I2NSF
Internet Draft
Intended status: Standard Track

Liang Xia
John Strassner
Huawei

Dean Bogdanovic
Juniper Networks

Expires: August 2015

February 10, 2015

Data Model of Interface to Network Security Functions Service Interface
draft-xia-i2nsf-service-interface-dm-00.txt

Status of this Memo

Copyright Notice

Abstract

This draft proposes a generic and intent-driven data model for NSF security policy configuration used by I2NSF clients to negotiate their security requirements with various kinds of entities that each provide one or more NSFs. The Role Based Access Control (RBAC) reference model [INCITS359 RBAC] is used to represent which operations can be performed on what set of secured objects.

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. Cloud-based security customers can be enterprises [I-D.zarny-i2nsf-data-center-use-cases], User Equipment (UE) of mobile networks 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], it should have two types of I2NSF interface to consider:

- o Interface between I2NSF user/client with network controller: 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 applications (e.g., openstack, various BSS/OSS, etc) with various network controllers. This interface is decoupled from various kinds of security device and their device-level security functions. Currently, various high-level applications have their own proprietary interfaces to network controller. This will greatly hinder network controllers interoperating (specifically, exchanging and reusing information) in a cloud-based model. In addition, it will make their management interoperability more difficult. A unified and standard interface is needed;

- o North-bound interface (NBI) provided by the network security functions (NSFs) (e.g., FW, AAA, IPS, Anti-DOS, Anti-Virus, etc), no matter whether the NSFs are contained within Virtual Machines (VM) on servers or physical appliances. [I-D.xia-i2nsf-capability-interface-IM] describes the information model used by this type of interface. Any network entities (e.g., I2NSF clients, network controller, etc) can use this interface to configure the required security functions of NSFs. But, the current situation is different NSF vendors have different proprietary interfaces, supported by different information models to configure their security functions.

For the I2NSF service interface, a more generic and intent-driven approach for security policy configuration is required in order to decouple security from the details of the infrastructure configuration. This also makes it easier to administer and manage security policies independent of device, vendor, and technology.

This draft proposes a generic and intent-driven data model for NSF security policy configuration used by I2NSF clients to negotiate their security requirements with network controllers. The Role Based Access Control (RBAC) reference model [INCITS359 RBAC] is used to represent which operations can be performed on what set of secured objects. Section 3 briefly describes the RBAC approach. Section 4 defines the data model for configuration. Section 5 describes the interaction between resources, services and RBAC. Section 6 clarifies the system control by using RBAC. Section 7 gives its grammar. Section 8 includes some using examples to clarify how to use the data 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 - Authentication, Authorization, Accounting

ACL - Access Control List

AD - Active Directory

ANSI - American National Standards Institute

DDoS - Distributed Denial 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

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. RBAC Overview

Security administration can be costly and prone to error, due to the high number of users, resources, and services that must be protected. Trying to assign access control lists (ACLs) for each user on the system individually cannot scale. With RBAC, security is managed at a level that corresponds closely to the organization's structure. Each user is assigned one or more roles, and each role is assigned one or more privileges that define which operations can be performed by that role. Conceptually, security administration with RBAC consists of determining the operations that must be executed by persons in particular jobs, defining which operations for which objects can be abstracted into a set of permissions, assigning roles to permissions, and assigning employees to the proper roles.

RBAC offers a number of benefits. While users, and the jobs that they perform, change frequently, roles do not. RBAC thus lowers maintenance costs and increases efficiency. It can be used to address specific concerns, such as Separation of Duty, and provides the ability to audit which role performed what operation on which object at what time. It simplifies tracking user activity, and enables stronger adherence to regulatory requirements through ensuring the privacy and confidentiality of user activities.

The NIST model for RBAC was adopted as an American National Standard by the American National Standards Institute, International Committee for Information Technology Standards (ANSI/INCITS) on February 11, 2004.

There are four levels of RBAC. Level 1, Core RBAC, defines a minimum collection of RBAC elements, element sets, and relations in order to achieve a Role-Based Access Control system. Figure 1 shows the reference model of core RBAC. In addition, there are other advanced levels of RBAC by extending the core RBAC.

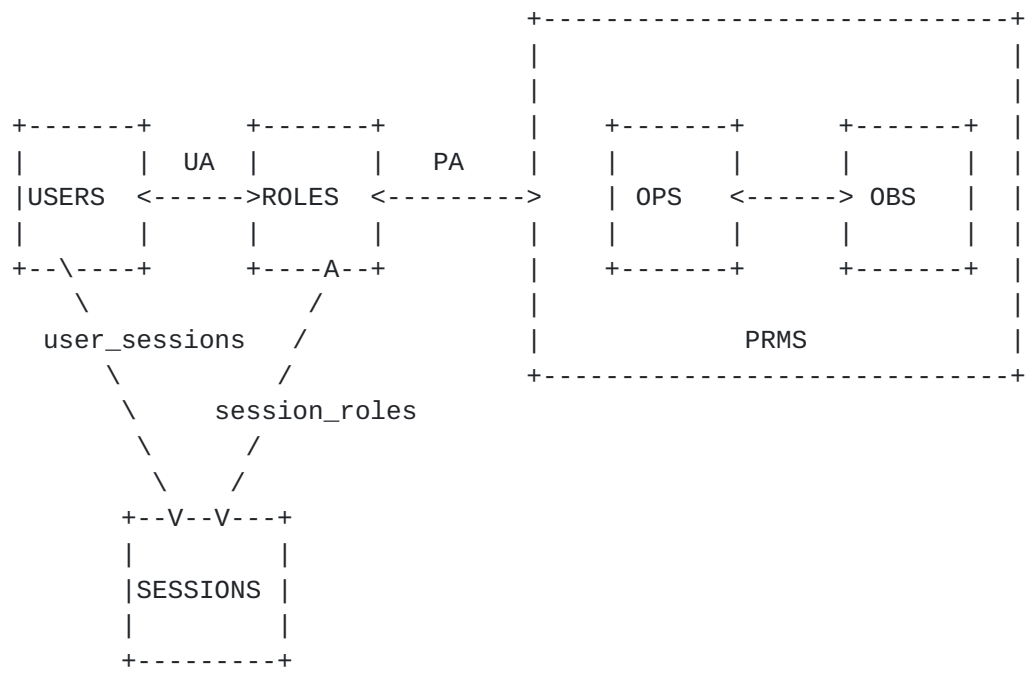

Figure 1. Core RBAC Reference Model

The core RBAC includes the following data elements and mapping relationships:

User - A user is defined as a human being. In this document, the concept of a user can be extended to include machines, networks, or intelligent autonomous agents.

Role - A role is a job function within the context of an organization with some associated semantics regarding the authority and responsibility conferred on the user assigned to the role.

Operations (OPS) - Each Object has a set of operations that are governed by a permission assignment (e.g., read, write, delete, open a file, execute).

Objects (OBS) - an object can be any system resource subject to access control, such as a file, printer, terminal, database record, computer, network, storage, etc.

Permissions (PRMS) - Permission is an approval to perform an operation on one or more RBAC protected objects. Typically, if not specified, then an object has no permissions.

Sessions - each session is a mapping between a user and an activated subset of roles that are assigned to the user.

User Assignment (UA) - a many-to-many relationship between users and roles.

Permission Assignment (PA) - a many-to-many relationship between roles and permissions.

Fundamentally, the RBAC model abstracts entities performing operations on a set of protected objects as roles. This decouples users from the operations that can be performed on a given object. As such, a role is a means for naming many-to-many relationships among individual or groups of users and permissions. In addition, the core RBAC model defines a set of sessions, where each session is a mapping between a user and an activated subset of roles that are assigned to that user.

4. I2NSF Security Policy Data

4.1. Overview

RBAC is a mature and comprehensive reference model for implementing access control. As such, it serves as a good foundation to design a generic security policy data model. The security policy data model should be expressed in an intent-driven style for making it user friendly to decrease its complexity. In addition, the data model should be kept simple by including only the essential logic elements. RBAC provides inherent scalability, which the data model uses to express I2NSF security policies. A high-level data model for I2NSF security policy is shown in Figure 2.

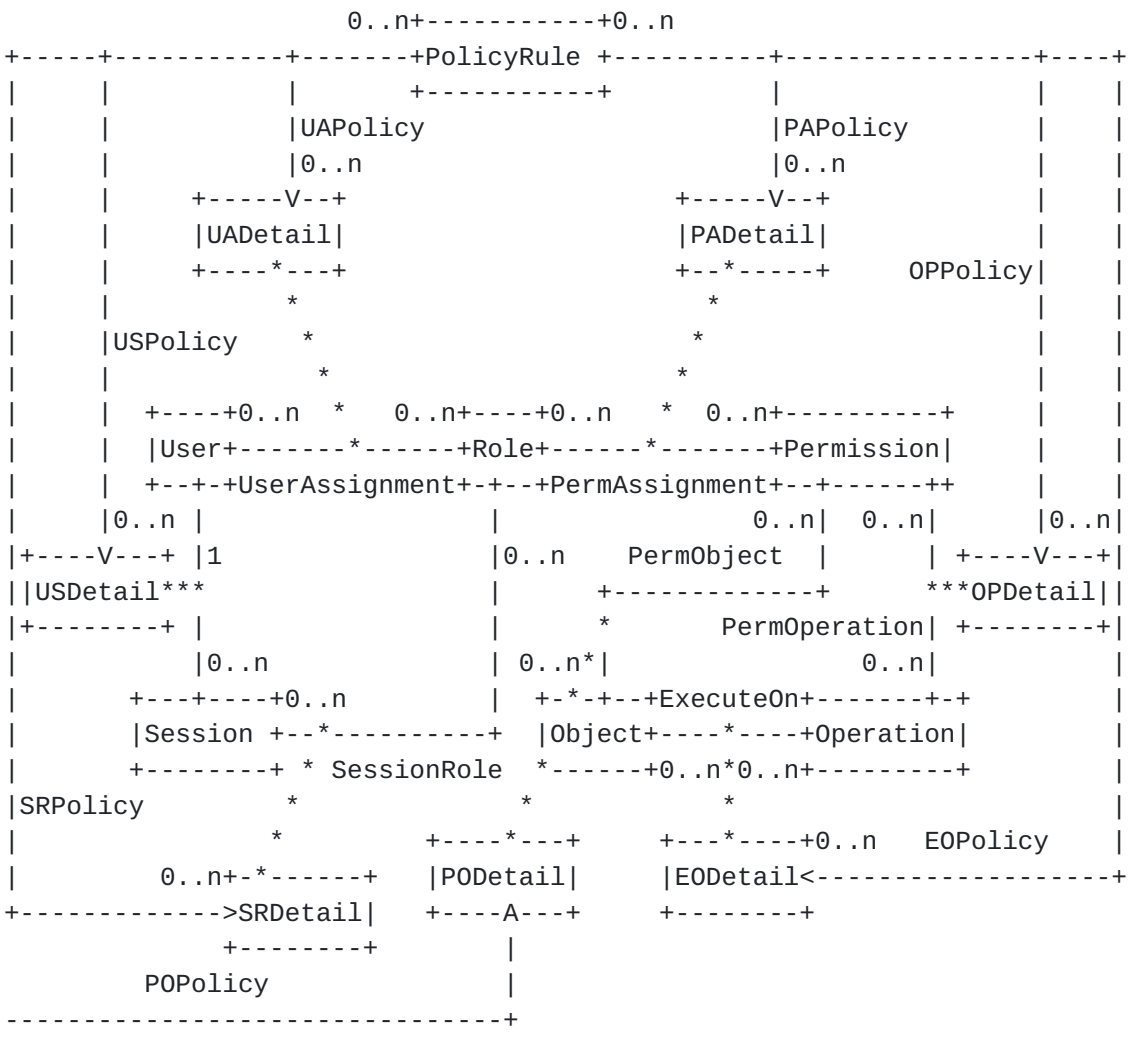

Figure 2. High-Level Data Model for I2NSF Service Interface

4.2. Policy Rule

Policy Rules can be used to control how the fundamental operations of RBAC (i.e., user-assignment and permission-assignment) are performed. This uses the inherent scalability of the RBAC approach, along with the consistency and scalability of policy management, to provide a robust solution.

Note that Policy Rules could also be defined to govern the population of key objects. For example, a policy rule could define whether a particular Role is active for a particular session, or

even whether a particular Role should be enabled, disabled, created, or edited. This will be shown in future revisions of this draft.

For the purposes of this document, a policy rule is an intelligent container. A policy rule can take several different forms (e.g., event-condition-action or goal). A future revision of this draft will specify several options for structuring policy rules. Regardless of what form the policy rule takes, it can operate on an administrative domain. Within that domain, it can represent security requirements, which can take several forms. For example, a security requirement could represent a condition to check for, or actions to take, given an event. An important advantage of this approach is to decouple the structure of a security policy rule from how it is implemented.

Security policies can be created and assigned to any NSFs depending on specific requirements and scenarios. Any NSF can have zero or more security policies. 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.

4.2.1. ECAPolicyRule

One type of policy rule is an Event-Condition-Action, or ECA, policy rule. This type of rule has the following semantics:

```
WHEN an Event-Clause occurs:
IF the Condition-Clause evaluates to TRUE
THEN execute the Action-Clause
```

This type of policy rule contains three mandatory objects, plus optional metadata.

4.2.1.1. Events

An Event is an atomic object that is aggregated by an ECAPolicyRule, and is used to trigger the evaluation of its condition clause. This enables an external application, such as a Policy Server, to dynamically adjust the set of events that are being used to trigger the evaluation of an ECAPolicyRule. For the purposes of this document, one or more Events may be aggregated into an Event Clause using standard Boolean operators (i.e., AND, OR, and NOT).

4.2.1.2. Conditions

A PolicyCondition is an atomic object that is aggregated by an ECAPolicyRule, and is used to define the necessary state and/or

prerequisites to test whether the PolicyActions aggregated by that same ECAPolicyRule should be executed. If the PolicyCondition is TRUE, then the PolicyActions aggregated by the ECAPolicyRule should be executed. For the purposes of this document, one or more Events may be aggregated into an Event Clause using standard Boolean operators (i.e., AND, OR, and NOT).

4.2.1.3. Actions

A PolicyAction is an atomic object that is aggregated by an ECAPolicyRule. It represents the necessary operations that should be performed if the PolicyCondition clause of this ECAPolicyRule evaluates to TRUE. These actions are applied to a set of managed objects, and have the effect of either maintaining an existing state, or transitioning to a new state, of those managed objects. For the purposes of this document, one or more Events may be aggregated into an Event Clause using standard Boolean operators (i.e., AND, OR, and NOT).

4.3. Using Policy Rules With RBAC

Figure 3 shows a generic software design pattern, called the "Policy Pattern", which defines how a set of PolicyRules (of any type and structure) can be used to realize a dependency (called an association) between managed entities. Note that the dependency can be restricted (e.g., to a whole-part dependency, called an aggregation, or to a composition).

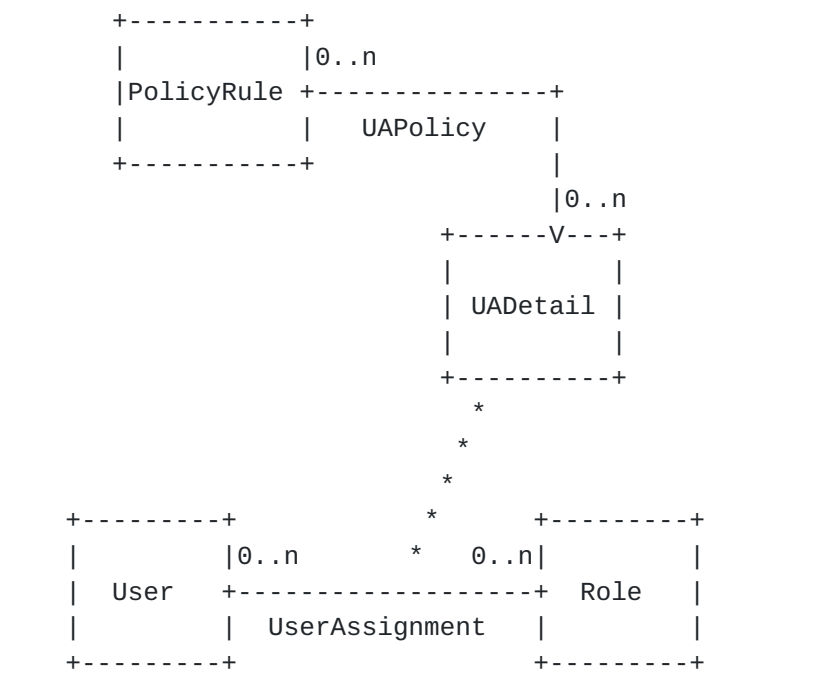

Figure 3. Generic Policy Pattern

In this approach, any dependency between two managed objects can be represented as an association class. (Conceptually, an association class means that the relationship between the two managed entities is not simply a set of pointers, but is a managed entity in its own right, and can be modeled as a class.) A set of PolicyRules can then be related to the association class; this enables the PolicyRules to (for example) control the values of attributes of the association class, which changes the nature of the relationship between the two managed entities.

For example, in Figure 3, the UserAssignment association is realized as an association class called UADetail (denoted by the dotted line). The UADetail class is related to PolicyRule via the UAPolicy aggregation. Hence, in the above example, the PolicyRule can configure attributes of the UADetail association class, which in turn defines how the UserAssignment is implemented for this set of User and Role.

The advantage of a Software Pattern is both its applicability as well as its simplicity. Note that all seven RBAC relationships shown in Figure 2 may be realized using this Pattern.

Policy Rules are used to control behavior. For example, they can be used to enable (or disable) one or more Roles, Users, and Permissions in a given context (e.g., a Session).

A Policy Rule (of any type or structure) can be applied multiple times on different places (e.g., links, devices, networks, vpns). The use of Policy Rules to guarantee that consistent intent is defined and enforced in the whole network, reduces the chances for manual error, and decreases the overall configuration workload.

5. Interaction between Resources, Services, and RBAC

To be finished. This section will describe how Resources (e.g., a port on a router), Services (e.g., a VPN), Users, Roles, and groups of all of these objects, interact.

6. System Control Using RBAC

To be finished. This section will describe how Policy Rules are used to manage and control Resources, Services, Users, Roles, and groups of all of these objects.

7. I2NSF Security Policy Yang Structure

To be finished or moved into a separate draft.

8. Use Examples of I2NSF Security Policy

To be finished.

9. Security Considerations

To be finished.

10. IANA Considerations

11. References

11.1. 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.

11.2. 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.

12. Acknowledgments

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

Authors' Addresses

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

John Strassner
Huawei Technologies
2330 Central Expressway

Santa Clara, CA 95138
USA
email: john.sc.strassner@huawei.com

Dean Bogdanovic
Juniper Networks
Email: deanb@juniper.net