I2NSF
Internet Draft
Intended status: Standard Track

L. Xia
Huawei
D Zhang
Alibaba
N. BOUTHORS
Qosmos

Expires: November 2015

May 25, 2015

Information Model of Interface to Network Security Functions Capability Interface
draft-xia-i2nsf-capability-interface-im-01.txt

Status of this Memo

Copyright Notice

Abstract

This draft is focused on the north-bound interface of NSFs (Network Security Functions) and proposes an information model for configuring various kinds NSF security functions, based on the packet-based paradigm. The Yang structure and use examples are also presented to clarify how to use the information model.

Table of Contents

1. Introduction

Due to the rapid development and deployment of cloud computing services, the demand of cloud-based security services is also rapidly growing. The customers of them can be enterprises [I-D.zarny-i2nsf-data-center-use-cases], User Equipment (UE) of mobile network and Internet of Things (IoT) [I-D.qi-i2nsf-access-network-usecase], residential access users [I-D.pastor-i2nsf-access-usecases], and so on.

Derived from [I-D.dunbar-i2nsf-problem-statement], two types of I2NSF interface should be considered:

- Interface between I2NSF user/client with network/security controller: [I-D.xia-i2nsf-service-interface-DM] describes the information model used by this type of interface. It's a service-oriented interface, the main objective is to unify the communication channel and the security service request information model between various high-level application (e.g., openstack, various BSS/OSS, etc) with various network controllers. The design goal of the service interface is to decouple security service in application layer from various kinds of security devices and their device-level security functions. The intent-based information model approach derived from RBAC model can be a feasible option for it;

- North-bound interface provided by NSFs (e.g., FW, AAA, IPS, Anti-DDOS, Anti-Virus, etc), no matter whether the NSFs are Virtual Machines (VM) on servers or physical appliances. In this document, this type of interface is also called "capability interface". Any network entities (e.g., I2NSF clients, network/security controller, etc) can use this interface to configure the required security functions of NSFs. Current situation is different NSF vendors have different proprietary interfaces and information models for configuring their security functions.

This draft is focused on the capability interfaces and proposes an information model for configuring various kinds NSFs. It's used by the NSFs to decouple from the various security services came from the application layer and highlight the security capabilities they can provide. Section 3 defines the information model for capability interface. Section 4 gives its representation by Yang data model. Section 5 includes some using examples to clarify how to use the information model.

2. Conventions used in this document

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].

2.1. Terminology

AAA -Access control, Authorization, Authentication

ACL - Access Control List

AD - Active Directory

ANSI - American National Standards Institute

DDoS - Distributed Deny of Services

FW - Firewall

I2NSF - Interface to Network Security Functions

INCITS - International Committee for Information Technology Standards

IoT - Internet of Things

IPS - Intrusion Prevention System

LDAP - Lightweight Directory Access Protocol

NAT - Network Address Translation

NBI - North-bound Interface

NIST - National Institute of Standard Technology

NSF - Network Security Function

RBAC - Role Based Access Control

UE - User Equipment

URL - Uniform/Universal Resource Locator

VM - Virtual Machine

3. Information Model for Capability Interface

3.1. Overview

Similar to switchs and routers, NSFs realize the security capabilities (e.g., antivirus, IPS, FW, etc) in the device level, not in the service level. Although in some conditions, they can provide certain service-aware capabilities, i.e., application recognition, virus detection, etc. In other words, the IM of the capability interface should be designed by the way of abstracting from the various specific security capabilities to a generic model, so that it can be used to configure NSFs directly or by the translation of the adaptor in NSF easily.

Below is the overall information model for I2NSF capability interface.

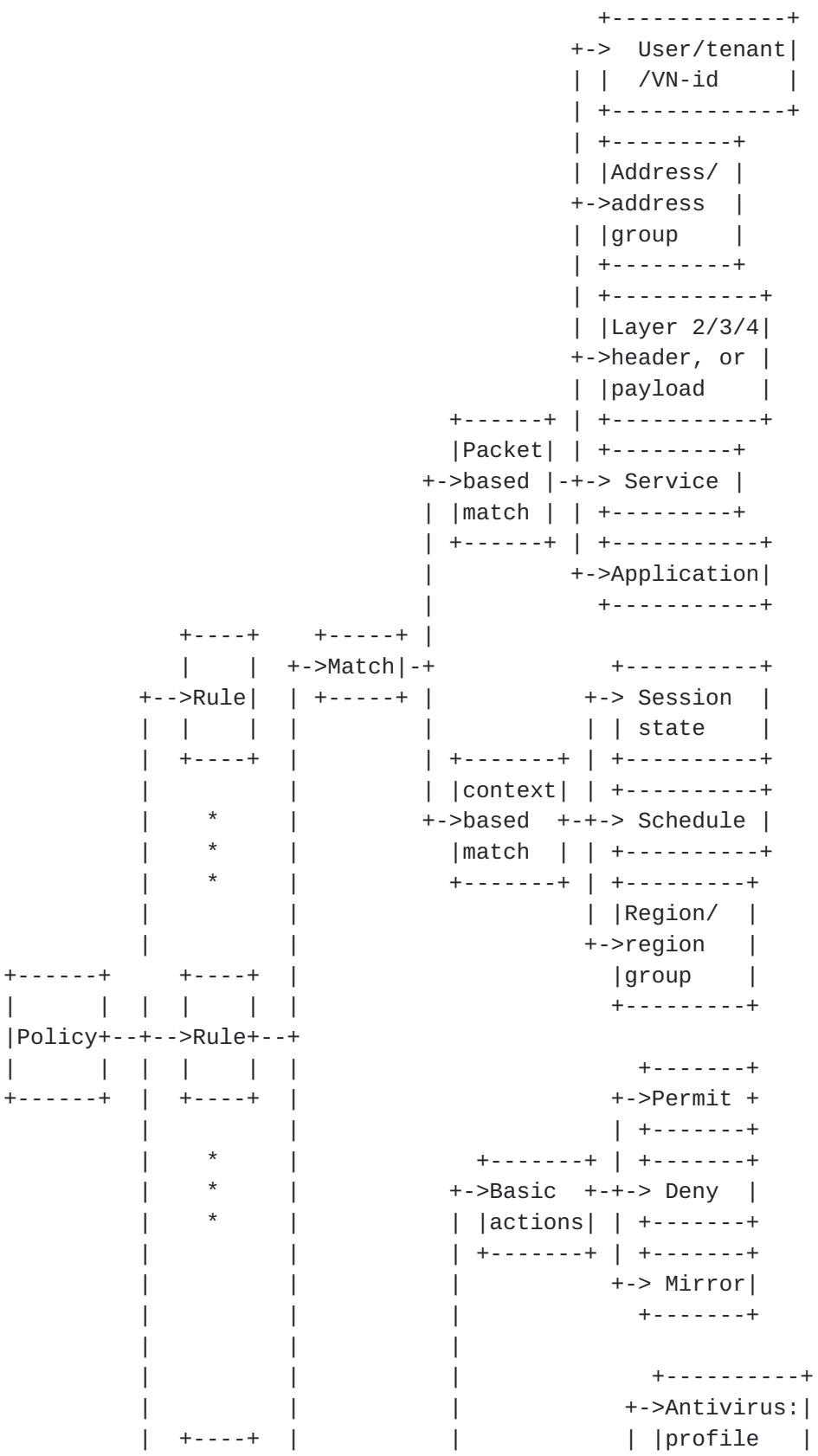
User/tenant
/VN-id
Address/
address
group
Layer 2/3/4
header, or
payload
Packet
based
match
Service
Application
Match
Rule
Session
state
context
based
match
Schedule
Region/
region
group
Policy
Rule
Permit
Basic
actions
Deny
Mirror
Antivirus:
profile

Figure 1. The Overall Information Model for I2NSF Capability Interface

At the top level, policy is a container including a set of security rules. Each rule represents some specific security requirements or actions. Security policy combines these rules together according to some logic, i.e., their similarity or mutual relations, etc.

A Security policy is created and assigned to any NSFs depending on specific requirements and scenarios. For example, a security policy can be responsible for an enterprise branch, or can be used for the access control to one set of services.

3.2. Packet-Based Paradigm

[I-D.lopez-i2nsf-packet] analyzes the common nature of NSF functions that NSFs ultimately are packet-processing engines that inspect packets traversing networks, either directly or in context to sessions to which the packet is associated. This draft uses this packet-based paradigm for the design of NSF capability interface IM.

This packet-based design approach is very general and easily extensible, and so can avoid any potential constraints which could limit NSFs' functional capabilities.

Considering from the perspective of packet-processing, NSFs differ in the depth of packet header or payload they can inspect, the various session/context states they can maintain, and the actions or specific profiles they can apply. Therefore, the NSF capabilities can be characterized by the level of packet-processing and context that a NSF can support, the actions and profiles that the NSF can apply. In the other hand, NSF Vendors can register their provided NSF capabilities by using the Subject-Object-Action-Function categories described by [I-D.lopez-i2nsf-packet].

Table 1-4 below lists some examples included in the categories for constructing the NSF capability:

Subject (packet) Capability Index	
Layer 2 Header	Layer 2 header fields: Source/Destination/s-VID/c-VID/EtherType/.
Layer 3 IPv4 objects	Layer header fields: protocol
	port
	src port
	dscp
	length
	flags
	ttl
IPv6 Object	
	addr
	protocol/nh
	src port
	length
	traffic class
	hop limit
	flow label
TCP	Port

SCTP		syn
DCCP		ack
		fin
		rst
		psh
		urg
		window
		sockstress
UDP		
		flood abuse
		fragment abuse
		Port
HTTP layer		
		hash collision
		http - get flood
		http - post flood
		http - random/invalid url
		http - slowloris
		http - slow read
		http - r-u-dead-yet (rudy)
		http - malformed request
		http - xss
		https - ssl session exhaustion
IETF PCP	Configurable Ports	
IETF TRAM	profile	

Table 1. Subject (packet) Capability Index

Object (context) Capability Index	
Session	Session state, bidirectional state

Time	time span days, minutes, seconds, Events
Events	Event URL, variables

Table 2. Object (context) Capability Index

Actions Capability Index	
Ingress port	SFC header termination , Pass Deny Mirror Functional call Encap various header
Egress	

Table 3. Actions Capability Index

Functional profile (advanced actions) Capability Index	
Profile types	Vendor specific Flexible Profile URL Accept external

Table 4. Functional profile (advanced actions) Capability Index

3.3. Rule

Each rule is defined in the classic "match & action" style that already implemented in most NSFs today to minimize the needed updates on existed NSFs and decrease the complexity.

The NSF follows the rules one by one to process the passing traffic as follows:

1. The NSF analyzes traffics by either one of packet-based match and context-based match, or both of them. Packet-based match inspects the packet header and/or payload to retrieve the traffic attributes at network or application layers. The traffic attributes include user, address, other packet attributes, service and application. Context-based match analyzes and retrieves a variety of contextual attributes associated with the packet such as session state, schedule and region;

2. The NSF compares the attributes with the match conditions defined in the first rule. If all the conditions are met, the traffic matches the rule. If one or more conditions are not met, the NSF compares the attributes with the conditions of objects defined in the next rule. If all rules are not met, the NSF denies the traffic by default;

3. If the traffic matches a rule, the NSF performs the defined basic actions over the traffic. If the basic action is deny, the NSF blocks the traffic. If the basic action is permit/mirror, the NSF resumes checking whether certain advanced actions are referenced in the rule. If yes, go to step 4. If no, the traffic is permitted;

4. If certain advanced actions (e.g., Antivirus, IPS, etc) are referenced in the rule and the basic action defined in the rule is permit/mirror, the NSF performs integrated checks on the content carried over the traffic. The integrated check inspects the content carried over the traffic based on the conditions defined in the referenced profiles of advanced action and implements appropriate actions based on the check result. If any advanced action determines to block the traffic, the NSF blocks the traffic. If all advanced actions determine to permit the traffic, the NSF allows the traffic through.

One rule can be applied multiple times on different places, i.e., links, devices, networks, vpns, etc. It not only guarantees the consistent policy enforcement in the whole network, but also decreases the configuration workload.

3.4. Match

Match (aka, Objects) consists of two categories of match condition: packet-based match and context-based match. Each category includes various match conditions representing different kinds of objects. The logic relation among all the conditions is flexible, it can be "AND", "OR". The former means the traffic must match all the

conditions, while the latter means the traffic only needs to match one of the conditions.

The general objects for packet-based match are as follows:

- User: A user is a person applicaiton who is authorized to access network resources. A user can be an internet access user who accesses Internet resources or intranet resources from inside the intranet through a FW, or a remote access user who connects to a FW in VPN, or PPPoE mode to access intranet resources. The NSFs need to know the IP address or other information (i.e., user's tenant or VN-id) of the user to identify the user's traffic and perform the pre-defined actions. It can also define a group of users to match and perform actions to them together;

- Source and destination address scope;

- Layer 2/3/4 header, or payload related attributes: other meaningful and useful attributes in packet except for existing objects;

- Service: A service is an application identified by a protocol type and port number. It can be a service or a group of services. NSF matches the service traffics based on the protocol types and port numbers and applies the security actions to them;

- Application: An application is a computer program for a specific task or purpose, and multiple applications constitute an application group. It provides a finer granularity than service in matching traffic. Even if different applications have the same service, they still can be distinguished by analyzing the data packets and comparing the signatures of each application. The hierarchy category method is appropriate for identifying applications. For example, the application of Gmail belongs to the category of business systems, and the subcategory of Email. Other key attributes that belongs to and can be used to identify an application are data transmission model (e.g., client-server, browser-based, networking, peer-to-peer, etc), risk level (e.g., Exploitable, Evasive, Data-loss, Bandwidth-consuming, etc).

The general objects for context-based match are as follows:

- Session state: any one specific state related to the user/operation sessions, such as authentication state, TCP/UDP session state, bidirectional state, etc;

- o Schedule: A schedule defines time ranges. A rule can reference a schedule to filter traffic that passes through the NSF within the schedule. A schedule can be a periodic schedule, or a one-time schedule;

- o Region/region group: the logical definition of users' location which can be pre-defined in the in the location signature database by the geographical information, or be manual defined by the user's IP information.

Objects are extensible, new match conditions can be defined and added into them any time according to requirements.

3.5. Actions

The action of a security rule is also divided into two categories logically: basic actions and advanced actions. Basic actions are either permit, deny or mirror. Deny simple means to block the matching traffics. Permit and mirror have more meanings by performing the referenced advanced actions. The all advanced actions in one rule can inspect traffic content during one-pass, which greatly improves system performance.

Every advanced action includes its own matching conditions to identify specific traffic and perform required actions. The advanced action is defined by specific requirements or for specific scenarios. Some typical advanced actions are Antivirus, IPS, URL filtering, File blocking, Data filtering, Application control, and so on.

By combining advanced actions and using them appropriately, NSFs can defend against possible attacks and reduce the waste of system resources.

4. I2NSF Capability Interface IM Yang Structure

This section specifies the I2NSF capability interface information model in Yang structure [RFC6020].

module: Security Policies

```
      +--security-policies

         +--rw policy-set* [policy-name]

            +--rw policy-name string

            +--rw policy-id  uint16
```

```
                +--rw security-rules
                    +--rw rule-set* [rule-name]
                      +--rw rule-name  string
                      +--rw rule-id  uint16
                      +--rw Match
                      |  +--rw packet-based-match
                    |  |  +--rw user* [login-name]
                      |  |  |  +--rw login-name string
                      |  |  |  +--rw display-name string
                      |  |  |  +--rw group-name string
                      |  |  |  +--rw description string
                      |  |  |  +--rw parent-group string
                      |  |  |  +--rw password string
                      |  |  |  +--rw expired-date data-and-time
                      |  |  |  +--rw allow-multi-account-login boolean
                      |  |  |  +--rw address-binding Boolean
                      |  |  |  +--rw tenant? uint32
                      |  |  |  +--rw VN-id? uint32
                      |  |  +--rw address-scope*
                      |  |  |  +--rw src-address inet:ip-prefix
                      |  |  |  +--rw dst-address inet:ip-prefix
                    |  |  +--rw layer-header-payload*
                      |  |  |  ...
                      |  |  +--rw service* [name]
```

```
|  |  |  +--rw name string
|  |  |  +--rw description string
|  |  |  +--rw protocol enumeration
|  |  |  +--rw protocol-num uint8
|  |  |  +--rw src-port-num uint16
|  |  |  +--rw dest-port-num uint16
|  |  +--rw application* [name]
|  |  |  +--rw name string
|  |  |  +--rw server-address inet:ip-address
|  |  |  +--rw protocol enumeration
|  |  |  +--rw dest-port-num uint16
|  |  |  +--rw category enumeration
|  |  |  +--rw subcategory enumeration
|  |  |  +--rw data-transmission-model enumeration
|  |  |  +--rw risk-level enumeration
|  +--rw context-based-match
|     +--rw session-state*
|     |  ...
|     +--rw schedule* [name]
|     |  +--rw name string
|     |  +--rw type enumeration
|     |  +--rw start-time data-and-time
|     |  +--rw end-time data-and-time
|     |  +--rw weekly-validity-time? data-and-time
```

```
                |     +--rw region*
                   |        ...
                  +--rw actions
                     +--rw basic-actions enumeration
                  +--rw advanced-actions* [name]
                     +--rw name string
                        +--rw profile-antivirus?
                        |  ...
                        +--rw profile-IPS?
                        |  ...
                        +--rw profile-url-filtering?
                        |  ...
                        +--rw profile-file-blocking?
                        |  ...
                        +--rw profile-data-filtering?
                        |  ...
                        +--rw profile-application-control?
                        |  ...
```

5. Use Examples of I2NSF Capability Interface IM

TBD

6. Security Considerations

TBD

7. IANA Considerations

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2234] Crocker, D. and Overell, P.(Editors), "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, Internet Mail Consortium and Demon Internet Ltd., November 1997.

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

8.2. Informative References

[INCITS359 RBAC] NIST/INCITS, "American National Standard for Information Technology - Role Based Access Control", INCITS 359, April, 2003

[I-D.zarny-i2nsf-data-center-use-cases] Zarny, M., et.al., "I2NSF Data Center Use Cases", Work in Progress, October 2014.

[I-D.qi-i2nsf-access-network-usecase] Qi, M., et.al., "Integrated Security with Access Network Use Case", Work in Progress, October, 2014.

[I-D.pastor-i2nsf-access-usecases] Pastor, A., et.al., "Access Use Cases for an Open OAM Interface to Virtualized Security Services", Work in Progress, October, 2014.

[I-D.dunbar-i2nsf-problem-statement] Dunbar, L., et.al., "Interface to Network Security Functions Problem Statement", Work in Progress, September, 2014.

[I-D.xia-i2nsf-service-interface-DM] Xia, L., et.al., "Data Model of Interface to Network Security Functions Service Interface", February, 2015.

[I-D.lopez-i2nsf-packet] Lopez, E., "Packet-Based Paradigm For Interfaces To NSFs", March, 2015.

9. Acknowledgments

This document was prepared using 2-Word-v2.0.template.dot.

Authors' Addresses

Liang Xia
Huawei
Email: Frank.xialiang@huawei.com

DaCheng Zhang
Alibaba
Email: Dacheng.zdc@alibaba-inc.com

Nicolas BOUTHORS
Qosmos
Email: Nicolas.BOUTHORS@qosmos.com