Information Model of NSF Capabilities Update Plan
draft-ietf-i2nsf-capability-00

L. Xia, J. Strassner, C. Basile, D. Lopez

IETF-101, London
March 22, 2018
Agenda

• Introduction
• The New Information Model Structure with Decorator Pattern
• Further Planning Updates
Introduction

• This draft is for:
  – Defining the concept of NSF Capability and its semantics (through an information model)
  – Defining NSF Policy Rule and its semantics (through an information model)
  – The goal:
    • Manage NSFs simply and effectively using Capabilities and Policy Rules
    • Provide an extensible definition of Capabilities and Policy Rules
    • Realize multi-vendor interoperability
ECA Model We Are Using

• The Current Model Uses ECA Policy Rules
  ▶ **Events**: significant occurrences the NSF is able to react to
  ▶ **Conditions**: how the NSF decides which actions to apply
  ▶ **Actions**: what operations to execute
  ▶ **PolicyRule**: a container that aggregates an Event Boolean clause, a Condition Boolean clause, a set of Actions, and metadata

• Behavior
  ▶ Actions MAY execute if Event and Condition (Boolean) clauses BOTH evaluate to TRUE
  ▶ Controlled by **resolution strategy** and **metadata**
    □ Capability Algebra used to make **resolution strategy** decidable
  ▶ **Default actions** MAY be specified
Information Model Structure with Decorator Pattern Design

Decorator Pattern is here
Switching to the Decorator Pattern

• Features are created using subclasses
  - Pros: intuitive, simple, easy to design
  - Cons: not very elegant, requires non-trivial maintenance at every minor update, does not work at run-time since new classes need to be recompiled and redeployed

• The Decorator Pattern
  - Defined in 1995 (!), used in java and windowing toolkits
  - Much more expressive
  - Reduces number of objects at runtime
  - Provides dynamic behavior (composition) instead of fragile, inheritance-based behavior (which is static)
Other Planned Updates

• Improvements / extensions to consider for the next revision of this draft
  • Event clause / Condition clause representation
    – e.g., CNF vs. DNF for Boolean clauses
  • Event clause / Condition clause evaluation function
    – More complex expressions than simple Boolean expressions to be used
  • Action clause evaluation strategies
    – e.g., execute first action only, execute last action only, execute all actions, execute all actions until an action fails
  • More on metadata
    – Authorship, time periods, (+ priorities)
    – More elaborate behavior description and specification
Revisit the IM and DM Drafts Relation and Design

**Information model:**
- draft-xibassnez-i2nsf-capability-02
- draft-xia-i2nsf-security-policy-object-01
- draft-kumar-i2nsf-client-facing-interface-im-03
- draft-hyun-i2nsf-registration-interface-im-02
- draft-zhang-i2nsf-info-model-monitoring-04

Per RFC 3444:
IMs are primarily useful for designers to describe the managed environment, for operators to understand the modeled objects, and for implementors as a guide to the functionality that must be described and coded in the DMs.

**Data Model:**
- draft-hares-i2nsf-capability-data-model-03
- draft-hong-i2nsf-monitoring-data-model-00
- draft-kim-i2nsf-nsf-facing-interface-data-model-02
- draft-jeong-i2nsf-consumer-facing-interface-dm-02
- draft-hyun-i2nsf-registration-interface-dm-01
Next Step

• An update I2NSF-Capability draft will be submitted in 2~3 weeks after this meeting, which basically includes the aforementioned contents.

• Provide our help to make the alignment between capability model and other IM/DM drafts.
Thanks!

Liang Xia (Frank)