SFC in several different ways, including: (1) path selection based on context, (2) which set of mechanisms to use to steer traffic through which set of service functions, (3) simplify the definition of dynamic service function chains (e.g., service paths that change based upon a set of data that is discovered at runtime), and (4) scalable mechanisms to monitor and control the configuration of SFC components.

4.1.7. NVO3 Working Group

The NVO3 group proposes a way to virtualize the network edge for data centers in order to be able to move virtual instances without impacting their network configuration. This is realized through a centrally controlled overlay layer-3 network. The NVO3 work is not about defining policy information; rather, it uses policy information to perform some functions. Both NVO3 and SUPA use YANG data models. SUPA could define policies that define how the logically centralized network virtualization management entity (or entities) of NVO3 behave (e.g., the functions in the network virtualization control plane).

4.1.8. ACTN BoF (IETF-90)

The ACTN proposed work, as described in [actn] framework, has two main goals, the abstraction of multiple optical transport domains into a single controller offering a common abstract topology, and the splitting of that topology into abstract client views that are usually a fraction of the complete network. The ACTN work is therefore about unification of several physical controllers into a virtual one, and also about the segmentation, isolation and sharing of network resources. The ACTN work is not about defining policy information. Both ACTN and SUPA use YANG data models. SUPA could define policies that define the behavior of the controller.

4.1.9. Previous IETF Policy Models

SUPA is technology-neutral, previous RFCs weren't. SUPA defines a common structure from which both ECA and declarative policies can be defined and combined; this was not possible in previous RFCs. Previous RFCs do NOT define metadata, and do NOT enable policies to formally define obligation, permission, and related concepts. Finally, SUPA uses software patterns, which previous RFCs didn't.

4.2. Related Work outside the IETF

4.2.1. TM Forum

The TM Forum (a.k.a., the TeleManagement Forum) develops standards and best practices, research, and collaborative programs focused on digital business transformation. It consists of three major programs:

1) Agile Business and IT
2) Customer Centricity (experience)
3) Open Digital Ecosystem

Of these, the ZOOM (Zero-touch Orchestration, Operations, and Management) project, located in the Agile Business and IT project, is the main sub-project in this area that is of interest to SUPA.

Within ZOOM, the Foundational Studies project contains work on an information model and management architecture that are directly relevant to SUPA. The TMF Information Model, Policy, and Security working groups are involved in this work.

The ZOOM information model updates the existing Shared Information and Data (SID) information model to add support for the management of physical and virtual infrastructure, event- and data-driven systems, policy management (architecture and model), metadata for describing and prescribing behavior that can support changes at runtime, and access control. The policy information model defines imperative (ECA), declarative (intent-based), utility function, and promise policies. The work in [ID.draft-strassner-supa-generic-policy-info-model] is based on this work. It currently extends the ZOOM ECA model and provides additional detail not currently present in ZOOM; the next version of this draft will do the same for declarative policies.

There is currently no plan to use the utility function and promise policies of ZOOM in SUPA. Finally, it should be noted that the data model work planned for SUPA is not currently planned for the ZOOM project.

4.2.2. MEF

The MEF (originally named the Metro Ethernet Forum) develops architecture, service and management specifications related to Carrier Ethernet (CE). The CE architecture includes the definition of several interfaces specific to CE like the User Network Interface (UNI) and External Network Network Interface (ENNI). Specifications developed in this space include the definitions of CE services, CE service attributes, Ethernet Access Services, Class of Service, OAM and Management interfaces, Service Activation and Test. The more recent vision of the MEF is described as The Third Network, and includes plans to develop Lifecycle Service Orchestration with APIs for existing network, NFV, and SDN implementations enabling Agile, Assured, and Orchestrated Services. This stage of the MEF activity is now in early phases with focus on architectural work.

The MEF has developed a number of Information and Data Models, and has recently started a project that used YANG to model and manage the services defined by the MEF. While the MEF has created rigorous definitions of these services, they are specific to transport technology, and they do not include and rely on policies.

4.2.3. Open Daylight

Open Daylight network controller implements a number of models through its service abstraction Layer (MD-SAL) based on draft IETF Yang models. Open Daylight is an open source project. Two of these are relevant to SUPA, and are described below.

4.2.3.1. Network Intent Composition (NIC)

The Network Intent Composition project aims at providing better flexibility by using declarative policies. It does not cover other types of policies, such as ECA policy rules. The intent-based interface aims to provide a high level of abstraction, primarily for use by an application developer. Its progress has recently stalled.

4.2.3.2. Group Based Policy

The Group Based Policy project defines an application-centric policy model for Open Daylight that separates information about application connectivity requirements from information about the underlying details of the network infrastructure. The model is positioned as declarative, but uses a relational approach to specifying policy.

4.2.4. Open Networking Foundation

The ONF created a group responsible of defining northbound interfaces, but this hasn't lead to the publication of standards in this area so far. A blog entry on the ONF web site showed an interest in using the principle of intents at ONF, but no details were provided on the status of this project. A members-only whitepaper was recently published.

4.2.5. OpenStack

OpenStack software controls large pools of compute, storage, and networking resources throughout a datacenter, managed through a dashboard or via the OpenStack API. OpenStack works with popular enterprise and open source technologies making it ideal for heterogeneous infrastructure. Few of the below mentioned OpenStack projects provides policy abstraction and better flexibility to the user.

4.2.5.1. Group-Based Policy

The Group-Based Policy project for OpenStack Neutron is built around entities assembled in Endpoints Groups (EPG) that provide or consume Contracts. Such Contracts are hierarchical entities containing policy rules. A first version was released in January 2015, based on the Juno release. This type of approach is more relational than declarative, but could be used to describe a large amount of possible scenarios. It has the advantage of providing a relatively simple policy model that covers a large applicability. From an OpenStack point of view, the scope of Group-Based Policies is limited to networking

within the Neutron module.

4.2.5.2. Congress

The Congress project within OpenStack provides a way to define complex policies using extensions to the Datalog language. Datalog is entirely declarative, and its evaluation is based on first-order logic with restrictions. This gives it interesting properties, such as providing the same result no matter the order in which the statements are made. The language allows for the definition of types and for active enforcement or verification of the policies.

There is a significant body of knowledge and experience relating to declarative languages and their implementation. Congress policies aim at manipulating objects exposed by multiple OpenStack modules, and is therefore larger in scope than network element policies.

The declarative policies of SUPA are similar to those in Congress; the primary difference relies in the characteristics and behavior (in the sense of restrictions) of the underlying logic for Congress vs. SUPA. SUPA's propositional logic statements are simpler but more limited than Congress, while SUPA's first-order logic statements are more complex but more powerful than those of Congress. If desired, a Congress model could be easily added to SUPA.

4.2.6. The NEMO Project (not a BoF yet)

The NEMO project is a research activity aimed at defining a simple framework for "intent-based" networking. This project concentrates on creating a domain-specific language and associated API, not a model or even a rigorous definition of what a policy rule is.

The NEMO syntax defines a very simple information model that has three basic elements for network manipulation: nodes, links, and flows. A policy rule is NOT defined in this model. Rather, policy is defined as a command. The NEMO project has been successfully demonstrated at IETF-91, along with a companion graphical user interface.

NEMO declarative policies are different than SUPA declarative policies. NEMO uses a flatter, simpler object model with fewer objects to represent targets of policy. NEMO does not define a policy model, and does not support ECA policies. NEMO uses a condition-action paradigm to execute its declarative policies. In contrast, SUPA uses a richer class model to represent ECA and declarative policies. SUPA declarative policies are executed using

formal logic. SUPA has not proposed a language.

4.2.7. The Floodlight Project

The Floodlight is an OpenFlow-enabled SDN controller. It uses another open source project called Indigo to support OpenFlow and manage southbound devices. The Indigo agent also supports an abstraction layer to make it easy to integrate with physical and virtual switches. It supports configuration of an abstraction layer so that it can configure OpenFlow in hybrid mode.

4.2.8. The ONOS Project

The ONOS is an SDN controller design for Service Provider networks. It uses a distributed architecture, and supports abstraction for both southbound and northbound interfaces. Its modules are managed as OSGi bundles. It is an open source project.

ONOS announced an "application-intent framework", which is similar in nature to SUPA's declarative policies. However, no object model or language has been defined yet.

5. Conclusions: the Value of SUPA

SUPA) defines an interface to a network management function that takes high-level, possibly network-wide policies as input and creates element configuration snippets as output. SUPA expresses policies using a generic policy information model, and produces generic policy YANG data models. SUPA focuses on management policies that control the configuration of network elements. Management policies can be interpreted outside of network elements, and the interpretation typically results in configuration changes of collections of network elements.

Policies embedded in the configuration of network elements are not in the scope of SUPA. In contrast to policies targeted by SUPA, embedded policies are usually interpreted on network elements in isolation, and often at timescales that require the representation of embedded policies to be optimized for a specific purpose.

The SUPA information model generalizes common concepts from multiple technology-specific data models, and makes it reusable. Conceptually, SUPA can be used to interface and manage existing and future data models produced by other IETF working groups. In addition, by defining an object-oriented information model with metdata, the characteristics and behavior of data models can be better defined.

6. Security Considerations

TBD.

7. IANA Considerations

This document has no actions for IANA.

8. Contributors

The following people all contributed to creating this document:

Jun Bi, Tsinghua University
Vikram Choudhary, Huawei Technologies
Luis M. Contreras, Telefonica I+D
Georgios Karagiannis, Huawei Technologies
Hosnieh Rafiee, Huawei Technologies Duesseldorf GmbH
Dan Romascanu, Avaya
Jon Saperia, JDS Consulting
J. Schoenwaelder, Jacobs University, Germany
Qiong Sun, China Telecom
Parviz Yegani, Juniper Networks
Cathy Zhou, Huawei Technologies

9. Acknowledgments

This document has benefited from reviews, suggestions, comments and proposed text provided by the following members, listed in alphabetical order: J. Bi, Luis M. Contreras, G. Karagiannis, D. Romascanu, J. Saperia, J. Schoenwaelder, Q. Sun, P. Yegani, and C. Zhou.

10. References

10.1. Informative References

[RFC3198] Westerinen, A., Schnizlein, J., Strassner, J., Scherling, M., Quinn, B., Herzog, S., Huynh, A., Carlson, M., Perry, J., Waldbusser, S., "Terminology for Policy-Based Management", RFC 3198, November, 2001

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

[RFC6991] J. Schoenwaelder, "Common YANG Data Types", July 2013

[RFC6241] R. Enns, M. Bjorklund, J. Schoenwaelder, A. Bierman, "Network Configuration Protocol (NETCONF)", June 2011

[RFC7285] R. Alimi, R. Penno, Y. Yang, S. Kiesel, S. Previdi, W. Roome, S. Shalunov, R. Woundy "Application-Layer Traffic Optimization (ALTO) Protocol", September 2014
[SUPA-framework] C. Zhou, L. M. Contreras, Q. Sun, and P. Yegani, " The Framework of Simplified Use of Policy Abstractions (SUPA) ", IETF Internet draft, draft-zhou-supa-framework, February 2015.

[SUPA-problem-statement] G. Karagiannis, Q. Sun, Luis M. Contreras, P. Yegani, JF Tremblay and J. Bi, "Problem Statement for Simplified Use of Policy Abstractions (SUPA)", IETF Internet draft, draft-karagiannis-supa-problem-statement, January 2015.

[SUPA-gap-analysis] J. Bi, H.Rafiee, V,Choudhary, J.Strassner, D.Romascanu "Simplified Use of Policy Abstractions (SUPA) Gap Analysis", IETF Internet draft, draft-bi-supa-gap-analysis, May 2015.

[SUPA-DDC] Y. Cheng, and JF. Tremblay, "Use Cases for Distributed Data Center Applications in SUPA", IETF Internet draft, draft-cheng-supa-ddc-use-cases, January 2015

[RaBe11] Raphael Romeikat, Bernhard Bauer, "Formal Specification of DomainSpecific ECA Policy Models", in Proc. 2011 Fifth IEEE International Conference on Theoretical Aspects of Software Engineering, 2011

[Stras02] John Strassner, "DEN-ng: Achieving Business-Driven Network Management" in Proc. IEEE Network Operations and Management Symposium (NOMS), 2002.

[TR235] John Strassner, ed., "ZOOM Policy Architecture and Information Model Snapshot", TR245, part of the TM Forum ZOOM project, October 26, 2014

Authors' Addresses

Maxim Klyus, Ed.
NetCracker
Kozhevnicheskaya str.,7 Bldg. #1
Moscow, Russia
Phone: +7-916-8575717
E-mail: klyus@netcracker.com

John Strassner, Ed.
Huawei Technologies
2330 Central Expressway
Santa Clara, CA 95138 USA
Email: john.sc.strassner@huawei.com