

I2nsf Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: January 19, 2017

J. You  
J. Strassner  
Huawei  
M. Zarny  
Independent  
C. Jacquenet  
France Telecom  
S. Majee  
F5 Networks  
July 18, 2016

User-Group-based Security Policy for Capability Layer  
draft-you-i2nsf-user-group-policy-capability-00

Abstract

This draft defines the I2NSF Capability APIs for implementing User-Group-based Security Policies using the Event-Condition-Action Policy Rule paradigm.

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 [RFC2119].

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). 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."

This Internet-Draft will expire on January 19, 2017.

Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
2. Terminology . . . . .	3
2.1. Abbreviations and Acronyms . . . . .	3
2.2. Definitions . . . . .	3
3. Overall Architecture . . . . .	4
3.1. Alternative 1: Using Packet User-Group Labels . . . . .	4
3.2. Alternative 2: Using User-group IDs Directly . . . . .	5
4. ECA for User-group-based Security Policy . . . . .	5
4.1. Information Model Design . . . . .	6
4.2. FlowSpecECAPolicyRule Class Definition . . . . .	6
4.3. Event . . . . .	8
4.4. Condition . . . . .	8
4.5. Action . . . . .	8
4.5.1. Traffic-rate Action . . . . .	8
4.5.2. Traffic-detail Action . . . . .	9
4.5.3. Redirect Action . . . . .	9
4.5.4. Traffic-marking Action. . . . .	9
4.6. Capability Layer Rules . . . . .	9
5. Security Considerations . . . . .	9
6. IANA Considerations . . . . .	9
7. Acknowledgements . . . . .	9
8. References . . . . .	9
8.1. Normative References . . . . .	9
8.2. Informative References . . . . .	10
Authors' Addresses . . . . .	11

## 1. Introduction

In traditional networks, network access is typically controlled through a combination of mechanisms. These include maintaining separate static VLAN/IP subnet assignments per organization, applying Control Lists (ACLs) on VLANs and/or IP subnets, and leveraging Network Access Control(NAC). However, traditional access control mechanisms ([I-D.ietf-i2nsf-problem-and-use-cases]) do not work well with newer network paradigms and architectures. Most network administrators have to manually plan and implement necessary changes because little, if any, automation exists across diverse sets of network security platforms.

[I-D.you-i2nsf-user-group-based-policy] discusses User-Group aware Policy Control (UAPC). This facilitates consistent enforcement of policies based on User-Group identity, since the User-Group IDs are generated by a set of ECA Policy Rules. It also discusses how this can be used in the I2NSF Service Layer [I-D.ietf-i2nsf-framework]. The UAPC mechanism calls for: (1) a User-Group identifier (e.g., source and destination IP address, time-of-day, device certificate, or a combination of these and other similar attributes); (2) a policy server service that maintains policies for defining and managing user-groups as well as permissions associated with user-groups; and (3) a logical security controller that is responsible for managing Network Security Functions (NSFs), and installing necessary policies on them.

This document proposes I2NSF capability layer APIs for implementing User-Group-based security policies by using the Event-Condition-Action (ECA) Policy Rule paradigm.

## 2. Terminology

This section contains terminology and definitions that are important for understanding the technical content of this document.

### 2.1. Abbreviations and Acronyms

AAA: Authentication, Authorization, and Accounting  
ECA: Event-Condition-Action  
NSF: Network Security Function  
UAPC: User-Group Aware Policy Control  
VRF: Virtual Routing and Forwarding instance

### 2.2. Definitions

**Event:** An event is defined as any important occurrence in time of a change in the system being managed, and/or in the environment of the system being managed. An Event, when used in the context of a Policy Rule, is used to determine whether the Condition clause of an imperative Policy Rule can be evaluated or not. For an ECA Policy Rule, if the Event clause is TRUE, then the Condition Clause MUST be evaluated.

**Condition:** A set of attributes, features, and/or values that are to be compared with a set of known attributes, features, and/or values in order to make a decision. A Condition, when used in the context of a Policy Rule, is used to determine whether or not the set of Actions in that Policy Rule can be executed or not. For an ECA Policy Rule, if the Condition clause is TRUE, then at least one Action contained in this ECA Policy Rule MUST be evaluated.

Action: An Action is a set of purposeful activity that has associated behavior. An Action, when used in the context of a Policy Rule, may be executed when both the Event and the Condition clauses of its owning Policy Rule evaluate to true. The execution of this Action MAY be influenced by applicable metadata.

### 3. Overall Architecture

The Security Controller coordinates various network security-related tasks on a set of NSFs under its administration, and invokes the set of NSFs that are required to implement security for particular packets.

The NSF may match on User-Group IDs in the packets, or it may match on common packet header fields such as an n-tuple, and then map the n-tuple to the appropriate User-Group ID supplied out-of-band by the Security Controller. If the packet matches a specified User-Group ID, the NSF enforces the corresponding policies.

The interface between the Security Controller and the NSF is called the capability interface in the I2NSF context. This document describes the I2NSF capability layer API for implementing User-Group-based security policies by using policy rules that are defined in an Event-Condition-Action (ECA) structure.

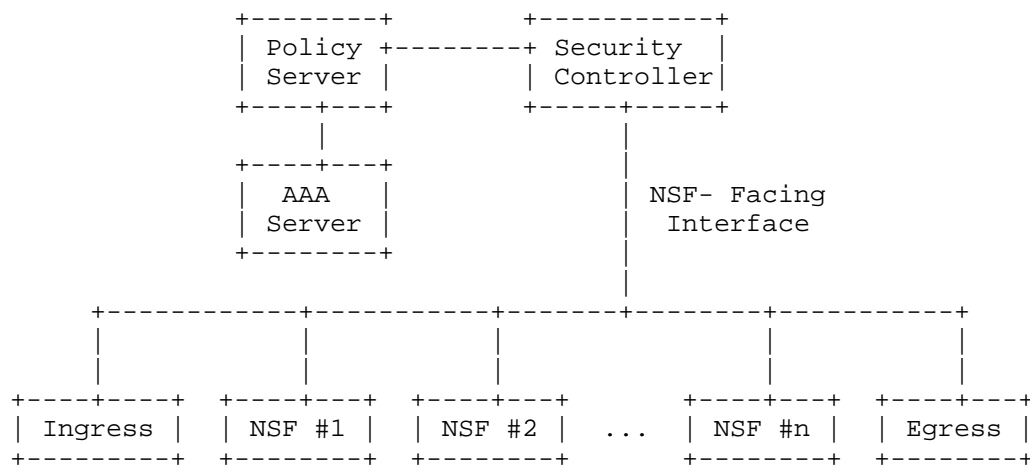


Figure 1. Functional Architecture

#### 3.1. Option 1: Using Packet User-Group Labels

This option employs the User-Group Label (UGL), a packet header field, to carry the User-Group ID. UGLs are disseminated using the Registration Interface (see [I-D.ietf-i2nsf-terminology] and [I-D.ietf-i2nsf-framework]).

UGLs work in a 'UGL-capable' domain. This is a domain in which all NSFs residing in that domain support both the UGL field as well as mapping the UGL field (and hence, the User-Group ID that the UGL field represents) to a set of ECA Policy Rules.

In this alternative, the ingress NSF matches on common packet header fields, such as an n-tuple, and then maps the n-tuple to the appropriate User-Group ID supplied out-of-band by the Security Controller. Then, the ingress NSF inserts the corresponding UGL into the packet. The UGL is used by NSFs to make forwarding decisions in the UGL-supported domain.

The UGL field can be inserted into the packet using, for example, SFC encapsulation [I-D.ietf-sfc-nsh]. One benefit of option 1 is that only edge NSFs need to do UGL insertion or removal; other NSFs can easily enforce User-Group based policies based on the UGL carried in the packet. If the UGL field does not match any policy, a UGL-compliant NSF may apply a default policy, such as dropping or forwarding, to the packet. For example, if no policy is matched, firewalls typically will drop the packet, since Firewalls define a deny action as the default action to use.

### 3.2. Option 2: Using User-Group IDs Directly

This alternative option relies on all NSFs being able to identify appropriate fields in a packet, map them to an appropriate User-Group, and then use that User-Group to control actions on the packet. In this option, User-Groups are disseminated using the Registration Interface.

The Ingress NSF matches on common packet header fields, such as an n-tuple, and then maps the n-tuple to the appropriate User-Group ID (which is supplied out-of-band by the Security Controller). Then, the ingress NSF enforces User-Group Policy Rules based on the identified User-Group ID, without inserting any field into the packet. Hence, option 2 requires all NSFs on the forwarding path to support User-Group mapping. If the NSF fails to associate the packet with a User-Group, it may use a default Policy Rule that is associated with, for example, a default unknown User-Group. The Policy associated with the default unknown User-Group should then be enforced. For example, an unknown User-Group could be mapped to a null VLAN, and a message sent to the appropriate Controller to perform basic security checking on it and then assign a real User-Group to the packet or flow (and a corresponding VLAN).

## 4. ECA for User-group-based Security Policy

This document uses Policy Rules, in the form of an Event-Condition-Action (ECA) structure, to describe User-Group-based security policies.

The fundamental constructs of ECA languages are reactive rules of the form:

```

    IF the event clause evaluates to TRUE
      IF the condition clause evaluates to TRUE
        THEN execute actions in the action clause
      ENDIF
    ENDIF
  
```

The Event clause, Condition clause, and Action clause collectively form a three-tuple. The Event and Condition clauses are made up of one or more Boolean statements. If the Event clause evaluates to FALSE, execution stops. If the Event clause evaluates to TRUE, then the Condition clause is evaluated. Once again, if the Condition clause evaluates to FALSE, execution stops; otherwise, Actions in the Policy Rule may be executed.

#### 4.1. Information Model Design

The information model design for User-Group-based Security Policies is based on the information model design of the capability interface [I-D.draft-xia-i2nsf-capability-interface-im].

It is assumed that an external model, or set of models, is used to define the concept of an ECA Policy Rule and its components (e.g., Event, Condition, and Action objects). A functional block diagram of the ECA Model is shown in Figure 2 below:

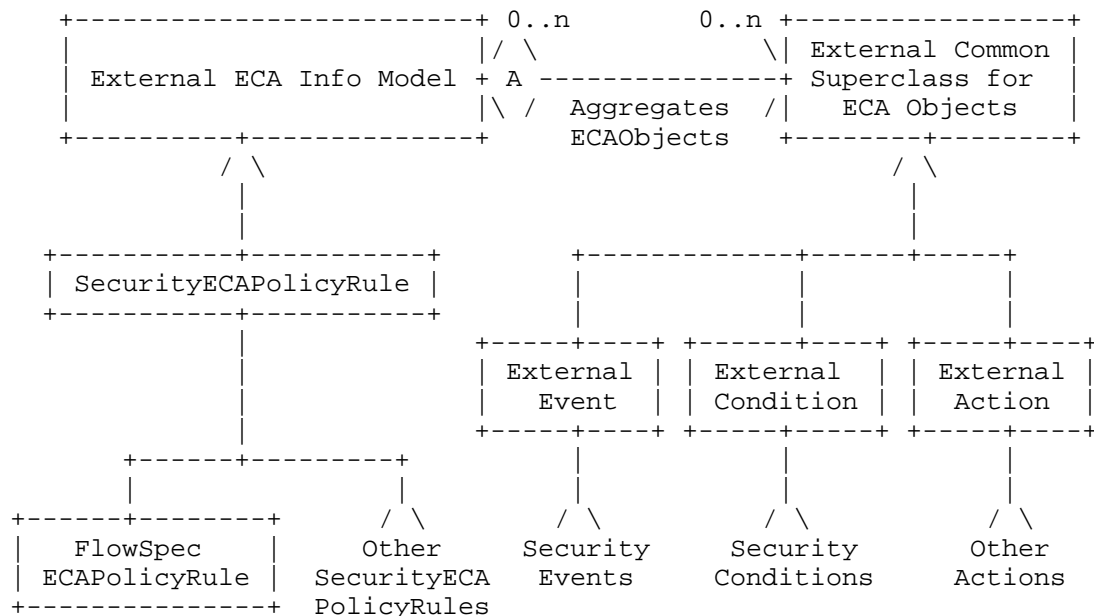


Figure 2. High-Level Information Model Design

The SecurityECAPolicyRule is the top of the User-Group ECA Policy Rule hierarchy. It inherits from the (external) generic ECA Policy Rule to define Security ECA Policy Rules that are specific to managing User-Groups. The SecurityECAPolicyRule contains all of the attributes, methods, and relationships defined in its (generic) superclass, and adds additional concepts that are required for Network Security (these will be defined in the next version of this draft).

The AggregatesECAObjects relationship defines the set of Events, Conditions, and Actions that are aggregated by a specific ECAPolicyRule. This aggregation is inherited by all subclasses of SecurityECAPolicyRule.

This draft defines a single subclass of SecurityECAPolicyRule, called FlowSpecECAPolicyRule. Additional ECA Policy Rules will be defined in future versions of this draft.

This draft defines four subclasses of the (external) generic Action class. Additional Event, Condition, and Action subclasses will be defined in future versions of this draft. Note that any of the Events, Conditions, and Actions defined in [I-D.draft-xia-i2nsf-capability-interface-im] can also be used; this is because the FlowSpecECAPolicyRule is really a container, which is meant to aggregate Events, Conditions, and Actions. The Flow Spec Rule is thus generic; application-specific needs are provided by choosing a suitable set of Events, Conditions, and Actions.

It is assumed that the (external) generic ECAPolicyRule class defines basic information in the form of attributes, such as an unique object ID, as well as a description and other basic, but necessary, information. It is also assumed that the (external) generic ECA Policy Rule is abstract; the SecurityECAPolicyRule is also abstract. This enables data model optimizations to be made while making this information model detailed but flexible and extensible.

The SecurityECAPolicyRule defines network security policy as a container that aggregates Event, Condition, and Action objects, which are described in Sections 4.3, 4.4, and 4.5, respectively. Events, Conditions, and Actions can be generic or security-specific.

Brief class descriptions of these classes are provided in the following sub-sections. In addition, none of the ECAPolicyRule subclasses will define attributes. This enables them to be viewed as simple object containers, and hence, applicable to a wide variety of content. It also means that the content of the function is defined solely by the set of Events, Conditions, and Actions that are contained by the particular subclass. This enables the Policy Rule, with its aggregated set of Events, Conditions, and Actions, to be treated as a reusable object.

#### 4.2. FlowSpecECAPolicyRule Class Definition

The purpose of a FlowSpecECAPolicyRule is to define an ECA Policy Rule that can invoke FlowSpecs as Actions. The set of invoked Actions are triggered by a set of Events and Conditions. This ECA Policy Rule serves as a reusable container, and hence, the set of Events, Conditions, and Actions that it aggregates are also reused.

#### 4.3. Event

Events are external stimuli, such as alarms, user actions (e.g., logon and logoff, or access requests), and packet arrival or departure occurrences. A set of exemplary Events are defined in [I-D.draft-xia-i2nsf-capability-interface-im].

#### 4.4. Condition

Conditions are typically attributes or values that affect the state of a managed entity. These include:

- n-tuple of the incoming packet
- cross checking with other data, such as correlation with packets received from different ports or past time, or
- the current state of a flow

A set of exemplary Conditions are defined in [I-D.draft-xia-i2nsf-capability-interface-im].

#### 4.5. Action

This document defines a minimum set of Actions. The first set of Actions are, based on [RFC5575] and [RFC7674]. This is not meant to be an inclusive list of all possible Actions, but only a subset that pertain specifically to flows, which can be interpreted consistently across the network. Other Actions will be defined in future versions of this document. An exemplary set of additional Actions is defined in [I-D.draft-xia-i2nsf-capability-interface-im].

Note that [RFC5575] and [RFC7674] define a general procedure to encode flow specification rules for aggregated traffic flows, so that they can be distributed as BGP [RFC4271] network layer reachability information.

##### 4.5.1. Traffic-rate Action

The traffic-rate instructs a system to shape a certain stream to a set of predefined bandwidth characteristics. This is implemented using BGP extended community values attributes [RFC4360]. A traffic-rate of 0 should result in all traffic for this particular flow to be discarded.

#### 4.5.2. Traffic-detail Action

The traffic-detail enables traffic sampling and logging for a particular flow. This is implemented using BGP extended community values attributes [RFC4360].

#### 4.5.3. Redirect Action

The Redirect allows the traffic to be redirected to a specified VRF routing instance that lists the specified route-target in its import policy. This is implemented using BGP extended community values attributes [RFC4360].

#### 4.5.4. Traffic-marking Action

The Traffic-marking instructs a system to modify the DSCP bits of a transiting IP packet to the corresponding value. This is implemented using BGP extended community values attributes [RFC4360].

#### 4.6. Capability Layer Rules

This will be completed in the next version of this document.

### 5. Security Considerations

This document provides an Event-Condition-Action model for describing user-group-based security policies. It is not intended to represent any particular system design or implementation, nor does it define a protocol, and as such it does not have any specific security requirements.

### 6. IANA Considerations

This document has no actions for IANA.

### 7. Acknowledgements

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, <<http://www.rfc-editor.org/info/rfc2119>>.

- [RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <<http://www.rfc-editor.org/info/rfc4271>>.
- [RFC4360] Sangli, S., Tappan, D., and Y. Rekhter, "BGP Extended Communities Attribute", RFC 4360, DOI 10.17487/RFC4360, February 2006, <<http://www.rfc-editor.org/info/rfc4360>>.
- [RFC5575] Marques, P., Sheth, N., Raszuk, R., Greene, B., Mauch, J., and D. McPherson, "Dissemination of Flow Specification Rules", RFC 5575, DOI 10.17487/RFC5575, August 2009, <<http://www.rfc-editor.org/info/rfc5575>>.
- [RFC7674] Haas, J., Ed., "Clarification of the Flowspec Redirect Extended Community", RFC 7674, DOI 10.17487/RFC7674, October 2015, <<http://www.rfc-editor.org/info/rfc7674>>.

## 8.2. Informative References

- [I-D.ietf-i2nsf-problem-and-use-cases]  
Hares, S., Dunbar, L., Lopez, D., Zarny, M., and C. Jacquenet, "I2NSF Problem Statement and Use cases", draft-ietf-i2nsf-problem-and-use-cases-01 (work in progress), July 2016.
- [I-D.ietf-i2nsf-framework]  
Lopez, E., Lopez, D., Dunbar, L., Strassner, J., Zhuang, J., Parrott, J., Krishnan, R., Durbha, S., "Framework for Interface to Network Security Functions", draft-ietf-i2nsf-framework-02 (work in progress), July 2016
- [I-D.ietf-sfc-nsh]  
Quinn, P. and Elzur, U., "Network Service Header", draft-ietf-sfc-nsh-05 (work in progress), May 2016.
- [I-D.you-i2nsf-user-group-based-policy]  
You, J., Zarny, M., Jacquenet, C., Boucadair, M., Yizhou, L., Strassner, J., and S. Majee, "User-group-based Security Policy for Service Layer", draft-you-i2nsf-user-group-based-policy-02 (work in progress), July 2016.
- [I-D.draft-xia-i2nsf-capability-interface-im]  
Xia, L. Strassner, J., Li, K., Zhang, D., Lopez, E., Bouthors, N., Fang, L., "Information Model of Interface to Network Security Functions Capability interface", draft-xia-i2nsf-capability-interface-im-06 (work in progress), July 2016

[I-D.ietf-i2nsf-terminology]

Hares, S., Strassner, J., Lopez, D., Xia, L., "Interface  
to Network Security Functions (I2NSF) Terminology",  
draft-ietf-i2nsf-terminology-01, July 2016

#### Authors' Addresses

Jianjie You  
Huawei  
101 Software Avenue, Yuhuatai District  
Nanjing, 210012  
China  
Email: youjianjie@huawei.com

John Strassner  
Huawei  
2330 Central Expressway  
San Jose, CA  
USA  
Email: john.sc.strassner@huawei.com

Myo Zarny  
Independent  
Email: myo.zarny@gmail.com

Christian Jacquenet  
France Telecom  
Rennes 35000  
France  
Email: christian.jacquenet@orange.com

Sumandra Majee  
F5 Networks  
3545 N 1st St  
San Jose, CA 95134  
Email: S.Majee@f5.com