Network Working Group
Internet-Draft
Intended status: Informational
Expires: October 29, 2016

J. Halpern
Ericsson
J. Strassner
Huawei Technologies

April 15, 2016

Generic Policy Data Model for Simplified Use of Policy Abstractions (SUPA) draft-halpern-supa-generic-policy-data-model-01

Abstract

This document defines two YANG policy data models. The first is a generic policy model that is meant to be extended on an application-specific basis. The second is an exemplary extension of the first generic policy model, and defines rules as event-condition-action policies. Both models are independent of the level of abstraction of the content and meaning of a policy.

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 October 29, 2016.

Copyright Notice

Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Overview 2
2. Conventions Used in This Document 2
3. Terminology 3
3.1. Acronyms 3
3.2. Definitions 3
3.3. Symbology 4
4. Design of the SUPA Policy Data Models 4
5. SUPA Policy Data Model YANG Module 5
6. IANA Considerations 47
7. Security Considerations 47
8. Acknowledgments 47
9. References 47
9.1. Normative References 48
9.2. Informative References 48
Authors' Addresses 48

1. Overview

This document defines two YANG [RFC6020] [RFC6991] policy data models. The first is a generic policy model that is meant to be extended on an application-specific basis. It is derived from the Generic Policy Information Model (GPIM) defined in [1]. The second is an exemplary extension of the first (generic policy) model, and defines policy rules as event-condition-action tuples. Both models are independent of the level of abstraction of the content and meaning of a policy.

The GPIM defines a common framework as a set of model elements (e.g., classes, attributes, and relationships) that specify a common set of policy management concepts that are independent of the type of policy (e.g., imperative, procedural, declarative, or otherwise). The first YANG data model is a translation of the GPIM to a YANG module. The Eca Policy Rule Information Model (EPRIM), also defined in [1], extends the GPIM to represent policy rules that use the Event-Condition-Action (ECA) paradigm. The second YANG data model maps the EPRIM to YANG. The second YANG data model MAY be used to augment the functionality of the first YANG 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 [RFC2119]. In this document, these words will appear with that interpretation only when in ALL CAPS. Lower case uses of these words are not to

be interpreted as carrying [RFC2119] significance.

3. Terminology

This section defines acronyms, terms, and symbology used in the rest of this document.

3.1. Acronyms

CNF	Conjunctive Normal Form
DNF	Disjunctive Normal Form
ECA	Event-Condition-Action
EPRIM	(SUPA) ECA Policy Rule Information Model
GPIM	(SUPA) Generic Policy Information Model
NETCONF	Network Configuration protocol
OAM&P	Operations, Administration, Management, and Provisioning
OCL	Object Constraint Language
OID	Object IDentifier
SUPA	Simplified Use of Policy Abstractions
UML	Unified Modeling Language
URI	Uniform Resource Identifier

3.2. Definitions

Action: a set of purposeful activities that have a set of associated behavior.

Boolean Clause: a logical statement that evaluates to either TRUE or FALSE. Also called Boolean Expression.

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 Rul can be executed or not.

Constraint: A constraint is a limitation or restriction. Constraints may be added to any type of object (e.g., events, conditions, and actions in Policy Rules).

Constraint Programming: a type of programming that uses constraints to define relations between variables in order to find a feasible (and not necessarily optimal) solution.

Data Model: a data model is a representation of concepts of interest to an environment in a form that is dependent on data repository, data definition language, query language, implementation language, and protocol (typically one or more of these).

ECA: Event - Condition - Action policy.

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.

Information Model: an information model is a representation of concepts of interest to an environment in a form that is independent of data repository, data definition language, query language, implementation language, and protocol.

Metadata: is data that provides descriptive and/or prescriptive information about the object(s) to which it is attached.

Policy Rule: A Policy Rule is a set of rules that are used to manage and control the changing or maintaining of the state of one or more managed objects.

3.3. Symbology

The following representation is used to describe YANG data modules defined in this draft.

- Brackets "[" and "]" enclose list keys.
- Abbreviations before data node names: "rw" means configuration data (read-write), and "ro" means state data (read-only).
- Symbols after data node names: "?" means an optional node, "!" means a presence container, and "*" denotes a list and leaf-list.
- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").
- Ellipsis ("...") stands for contents of subtrees that are not shown.

4. Design of the SUPA Policy Data Models

This will be completed in the next version of this draft. Three important points are:

- different policy models have common semantics
- capture those semantics within a common framework (GPIM)
- extend these semantics with a specific ECA example (EPRIM)

5. SUPA Policy Data Model YANG Module

The SUPA YANG data model module is divided into two main parts:

1) a set of containers that represent the objects that make updated a Policy Rule and its Policy Rule Components
2) a set of containers that represent the objects that define and apply metadata to Policy Rules and/or Policy Rule Components

< This will be finished in version 02 >

<CODE BEGINS> file "ietf-supa-policydatamodel@2016-03-21.yang"

```
module ietf-supa-policydatamodel {

    yang-version 1.1;
    namespace "urn:ietf:params:xml:ns:yang:ietf-supa-policydatamodel";
    prefix supa-pdm;

    import ietf-yang-types {
        prefix yang;
    }

    organization "IETF";
        contact
            "Editor: Joel Halpern
             email: jmh@joelhalpern.com;
             Editor: John Strassner
             email: strazpdj@gmail.com;";

    description
        "This module defines a data model for generic high level
         definition of policies to be applied to a network.
         This module is derived from and aligns with
         draft-strassner-supa-generic-policy-info-model-04.
         Details on all classes, associations, and attributes
         can be found there.
         Copyright (c) 2015 IETF Trust and the persons identified
         as the document authors.  All rights reserved.
         Redistribution and use in source and binary forms, with or
         without modification, is permitted pursuant to, and
         subject to the license terms contained in, the Simplified
         BSD License set forth in Section 4.c of the IETF Trust's
         Legal Provisions Relating to IETF Documents
         (http://trustee.ietf.org/license-info).";
```

```
    revision 2016-04-15 {
        description
            "Fixed pyang 1.1 compilation errors. Fixed must clause
             derefencing used in grouping statements. Reformatted
             and expanded descriptions. Fixed various typos.";
        reference
            "draft-halpern-supa-policy-data-model-01";
    }
    revision 2016-03-21 {
        description
            "Version 1 - initial version";
        reference
            "draft-halpern-supa-policy-data-model-00";
    }

    typedef policy-constraint-language-list {
        type enumeration {
            enum "undefined" {
                description
                    "This may be used as an initialization and/or
                     an error state.";
            }
            enum "OCL2.4" {
                description
                    "Object Constraint Language v2.4. This is a
                     declarative language for describing rules for
                     defining constraints and query expressions.";
            }
            enum "OCL2.x" {
                description
                    "Object Constraint Language, v2.0 through 2.3.1.";
            }
            enum "OCL1.x" {
                description
                    "Object Constraint Language, any version prior
                     to v2.0.";
            }
            enum "QVT1.2R" {
                description
                    "QVT Relational Language.";
            }
            enum "QVT1.2O" {
                description
                    "QVT Operational language.";
            }
            enum "Alloy" {
                description
                    "A language for defining structures and
                     and relations using constraints.";
```

```
            }
        }
```

```
        description
            "The language used to encode the constraints
             relevant to the relationship between the metadata
             and the underlying policy object.";
    }

    typedef policy-data-type-id-encoding-list {
        type enumeration {
            enum "undefined" {
                description
                    "This can be used for either initialization
                     or for signifying an error.";
            }
            enum "String" {
                description
                    "The clause is directly present in
                     the content.";
            }
            enum "GUID" {
                description
                    "The clause is referenced by this GUID.";
            }
            enum "UUID" {
                description
                    "The clause is referenced by this UUID.";
            }
            enum "URI" {
                description
                    "The clause is referenced by this URI.";
            }
            enum "FQDN" {
                description
                    "The clause is referenced by this FQDN.";
            }
        }
        description
            "The list of possible data types used to represent object
             IDs in the SUPA policy hierarchy.";
    }

    typedef policy-data-type-encoding-list {
        type enumeration {
            enum "undefined" {
                description
                    "This can be used for either initialization
                     or for signifying an error.";
            }
            enum "string" {
                description
```

```
                "This represents a string data type.";
        }
```

```
            enum "integer" {
                description
                    "This represents an integer data type.";
            }
            enum "boolean" {
                description
                    "This represents a Boolean data type.";
            }
            enum "floating point" {
                description
                    "This represents a floating point data type.";
            }
            enum "date-and-time" {
                description
                    "This represents a data type that can specify
                     date and/or time.";
            }
            enum "GUID" {
                description
                    "This represents a GUID data type.";
            }
            enum "UUID" {
                description
                    "This represents a UUID data type.";
            }
            enum "URI" {
                description
                    "This represents a Uniform Resource Identifier
                     (URI) data type.";
            }
            enum "DN" {
                description
                    "This represents a Distinguished Name (DN)
                     data type.";
            }
            enum "NULL" {
                description
                    "This represents a NULL data type. NULL means the
                     absence of an actual value. NULL is frequently
                     used to represent a missing or invalid value.";
            }
        }
        description
            "The set of allowable data types used to encode
             multi-valued SUPA Policy attributes.";
    }

// identities are used in this model as a means to provide simple
// reflection to allow an instance-identifier to be tested as to what
```

```
// class it represents. In turn, this allows must clauses to specify
// that the target of a particular instance-identifier leaf must be a
// specific class, or within a certain branch of the inheritance tree.
```

```
// This depends upon the ability to refine the entity class default
// value. The entity class should be read-only. Howeverm as this is
// the target of a MUST condition, it cannot be config-false. Also,
// it appears that we cannot put a MUST condition on its definition,
// as the default (actual) value changes at each inheritance.

    identity POLICY-OBJECT-TYPE {
        description
            "The identity corresponding to a SUPAPolicyObject
             object instance.";
    }

    grouping supa-policy-object-type {
        leaf supa-policy-ID {
            type string;
            mandatory true;
            description
                "The string identifier of this policy object.
                 It must be unique within the policy system.";
        }
        leaf entity-class {
            type identityref {
                base POLICY-OBJECT-TYPE;
            }
            default POLICY-OBJECT-TYPE;
            description
                "The identifier of the class of this grouping.";
        }
        leaf supa-policy-object-ID-encoding {
            type policy-data-type-id-encoding-list;
            mandatory true;
            description
                "The encoding used by the supa-object-ID.";
        }
        leaf supa-policy-object-description {
            type string;
            description
                "Human readable description of the characteristics
                 and behavior of this policy object.";
        }
        leaf supa-policy-name {
            type string;
            description
                "A human-readable name for this policy.";
        }
        leaf-list supa-has-policy-metadata-agg {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  SUPA-HAS-POLICY-METADATA-ASSOC)";
```

```
            description
                "The SUPAPolicyObject object instance that aggregates
                 this set of SUPAPolicyMetadata object instances. As
                 there are attributes on this association, the
                 instance-identifier MUST point to an instance using
                 the grouping supa-has-policy-metadata-detail (which
                 includes subclasses of this association class).";
        }
        description
            "This is the superclass for all SUPA objects. It is
             used to define common attributes and relationships
             that all SUPA subclasses inherit.";
    }

    identity POLICY-COMPONENT-TYPE {
        base POLICY-OBJECT-TYPE;
        description
            "The identity corresponding to a
             SUPAPolicyComponentStructure object instance.";
    }

    grouping supa-policy-component-structure-type {
        uses supa-policy-object-type {
            refine entity-class {
                default POLICY-COMPONENT-TYPE;
            }
        }
        leaf supa-has-policy-component-decorator-part {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  SUPA-HAS-POLICY-COMPONENT-DECORATOR-ASSOC)";
            mandatory true;
            description
                "A reference to the association class for relating
                 policy component decorators to the policy components
                 they decorate. This is the set of
                 SUPAPolicyComponentStructure object instances that are
                 aggregated by a SUPAPolicyComponentDecorator object
                 instance. As there are attributes on this association,
                 the instance-identifier MUST point to an instance
                 using the specified grouping. This defines the object
                 class that this instance-identifier points to.";
       }
       description
          "A superclass for all objects that represent different types
           of components of a Policy Rule. Important subclasses include
           the SUPAPolicyClause and the SUPAPolicyComponentDecorator.
           This object is the root of the decorator pattern; as such,
           it enables all subclasses to be decorated.";
```

```
    }
```

```
    identity POLICY-COMPONENT-DECORATOR-TYPE {
        base POLICY-COMPONENT-TYPE;
        description
            "The identity corresponding to a
             SUPAPolicyComponentDecorator object instance.";
    }

    grouping supa-policy-component-decorator-type {
        uses supa-policy-component-structure-type {
            refine entity-class {
                default POLICY-COMPONENT-DECORATOR-TYPE;
            }
        }
        leaf-list supa-has-policy-component-decorator-agg {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  SUPA-HAS-POLICY-COMPONENT-DECORATOR-ASSOC)";
            max-elements 1;
            description
                "The SUPAPolicyComponentDecorator object instance
                 that aggregates this set of
                 SUPAPolicyComponentStructure object instances. This
                 is a list of associations to the SUPA policy components
                 that this decorator decorates. As there are attributes
                 on this association, the instance-identifier MUST
                 point to an instance using the specified grouping.
                 This defines the object class that this
                 instance-identifier points to.";
        }
        leaf-list supa-decorator-constraints {
            type string;
            description
               "A constraint expression applying to this
                decorator, allowing specification of details not
                captured in its subclasses, using an appropriate
                constraint language.";
        }
        leaf supa-has-decorator-constraint-encoding {
            type policy-constraint-language-list;
            description
               "The language in which the constraints on the
                policy component decorator is expressed.";
        }
        description
           "This object implements the decorator pattern, which
            enables all or part of one or more objects to wrap
            another concrete object.";
    }
```

```
    identity POLICY-COMPONENT-CLAUSE-TYPE {
        base POLICY-COMPONENT-TYPE;
        description
            "The identity corresponding to a SUPAPolicyClause
             object instance.";
    }

    grouping supa-policy-clause-type {
        uses supa-policy-component-structure-type {
            refine entity-class {
               default POLICY-COMPONENT-CLAUSE-TYPE;
            }
        }
        leaf supa-policy-clause-exec-status {
            type enumeration {
                enum "Unknown" {
                    description
                       "This may be used as an initialization and/or
                        an error state.";
                }
                enum "Completed" {
                    description
                       "This signifies that this particular policy
                        clause has run successfully, and is now idle.";
                }
                enum "Working" {
                    description
                       "This signifies that this particular policy
                        clause is currently in use, and no errors have
                        been reported.";
                }
                enum "Not Working" {
                    description
                       "This signifies that this particular policy
                        clause is currently in use, but one or more
                        errors have been reported.";
                }
                enum "Available" {
                    description
                       "This signifies that this particular policy
                        clause could be used, but currently is not
                        in use.";
                }
                enum "In Test" {
                    description
                       "This signifies that this particular policy
                        clause is not for use in operational policies.";
                }
```

```
                enum "Disabled" {
                    description
                       "This signifies that this particular policy
                        clause is not available for use.";
                }
            }
            description "This describes whether this policy clause is in
                use and if so whether it is working properly.";
        }
        leaf-list supa-has-policy-clause-part {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  SUPA-HAS-POLICY-CLAUSE-ASSOC)";
            min-elements 1;
            description
                "The set of SUPAPolicyClause object instances that are
                 aggregated by this SUPAPolicyStructure (i.e., this
                 SUPA Policy Rule) object instance. This defines the
                 object class that this instance-identifier points to.";
        }
        description "The parent class for all SUPA Policy Clauses.";
    }

    identity POLICY-ENCODED-CLAUSE-TYPE {
        base POLICY-COMPONENT-CLAUSE-TYPE;
        description
            "The identity corresponding to a SUPAPolicyEncodedClause
             object instance.";
    }

    grouping supa-encoded-clause-type {
        uses supa-policy-clause-type {
            refine entity-class {
                default POLICY-ENCODED-CLAUSE-TYPE;
            }
        }
        leaf supa-encoded-clause-content {
            type string;
            mandatory true;
            description
               "Either a reference to a source for this clause or the
                string representation of the clause.";
        }
        leaf supa-encoded-clause-encoding {
            type policy-data-type-id-encoding-list;
            mandatory true;
            description
               "The encoding for the encoding clause content.";
```

```
        }
```

```
        leaf supa-encoded-clause-language {
            type enumeration {
                enum "undefined" {
                    description
                        "This may be used as an initialization and/or
                         an error state.";
                }
                enum "CLI" {
                    description
                        "This defines the language as a type of Command
                         Line Interface.";
                }
                enum "TL1" {
                    description
                        "This defines the language as a type of
                         Transaction Language 1.";
                }
                enum "YANG" {
                    description
                        "This defines the language as a type of YANG.";
                }
            }
            mandatory true;
            description
               "Indicates the lanaguage used for this object instance.";
        }
        leaf supa-encoded-clause-response {
            type boolean;
            description
               "If present, this represents the success or failure
                of the last invocation of this clause.";
        }
        description
           "This class refines the behavior of the supa-policy-clause
            by encoding the contents of the clause into the attributes
            of this object. This enables clauses that are not based on
            other SUPA objects to be modeled.";
    }

    container supa-encoding-clause-container {
        description
            "This is a container to collect all object instances of
             type SUPAEncodedClause.";
        list supa-encoding-clause-list {
            key supa-policy-ID;
            uses supa-encoded-clause-type;
            description
                "List of all instances of supa-encoding-clause-type.
                 If a module defines subclasses of the encoding clause,
```

```
                those will be stored in a separate container.";
        }
    }
```

```
    identity POLICY-COMPONENT-TERM-TYPE {
        base POLICY-COMPONENT-DECORATOR-TYPE;
        description
            "The identity corresponding to a
             SUPAPolicyComponentDecorator object instance.";
    }

    grouping supa-policy-term-type {
        uses supa-policy-component-decorator-type {
            refine entity-class {
               default POLICY-COMPONENT-TERM-TYPE;
            }
        }
        leaf supa-policy-term-is-negated {
            type boolean;
            description
               "If the value of this attribute is true, then
                this particular term is negated.";
        }
        description
           "This is the superclass of all SUPA policy objects that are
            used to test or set the value of a variable.";
    }

    identity POLICY-COMPONENT-VARIABLE-TYPE {
        base POLICY-COMPONENT-TERM-TYPE;
        description
            "The identity corresponding to a SUPAPolicyVariable
             object instance.";
    }

    grouping supa-policy-variable-type {
        uses supa-policy-term-type {
            refine entity-class {
               default POLICY-COMPONENT-VARIABLE-TYPE;
            }
        }
        leaf supa-policy-variable-name {
             type string;
             description
                "A human-readable name for this policy variable.";
         }
        description
           "This is one formulation of a SUPA Policy Clause. It uses
            an object, defined in the SUPA hierarchy, to represent the
            variable portion of a SUPA Policy Clause. The attribute
            defined by the supa-policy-variable-name specifies an
            attribute whose content should be compared to a value,
```

```
            which is typically specified by supa-policy-value-type.";
    }
```

```
    container supa-policy-variable-container {
        description
            "This is a container to collect all object instances of
             type SUPAPolicyVariable.";
        list supa-policy-variable-list {
            key supa-policy-ID;
            uses supa-policy-variable-type;
            description
                "List of all instances of supa-policy-variable-type.
                 If a module defines subclasses of this class,
                 those will be stored in a separate container.";
        }
    }

    identity POLICY-COMPONENT-OPERATOR-TYPE {
        base POLICY-COMPONENT-TERM-TYPE;
        description
            "The identity corresponding to a SUPAPolicyOperator
             object instance.";
    }

    grouping supa-policy-operator-type {
        uses supa-policy-term-type {
            refine entity-class {
               default POLICY-COMPONENT-OPERATOR-TYPE;
            }
        }
        leaf supa-policy-value-op-type {
            type enumeration {
                enum "unknown" {
                    description
                        "This may be used as an initialization and/or
                         an error state.";
                }
                enum "greater than" {
                    description
                        "A greater-than operator.";
                }
                enum "greater than or equal to" {
                    description
                        "A greater-than-or-equal-to operator.";
                }
                enum "less than" {
                    description
                        "A less-than operator.";
                }
                enum "less than or equal to" {
                    description
```

```
                    "A less-than-or-equal-to operator.";
                }
```

```
                enum "equal to" {
                    description
                        "An equal-to operator.";
                }
                enum "not equal to"{
                    description
                        "A not-equal-to operator.";
                }
                enum "IN" {
                    description
                        "An operator that determines whether a given
                         value matches any of the specified values.";
                }
                enum "NOT IN" {
                    description
                        "An operator that determines whether a given
                         value does not match any of the specified
                         values.";
                }
                enum "SET" {
                    description
                        "An operator that makes the value of the
                         result equal to the input value.";
                }
                enum "CLEAR"{
                    description
                        "An operator that deletes the value of the
                         specified object.";
                }
                enum "BETWEEN" {
                    description
                        "An operator that determines whether a given
                         value is within a specified range of values.";
                }
            }
            mandatory true;
            description
                "The type of operator used to compare the variable
                 and value portions of this SUPA Policy Clause.";
        }
        description
           "This is one formulation of a SUPA Policy Clause. It uses
            an object, defined in the SUPA hierarchy, to represent the
            operator portion of a SUPA Policy Clause. The attribute
            defined by the supa-policy-op-type specifies an attribute
            whose content defines the type of operator used to compare
            the variable and value portions of this policy clause.";
    }
```

```
    container supa-policy-operator-container {
        description
            "This is a container to collect all object instances of
             type SUPAPolicyOperator.";
        list supa-policy-operator-list {
            key supa-policy-ID;
            uses supa-policy-operator-type;
            description
                "List of all instances of supa-policy-operator-type.
                 If a module defines subclasses of this class,
                 those will be stored in a separate container.";
        }
    }

    identity POLICY-COMPONENT-VALUE-TYPE {
        base POLICY-COMPONENT-TERM-TYPE;
        description
            "The identity corresponding to a SUPAPolicyValue
             object instance.";
    }

    grouping supa-policy-value-type {
        uses supa-policy-term-type {
            refine entity-class {
               default POLICY-COMPONENT-VALUE-TYPE;
            }
        }
        leaf-list supa-policy-value-content {
            type string;
            description
               "The content of the value portion of this SUPA Policy
                Clause. The data type of the content is specified in
                the supa-policy-value-encoding.";
        }
        leaf supa-policy-value-encoding {
            type policy-data-type-encoding-list;
            description
                "The data type of the supa-policy-value-content.";
        }
        description
           "This is one formulation of a SUPA Policy Clause. It uses
            an object, defined in the SUPA hierarchy, to represent the
            value portion of a SUPA Policy Clause. The attribute
            defined by the supa-policy-value-content specifies an
            attribute whose content should be compared to a variable,
            which is typically specified by supa-policy-variable-type.";
    }
```

```
    container supa-policy-value-container {
        description
            "This is a container to collect all object instances of
             type SUPAPolicyValue.";
        list supa-policy-value-list {
            key supa-policy-ID;
            uses supa-policy-value-type;
            description
                "List of all instances of supa-policy-value-type.
                 If a module defines subclasses of this class,
                 those will be stored in a separate container.";
        }
    }

    identity POLICY-GENERIC-DECORATED-TYPE {
        base POLICY-COMPONENT-DECORATOR-TYPE;
        description
            "The identity corresponding to a
             SUPAGenericDecoratedComponent object instance.";
    }

    grouping supa-policy-generic-decorated-type {
        uses supa-policy-component-decorator-type {
            refine entity-class {
               default POLICY-GENERIC-DECORATED-TYPE;
            }
        }
        leaf-list supa-policy-generic-decorated-content {
            type string;
            description
               "The content of this SUPA Policy Clause. The data type
                of this attribute is specified in the
                supa-policy-generic-decorated-encoding.";
        }
        leaf supa-policy-generic-decorated-encoding {
            type policy-data-type-encoding-list;
            description
                "The data type of the
                 supa-policy-generic-decorated-content attribute.";
        }
        description
           "This object enables a generic object to be defined and
            used as a decorator in a SUPA Policy Clause.
            This should not be confused with the SUPAEncodedClause
            class. This class represents a single, atomic,
            vendor-specific object that defines a portion of a SUPA
            Policy Clause, whereas a SUPA Policy Encoded Clause
            represents the entire policy clause.";
    }
```

```
    container supa-policy-generic-decorated-container {
        description
            "This is a container to collect all object instances of
             type SUPAGenericDecoratedComponent.";
        list supa-encoding-clause-list {
            key supa-policy-ID;
            uses supa-policy-generic-decorated-type;
            description
                "List of all instances of
                 supa-policy-generic-decorated-type. If a module
                 defines subclasses of this class, those will be
                 stored in a separate container.";
        }
    }

    identity POLICY-COLLECTION {
        base POLICY-COMPONENT-DECORATOR-TYPE;
        description
            "The identity corresponding to a SUPAPolicyCollection
             object instance.";
    }

    grouping supa-policy-collection {
        uses supa-policy-component-decorator-type {
            refine entity-class { default POLICY-COLLECTION;
            }
        }
        leaf-list supa-policy-collection-content {
            type string;
            description
               "The content of this collection object. The data type
                is specified in supa-policy-collection-encoding.";
        }
        leaf supa-policy-collection-encoding {
            type enumeration {
                enum "undefined" {
                    description
                        "This may be used as an initialization and/or
                         an error state.";
                }
                enum "by regex" {
                    description
                        "This defines the data type of the content of
                         this collection instance to be a regular
                         expression that contains all or part of a
                         string to match the class name of the object
                         that is to be collected by this instance of a
                         SUPAPolicyCollection class.";
```

```
                }
```

```
                enum "by URI" {
                    description
                        "This defines the data type of the content of
                         this collection instance to be a Uniform
                         Resource Identifier. It identifies the object
                         instance that is to be collected by this
                         instance of a SUPAPolicyCollection class.";
                }
            }
            mandatory true;
            description
                "The data type of the supa-policy-collection-content.";
        }
        leaf supa-policy-collection-function {
            type enumeration {
                enum "undefined" {
                    description
                        "This may be used as an initialization and/or
                         an error state.";
                }
                enum "event collection" {
                    description
                        "This collection contains objects that are used
                         to populate the event clause of a
                         SUPA Policy.";
                }
                enum "condition collection" {
                    description
                        "This collection contains objects that are used
                         to populate the condition clause of a
                         SUPA Policy.";
                }
                enum "action collection" {
                     description
                        "This collection contains objects that are used
                         to populate the action clause of a
                         SUPA Policy.";
                }
               enum "logic collection" {
                    description
                        "This collection contains objects that define
                         logic for processing a SUPA Policy.";
                }
            }
            description
                "Defines how this collection instance is to be used.";
        }
```

```
        leaf supa-policy-collection-is-ordered {
            type boolean;
            description
                "If the value of this leaf is true, then all elements
                 in this collection are ordered.";
        }
        leaf supa-policy-collection-type {
            type enumeration {
                enum "undefined" {
                    description
                       "This may be used as an initialization and/or
                        an error state.";
                }
                enum "set" {
                    description
                       "An unordered collection of elements that MUST
                        NOT have duplicates.";
                }
                enum "bag" {
                    description
                       "An unordered collection of elements that MAY
                        have duplicates.";
                }
                enum "dictionary" {
                    description
                       "A list of values that is interpreted as a set
                        of pairs, with the first entry of each pair
                        interpreted as a dictionary key, and the
                        second entry interpreted as a value for that
                        key. As a result, collections using this value
                        of supa-policy-collection-type MUST have
                        supa-policy-collection-is-ordered set to true.";
                }
            }
            mandatory true;
            description
                "The type of the supa-policy-collection.";
        }
        description
           "This enables a collection of arbitrary objects to be
            defined and used in a SUPA Policy Clause.
            This should not be confused with the SUPAEncodedClause
            class. This class represents a single, atomic, object that
            defines a portion of a SUPA Policy Clause, whereas a SUPA
            Policy Encoded Clause represents the entire policy clause.";
    }
```

```
    container supa-policy-collection-container {
        description
            "This is a container to collect all object instances of
             type SUPAPolicyCollection.";
        list supa-policy-collection-list {
            key supa-policy-ID;
            uses supa-policy-collection;
            description
                "List of all instances of supa-policy-collection.
                 If a module defines subclasses of this class,
                 those will be stored in a separate container.";
        }
    }

    identity POLICY-STRUCTURE-TYPE {
        base POLICY-OBJECT-TYPE;
        description
            "The identity corresponding to a SUPAPolicyStructure
             object instance.";
    }

    grouping supa-policy-structure-type {
        uses supa-policy-object-type {
            refine entity-class {
                default POLICY-STRUCTURE-TYPE;
            }
        }
        leaf supa-policy-admin-status {
            type enumeration {
                enum "unknown" {
                    description
                        "This may be used as an initialization and/or
                         an error state.";
                }
                enum "enabled" {
                    description
                        "This SUPA Policy Rule has been
                         administratively enabled.";
                }
                enum "disabled" {
                    description
                        "This SUPA Policy Rule has been
                         administratively disabled.";
                }
                enum "in test" {
                    description
                        "This SUPA Policy Rule has been
                         administratively placed into test mode, and
                         SHOULD NOT be used as part of an operational
```

```
                        policy rule.";
                }
            }
```

```
            mandatory true;
            description
                "The current admnistrative status of this SUPA POLICY
                 Rule.";
        }
        leaf supa-policy-continuum-level {
            type uint32;
            description
                "This is the current level of abstraction of this
                 particular SUPA Policy Rule.";
        }
        leaf supa-policy-deploy-status {
            type enumeration {
                enum "undefined" {
                    description
                        "This may be used as an initialization and/or
                         an error state.";
                }
                enum "deployed and enabled" {
                    description
                        "This SUPA Policy Rule has been deployed and
                         enabled.";
                }
                enum "disabled" {
                    description
                        "This SUPA Policy Rule has been
                         administratively disabled.";
                }
                enum "in test" {
                    description
                        "This SUPA Policy Rule has been
                         administratively placed into test mode, and
                         SHOULD NOT be used as part of an operational
                         policy rule.";
                }
            }
            mandatory true;
            description
                "This is the current level of abstraction of this
                 particular SUPA Policy Rule.";
        }
        leaf supa-policy-exec-status {
            type enumeration {
                enum "undefined" {
                    description
                        "This may be used as an initialization and/or
                         an error state.";
                }
```

```
                enum "operational success" {
                    description
                        "This SUPA Policy Rule has been executed in
                         operational mode, and produced no errors.";
                }
                enum "operational failure" {
                    description
                        "This SUPA Policy Rule has been executed in
                         operational mode, but has produced at least
                         one error.";
                }
                enum "currently in operation" {
                    description
                        "This SUPA Policy Rule is currently still
                         executing in operational mode.";
                }
                enum "ready" {
                    description
                        "This SUPA Policy Rule is ready to be
                         executed in operational mode.";
                }
                enum "test success" {
                    description
                        "This SUPA Policy Rule has been executed in
                         test mode, and produced no errors.";
                }
                enum "test failure" {
                    description
                        "This SUPA Policy Rule has been executed in
                         test mode, but has produced at least
                         one error.";
                }
                enum "currently in test" {
                    description
                        "This SUPA Policy Rule is currently still
                         executing in test mode.";
                }
            }
            mandatory true;
            description
                "This is the current level of abstraction of this
                 particular SUPA Policy Rule.";
        }
        leaf supa-policy-exec-fail-strategy {
            type enumeration {
                enum "undefined" {
                    description
                        "This may be used as an initialization and/or
```

```
                    an error state.";
                }
```

```
                enum "rollback all" {
                    description
                        "This means that execution of this SUPA
                         Policy Rule is stopped, rollback of all
                         actions (whether successful or not) is
                         attempted, and all SUPA Policy Rules that
                         otherwise would have executed are ignored.";
                }
                enum "rollback failure" {
                    description
                        "This means that execution of this SUPA
                         Policy Rule is stopped, and rollback is
                         attempted for only the SUPA Policy Rule that
                         failed to execute correctly.";
                }
                enum "stop execution" {
                    description
                        "This means that execution of this SUPA Policy
                         Rule SHOULD be stopped.";
                }
                enum "ignore" {
                    description
                        "This means that any failures produced by this
                         SUPA Policy Rule SHOULD be ignored.";
                }
            }
            mandatory true;
            description
                "This defines what actions, if any, should be taken by
                 this particular SUPA Policy Rule if it fails to
                 execute correctly. Some implementations may not be
                 able to accommodate the rollback failure option;
                 hence, this option may be skipped.";
        }
        leaf-list supa-has-policy-source-agg {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  SUPA-HAS-POLICY-SOURCE-ASSOC)";
            description
                "The SUPAPolicyStructure (i.e., the type of SUPA
                 Policy Rule) object instance that aggregates this set
                 set of SUPAPolicySource object instances. This
                 defines the object class that this instance-identifier
                 points to.";
        }
        leaf-list supa-has-policy-target-agg {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
```

```
                SUPA-HAS-POLICY-TARGET-ASSOC)";
```

```
            description
                 "This represents the aggregation of Policy Target
                 objects by this particular SUPA Policy Rule. It is
                 the SUPAPolicyStructure object instance that
                 aggregates this set of SUPAPolicyTarget object
                 instances. This defines the object class that
                 this instance-identifier points to.";
        }
        leaf-list supa-has-policy-clause-agg {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  SUPA-HAS-POLICY-CLAUSE-ASSOC)";
            description
                "The SUPAPolicyStructure object instance that
                 aggregates this set of SUPAPolicyClause object
                 instances. This defines the object class that
                 this instance-identifier points to.";
        }
        leaf-list supa-has-policy-exec-action-assoc-src-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  SUPA-HAS-POLICY-EXEC-ACTION-ASSOC)";
            description
                "This associates a SUPAPolicyStructure (i.e., a SUPA
                 Policy Rule) object instance to zero or more SUPA
                 Policy Actions to be used to correct errors caused if
                 this SUPA Policy Rule does not execute correctly.";
        }
        leaf-list supa-has-policy-exec-action-assoc-dst-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  SUPA-HAS-POLICY-EXEC-ACTION-ASSOC)";
            min-elements 1;
            description
                "The set of zero or more SUPA Policy Actions to be used
                 by this particular SUPAPolicyStructure (i.e., SUPA
                 Policy Rule to correct errors caused if this SUPA
                 Policy Rule does not execute correctly.";
        }
        description
           "A superclass for all objects that represent different types
            of Policy Rules. Currently, this is limited to a single
            type - the event-condition-action (ECA) policy rule.
            A SUPA Policy may be an individual policy, or a set of
            policies. This is supported by applying the composite
            pattern to this class.";
    }
```

```
    identity POLICY-SOURCE-TYPE {
        base POLICY-OBJECT-TYPE;
        description
            "The identity corresponding to a SUPAPolicySource
             object instance.";
    }

    grouping supa-policy-source-type {
        uses supa-policy-object-type {
            refine entity-class {
                default POLICY-SOURCE-TYPE;
            }
        }
        leaf-list supa-has-policy-source-part {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  SUPA-HAS-POLICY-SOURCE-ASSOC)";
            description
                "This represents the aggregation of one or more SUPA
                 Policy Source objects to this particular SUPA Policy
                 Rule object. In other words, it is the set of
                 SUPAPolicySource object instances that are aggregated
                 by this SUPAPolicyStructure (i.e., this SUPA Policy
                 Rule). This defines the object class that this
                 instance-identifier points to.";
        }
        description
           "This object defines a set of managed entities that
            authored, or are otherwise responsible for, this SUPA
            Policy Rule. Note that a SUPA Policy Source does not
            evaluate or execute SUPAPolicies. Its primary use is for
            auditability and the implementation of deontic and/or
            alethic logic.";
    }

    identity POLICY-TARGET-TYPE {
        base POLICY-OBJECT-TYPE;
        description
            "The identity corresponding to a SUPAPolicyTarget
             object instance.";
    }

    grouping supa-policy-target-type {
        uses supa-policy-object-type {
            refine entity-class {
                default POLICY-TARGET-TYPE;
            }
        }
```

```
        leaf-list supa-has-policy-target-part {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  SUPA-HAS-POLICY-TARGET-ASSOC)";
            description
                "This represents the aggregation of one or more SUPA
                 Policy Target objects to this particular SUPA Policy
                 Rule object. In other words, it is the set of
                 SUPAPolicyTarget object instances that are aggregated
                 by this SUPAPolicyStructure (i.e., this SUPA Policy
                 Rule). This defines the object class that this
                 instance-identifier points to.";
        }
        description
           "This object defines a set of managed entities that a
            SUPA Policy Rule is applied to.";
    }

    identity POLICY-METADATA-TYPE {
        description
            "The identity corresponding to a SUPAPolicyMetadata
             object instance.";
    }

    grouping supa-policy-metadata-type {
        leaf supa-policy-metadata-id {
            type string;
            mandatory true;
            description
                "This represents part of the object identifier of an
                 instance of this class. It defines the content of the
                 object identifier.";
        }
        leaf entity-class {
            type identityref {
                base POLICY-METADATA-TYPE;
            }
            default POLICY-METADATA-TYPE;
            description
               "The identifier of the class of this grouping.";
        }
        leaf supa-policy-metadata-id-encoding {
            type policy-data-type-id-encoding-list;
            mandatory true;
            description
                "This represents part of the object identifier of an
                 instance of this class. It defines the format of the
                 object identifier.";
```

```
        }
```

```
        leaf supa-policy-metadata-description {
            type string;
            description
                "This contains a free-form textual description of this
                 metadata object.";
        }
        leaf supa-policy-metadata-name {
            type string;
            description
                "This contains a human-readable name for this
                 metadata object.";
        }
        leaf-list supa-has-policy-metadata-part {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  SUPA-HAS-POLICY-METADATA-ASSOC)";
            description
                "This represents the set of SUPAPolicyMetadata object
                 instances that are aggregated by this SUPAPolicyObject
                 object instance (i.e., this is the set of policy
                 metadata aggregated by this SUPAPolicyObject). As
                 there are attributes on this association, the
                 instance-identifier MUST point to an instance using
                 the grouping supa-has-policy-metadata-detail (which
                 includes the subclasses of the association class).";
        }
        leaf supa-policy-metadata-decorator-part {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  SUPA-HAS-POLICY-COMPONENT-DECORATOR-ASSOC)";
            mandatory true;
            description
               "This object implements the decorator pattern, which is
                applied to SUPA metadata objects. This enables all or
                part of one or more metadata objects to wrap another
                concrete metadata object.";
        }
        description
           "This is the superclass of all metadata classes. Metadata
            is information that describes and/or prescribes the
            characteristics and behavior of another object that is
            not an inherent, distinguishing characteristics or
            behavior of that object.";
    }

    identity POLICY-METADATA-CONCRETE-TYPE {
        base POLICY-METADATA-TYPE;
        description
```

```
            "The identity corresponding to a SUPAPolicyConcreteMetadata
             object instance.";
    }
```

```
    grouping supa-policy-concrete-metadata-type {
        uses supa-policy-metadata-type {
            refine entity-class {
                default POLICY-METADATA-TYPE;
            }
        }
        leaf supa-policy-metadata-valid-period-end {
            type yang:date-and-time;
            description
                "This defines the ending date and time that this
                 metadata object is valid for.";
        }
        leaf supa-policy-metadata-valid-period-start {
            type yang:date-and-time;
            description
                "This defines the starting date and time that this
                 metadata object is valid for.";
        }
        description
           "This is a concrete class that will be wrapped by concrete
            instances of the SUPA Policy Metadata Decorator class. It
            can be viewed as a container for metadata that will be
            attached to a subclass of SUPA Policy Object. It may
            contain all or part of one or more metadata subclasses.";
    }

    container supa-policy-concrete-metadata-container {
        description
            "This is a container to collect all object instances of
             type SUPAPolicyConcreteMetadata.";
        list supa-policy-concrete-metadata-list {
            key supa-policy-metadata-id;
            uses supa-policy-concrete-metadata-type;
            description
                "A list of all supa-policy-metadata instances in the
                 system.";
        }
    }

    identity POLICY-METADATA-DECORATOR-TYPE {
        base POLICY-METADATA-TYPE;
        description
            "The identity corresponding to a
             SUPAPolicyMetadataDecorator object instance.";
    }
```

```
    grouping supa-policy-metadata-decorator-type {
        uses supa-policy-metadata-type {
            refine entity-class {
                default POLICY-METADATA-DECORATOR-TYPE;
            }
        }
        leaf-list supa-policy-metadata-decorator-agg {
            type instance-identifier;
             must "derived-from-or-self (deref(.)/entity-class,
                   SUPA-HAS-POLICY-COMPONENT-DECORATOR-ASSOC)";
            max-elements 1;
            description
                "This represents the decorator pattern being applied to
                 metadata. This is the aggregate part (i.e., the
                 concrete subclass of the SUPAPolicyMetadataDecorator
                 class that wraps a concrete subclass of
                 SUPAPolicyMetadata; currently, the only such class is
                 SUPAPolicyConcreteMetadata).";
        }
        description
           "This object implements the decorator pattern, which is
            applied to SUPA metadata objects. This enables all or part
            of one or more metadata objects to wrap another concrete
            metadata object.";
    }

    identity POLICY-METADATA-DECORATOR-ACCESS-TYPE {
        base POLICY-METADATA-DECORATOR-TYPE;
        description
            "The identity corresponding to a
             SUPAPolicyAccessMetadataDef object instance.";
    }

    grouping supa-policy-metadata-decorator-access-type {
        uses supa-policy-metadata-decorator-type {
            refine entity-class {
                default POLICY-METADATA-DECORATOR-ACCESS-TYPE;
            }
        }
        leaf supa-policy-metadata-access-priv-def {
            type enumeration {
                enum "undefined" {
                    description
                        "This may be used as an initialization and/or
                         an error state.";
                }
                enum "read only" {
                    description
                        "This defines access as read only for ALL SUPA
```

```
                    Policy object instances that are adorned with
                    this metadata object.";
                }
```

```
                enum "read write" {
                    description
                        "This defines access as read and/or write for
                         ALL SUPA Policy object instances that are
                         adorned with this metadata object.";
                }
                enum "specified by MAC" {
                    description
                        "This defines access as defined by an external
                         Mandatory Access Control model. The name and
                         location of this model are specified in the
                         supa-policy-metadata-access-priv-model-name
                         and supa-policy-metadata-access-priv-model-ref
                         attributes of this metadata object.";
                }
                enum "specified by DAC" {
                    description
                        "This defines access as defined by an external
                         Discretionary Access Control model. The name
                         and location of this model are specified in the
                         supa-policy-metadata-access-priv-model-name
                         and supa-policy-metadata-access-priv-model-ref
                         attributes of this metadata object.";
                }
                enum "specified by RBAC" {
                    description
                        "This defines access as defined by an external
                         Role Based Access Control model. The name
                         and location of this model are specified in the
                         supa-policy-metadata-access-priv-model-name
                         and supa-policy-metadata-access-priv-model-ref
                         attributes of this metadata object.";
                }
                enum "specified by ABAC" {
                    description
                        "This defines access as defined by an external
                         Attribute Based Access Control model. The name
                         and location of this model are specified in the
                         supa-policy-metadata-access-priv-model-name
                         and supa-policy-metadata-access-priv-model-ref
                         attributes of this metadata object.";
                }
                enum "specified by custom" {
                    description
                        "This defines access as defined by an external
                         Custom Access Control model. The name and
                         location of this model are specified in the
                         supa-policy-metadata-access-priv-model-name
                         and supa-policy-metadata-access-priv-model-ref
```

```
                        attributes of this metadata object.";
                }
            }
```

```
            description
                "This defines the type of access control model that is
                 used by this object instance.";
        }
        leaf supa-policy-metadata-access-priv-model-name {
            type string;
            description
                "This contains the name of the access control model
                 being used. If the value of the
                 supa-policy-metadata-access-priv-model-ref is 0-2,
                 then the value of this attribute is not applicable.
                 Otherwise, the text in this class attribute should be
                 interpreted according to the value of the
                 supa-policy-metadata-access-priv-model-ref class
                 attribute.";
        }
        leaf supa-policy-metadata-access-priv-model-ref {
            type enumeration {
                enum "undefined" {
                    description
                        "This can be used for either initialization
                        or for signifying an error.";
                }
                enum "URI"  {
                    description
                        "The clause is referenced by this URI.";
                }
                enum "GUID"  {
                    description
                        "The clause is referenced by this GUID.";
                }
                enum "UUID"  {
                    description
                        "The clause is referenced by this UUID.";
                }
                enum "FQDN"  {
                    description
                        "The clause is referenced by this FQDN.";
                }
            }
            description
                "This defines the data type of the
                 supa-policy-metadata-access-priv-model-name
                 attribute.";
        }
        description
            "This is a concrete class that defines metadata for access
             control information that can be added to a SUPA Policy
             object. This is done using the SUPAHasPolicyMetadata
```

```
            aggregation.";
    }
```

```
    container supa-policy-metadata-decorator-access-container {
        description
            "This is a container to collect all object instances of
             type SUPAPolicyAccessMetadataDef.";
        list supa-policy-metadata-decorator-access-list {
            key supa-policy-metadata-id;
            uses supa-policy-metadata-decorator-type;
            description
                "A list of all supa-policy-metadata-decorator-access
                instances in the system.  Instances of subclasses
                will be in a separate list.";
        }
    }

    identity POLICY-METADATA-DECORATOR-VERSION-TYPE {
        base POLICY-METADATA-DECORATOR-TYPE;
        description
            "The identity corresponding to a
             SUPAPolicyVersionMetadataDef object instance.";
    }

    grouping supa-policy-metadata-decorator-version-type {
        uses supa-policy-metadata-decorator-type {
            refine entity-class {
                default POLICY-METADATA-DECORATOR-VERSION-TYPE;
            }
        }
        leaf supa-policy-metadata-version-major {
            type string;
            description
                "This contains a string (typically representing an
                 integer in the overall version format) that indicates
                 a significant increase in functionality is present in
                 this version.";
        }
        leaf supa-policy-metadata-version-minor {
            type string;
            description
                "This contains a string (typically representing an
                 integer in the overall version format) that indicates
                 that this release contains a set of features and/or bug
                 fixes that collectively do not warrant incrementing the
                 supa-policy-metadata-version-major attribute.";
        }
        leaf supa-policy-metadata-version-rel-type {
            type enumeration {
                enum "undefined" {
                    description
```

```
                    "This can be used for either initialization
                    or for signifying an error.";
                }
```

```
                enum "internal" {
                    description
                        "This indicates that this version should only
                         be used for internal (development) purposes.";
                }
                enum "alpha" {
                    description
                        "This indicates that this version is considered
                         to be alpha quality.";
                }
                enum "beta" {
                    description
                        "This indicates that this version is considered
                         to be beta quality.";
                }
                enum "release candidate" {
                    description
                        "This indicates that this version is considered
                         to be a candidate for full production.";
                }
                enum "release production" {
                    description
                        "This indicates that this version is considered
                         to be ready for full production.";
                }
                enum "maintenance" {
                    description
                        "This indicates that this version is considered
                         to be for maintenance purposes.";
                }
            }
            description
                "This defines the type of this version's release.";
        }
        leaf supa-policy-metadata-version-rel-type-num {
            type string;
            description
                "This contains a string (typically representing an
                 integer in the overall version format) that indicates
                 a significant increase in functionality is present in
                 this version.";
        }
        description
            "This is a concrete class that defines metadata for version
             control information that can be added to a SUPA Policy
             object. This is done using the SUPAHasPolicyMetadata
             aggregation.";
    }
```

```
    container supa-policy-metadata-decorator-version-container {
        description
            "This is a container to collect all object instances of
             type SUPAPolicyVersionMetadataDef.";
        list supa-policy-metadata-decorator-version-list {
            key supa-policy-metadata-id;
            uses supa-policy-metadata-decorator-type;
            description
                "A list of all supa-policy-metadata-decorator-version
                instances in the system. Instances of subclasses
                will be in a separate list.";
        }
    }

    identity SUPA-HAS-POLICY-METADATA-ASSOC {
        description
            "The identity corresponding to a
             SUPAHasPolicyMetadataDetail association class
             object instance.";
    }

    grouping supa-has-policy-metadata-detail {
        leaf supa-policy-ID {
            type string;
            description
                "This is a globally unique ID for this association
                 instance in the overall policy system.";
        }
        leaf entity-class {
            type identityref {
                base SUPA-HAS-POLICY-METADATA-ASSOC;
            }
            default SUPA-HAS-POLICY-METADATA-ASSOC;
            description
                "The identifier of the class of this assocation.";
        }
        leaf supa-has-policy-metadata-object-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  POLICY-OBJECT-TYPE)";
            description
                "This is a reference from the SUPAPolicyObject object
                 instance that is aggregating SUPAPolicyMetadata object
                 instances using the SUPAHasPolicyMetadata aggregation.
                 This SUPAPolicyMetadataDetail association class is
                 used to define part of the semantics of the
                 SUPAHasPolicyMetadata aggregation. For example, it can
                 define which SUPAPolicyMetadata object instances can
```

```
                be aggregated by this particular SUPAPolicyObject
                object instance.";
        }
```

```
        leaf supa-has-policy-metadata-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  POLICY-METADATA-TYPE)";
            description
                "This is a reference from the SUPAPolicyMetadata object
                 instance(s) that are being aggregated by this
                 SUPAPolicyObject object instance using the
                 SUPAHasPolicyMetadata aggregation. The class
                 SUPAPolicyMetadataDetail association class is used to
                 define part of the semantics of the
                 SUPAHasPolicyMetadata aggregation. For example, it can
                 define which SUPAPolicyMetadata object instances can
                 be aggregated by this particular SUPAPolicyObject
                 object instance.";
        }
        leaf supa-policy-metadata-detail-is-applicable {
            type boolean;
            description
                "This attributes controls whether the associated
                 metadata is currently considered applciable to this
                 policy object; this enables metadata to be turned on
                 and off when needed without disturbing the structure
                 of the object that the metadata applies to.";
        }
        leaf-list supa-policy-metadata-detail-constraint {
            type string;
            description
                "A list of constraints, expressed as strings
                in the language defined by the
                supa-policy-metadata-detail-encoding.";
        }
        leaf supa-policy-metadata-detail-encoding {
            type string;
            description
                "The langauge used to encode the constraints
                relevant to the relationship between the metadata
                and the underlying policy object.";
        }
        description
            "This is a concrete association class that defines the
             semantics of the SUPAPolicyMetadata aggregation. This
             enables the attributes and relationships of the
             SUPAPolicyMetadataDetail class to be used to constrain
             which SUPAPolicyMetadata objects can be aggregated by
             this particular SUPAPolicyObject instance.";
    }
```

```
    container supa-policy-metadata-detail-container {
        description
            "This is a container to collect all object instances of
             type SUPAPolicyMetadataDetail.";
        list supa-policy-metadata-detail-list {
            key supa-policy-ID;
            uses supa-has-policy-metadata-detail;
            description
                "This is a list of all supa-policy-metadata-detail
                instances in the system.  Instances of subclasses
                will be in a separate list.
                Note that this policy is made concrete for exemplary
                purposes.  To be useful, it almost certainly needs
                refinement.";
        }
    }

    identity SUPA-HAS-POLICY-COMPONENT-DECORATOR-ASSOC {
        description
            "The identity corresponding to a SUPAHasMetadataDecorator
             association class object instance.";
    }

    grouping supa-has-decorator-policy-component-detail {
        leaf supa-policy-ID {
            type string;
            description
                "This is a globally unique ID for this association
                 instance in the overall policy system.";
        }
        leaf entity-class {
            type identityref {
                base SUPA-HAS-POLICY-COMPONENT-DECORATOR-ASSOC;
            }
            default SUPA-HAS-POLICY-COMPONENT-DECORATOR-ASSOC;
            description
                "The identifier of the class of this assocation.";
        }
        leaf supa-policy-component-decorator-ptr {
            type instance-identifier;
             must "derived-from-or-self (deref(.)/entity-class,
                   SUPA-POLICY-COMPONENT-DECORATOR-TYPE)";
            description
                "This associates the SUPAPolicyComponentStructure
                 object instance participating in a
                 SUPAHasDecoratedPolicyComponent aggregation to the
                 SUPAHasDecoratedPolicyComponentDetail association
                 class that provides the semantics of this aggregation.
```

```
                This defines the object class that this
                instance-identifier points to.";
        }
```

```
        leaf supa-policy-component-ptr {
            type instance-identifier;
             must "derived-from-or-self (deref(.)/entity-class,
                   SUPA-POLICY-COMPONENT-TYPE)";
            description
                "This associates the SUPAPolicyComponentDecorator
                 object instance participating in a
                 SUPAHasDecoratedPolicyComponent aggregation to the
                 SUPAHasDecoratedPolicyComponentDetail association
                 class that provides the semantics of this aggregation.
                 This defines the object class that this
                 instance-identifier points to.";
        }
        leaf-list supa-has-decorator-constraint {
            type string;
            description
                "A constraint expression applying to this association
                 between a policy component decorator and the
                 decorated component.";
        }
        leaf supa-has-decorator-constraint-encoding {
            type string;
            description
                "The language in which the constraints on the
                 policy component-decoration is expressed.";
        }
        description
            "This is a concrete association class that defines the
             semantics of the SUPAHasDecoratedPolicyComponent
             aggregation. The purpose of this class is to use the
             Decorator pattern to detemine which
             SUPAPolicyComponentDecorator object instances, if any,
             are required to augment the functionality of the concrete
             subclass of SUPAPolicyClause that is being used.";
    }

    container supa-policy-component-decorator-detail-container {
        description
            "This is a container to collect all object instances of
             type SUPAPolicyComponentDecoratorDetail.";
        list supa-policy-component-decorator-detail-list {
            key supa-policy-ID;
            uses supa-has-decorator-policy-component-detail;
            description
                "This is a list of all
                 supa-policy-component-decorator-details.";
        }
    }
```

```
    identity SUPA-HAS-POLICY-SOURCE-ASSOC {
        description
            "The identity corresponding to a SUPAHasPolicySource
             association class object instance.";
    }

    grouping supa-has-policy-source-detail {
        leaf supa-policy-ID {
            type string;
            description
                "This is a globally unique ID for this association
                 instance in the overall policy system.";
        }
        leaf entity-class {
            type identityref {
                base SUPA-HAS-POLICY-SOURCE-ASSOC;
            }
            default SUPA-HAS-POLICY-SOURCE-ASSOC;
            description
                "The identifier of the class of this assocation.";
        }
        leaf supa-policy-source-structure-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  POLICY-STRUCTURE-TYPE)";
            description
                "This associates the SUPAPolicyStructure object
                 instance participating in a SUPAHasPolicySource
                 aggregation to the SUPAHasPolicySourceDetail
                 association class that provides the semantics of
                 this aggregation. This defines the object class
                 that this instance-identifier points to.";
        }
        leaf supa-policy-source-ptr {
            type instance-identifier;
             must "derived-from-or-self (deref(.)/entity-class,
                   SUPA-POLICY-SOURCE-TYPE)";
            description
                "This associates the SUPAPolicySource object
                 instance participating in a SUPAHasPolicySource
                 aggregation to the SUPAHasPolicySourceDetail
                 association class that provides the semantics of
                 this aggregation. This defines the object class
                 that this instance-identifier points to.";
        }
        leaf supa-policy-source-is-authenticated {
            type boolean;
            description
                "If the value of this attribute is true, then this
```

```
                SUPAPolicySource object has been authenticated by
                this particular SUPAPolicyStructure object.";
        }
```

```
        leaf supa-policy-source-is-trusted {
            type boolean;
            description
                "If the value of this attribute is true, then this
                 SUPAPolicySource object has been verified to be
                 trusted by this particular SUPAPolicyStructure
                 object.";
        }
        description
            "This is an association class, and defines the semantics of
             the SUPAHasPolicySource aggregation. The attributes and
             relationships of this class can be used to define which
             SUPAPolicySource objects can be attached to which
             particular set of SUPAPolicyStructure objects.";
    }

    container supa-policy-source-detail-container {
    description
            "This is a container to collect all object instances of
             type SUPAPolicySourceDetail.";
        list supa-policy-source-detail-list {
            key supa-policy-ID;
            uses supa-has-policy-source-detail;
            description
                "This is a list of all supa-policy-source-detail
                 objects.";
        }
    }

    identity SUPA-HAS-POLICY-TARGET-ASSOC {
        description
            "The identity corresponding to a SUPAHasPolicyTarget
             association class object instance.";
    }

    grouping supa-has-policy-target-detail {
        leaf supa-policy-ID {
            type string;
            description
                "This is a globally unique ID for this association
                 instance in the overall policy system.";
        }
        leaf entity-class {
            type identityref {
                base SUPA-HAS-POLICY-TARGET-ASSOC;
            }
            default SUPA-HAS-POLICY-TARGET-ASSOC;
            description
```

```
                "The identifier of the class of this assocation.";
        }
```

```
        leaf supa-policy-target-structure-ptr {
            type instance-identifier;
             must "derived-from-or-self (deref(.)/entity-class,
                   POLICY-STRUCTURE-TYPE)";
            description
                "This associates the SUPAPolicyStructure object
                 instance participating in a SUPAHasPolicyTarget
                 aggregation to the SUPAHasPolicyTargetDetail
                 association class that provides the semantics of
                 this aggregation. This defines the object class
                 that this instance-identifier points to.";
        }
        leaf supa-policy-target-ptr {
            type instance-identifier;
             must "derived-from-or-self (deref(.)/entity-class,
                   SUPA-POLICY-TARGET-TYPE)";
            description
                "This associates the SUPAPolicyTarget object
                 instance participating in a SUPAHasPolicyTarget
                 aggregation to the SUPAHasPolicyTargetDetail
                 association class that provides the semantics of
                 this aggregation. This defines the object class
                 that this instance-identifier points to.";
        }
        leaf supa-policy-source-is-authenticated {
            type boolean;
            description
                "If the value of this attribute is true, then this
                 SUPAPolicyTarget object has been authenticated by
                 this particular SUPAPolicyStructure object.";
        }
        leaf supa-policy-source-is-enabled {
            type boolean;
            description
                "If the value of this attribute is true, then this
                 SUPAPolicyTarget object is able to be used as a
                 SUPAPolicyTarget. This means that it has agreed to
                 play the role of a SUPAPolicyTarget, and that it is
                 able to either process (directly or with the aid of a
                 proxy) SUPAPolicies, or receive the results of a
                 processed SUPAPolicy and apply those results to
                 itself.";
        }
        description
            "This is an association class, and defines the semantics of
             the SUPAHasPolicyTarget aggregation. The attributes and
             relationships of this class can be used to define which
             SUPAPolicyTarget objects can be attached to which
```

```
            particular set of SUPAPolicyStructure objects.";
    }
```

```
    container supa-policy-target-detail-container {
    description
            "This is a container to collect all object instances of
             type SUPAPolicyTargetDetail.";
        list supa-policy-target-detail-list {
            key supa-policy-ID;
            uses supa-has-policy-target-detail;
            description
                "This is a list of all supa-policy-target-detail
                 objects.";
        }
    }

    identity SUPA-HAS-POLICY-CLAUSE-ASSOC {
        description
            "The identity corresponding to a SUPAHasPolicyClause
             association class object instance.";
    }

    grouping supa-has-policy-clause-detail {
        leaf supa-policy-ID {
            type string;
            description
                "This is a globally unique ID for this association
                 instance in the overall policy system.";
        }
        leaf entity-class {
            type identityref {
                base SUPA-HAS-POLICY-CLAUSE-ASSOC;
            }
            default SUPA-HAS-POLICY-CLAUSE-ASSOC;
            description
                "The identifier of the class of this assocation.";
        }
        leaf supa-policy-clause-structure-ptr {
            type instance-identifier;
             must "derived-from-or-self (deref(.)/entity-class,
                   POLICY-STRUCTURE-TYPE)";
            description
                "This associates the SUPAPolicyStructure object
                 instance participating in a SUPAHasPolicyClause
                 aggregation to the SUPAHasPolicyClauseDetail
                 association class that provides the semantics of
                 this aggregation. This defines the object class
                 that this instance-identifier points to.";
        }
        leaf supa-policy-clause-ptr {
            type instance-identifier;
```

```
            must "derived-from-or-self (deref(.)/entity-class,
                  SUPA-POLICY-CLAUSE-TYPE)";
```

```
            description
                "This associates the SUPAPolicyClause object
                 instance participating in a SUPAHasPolicyClause
                 aggregation to the SUPAHasPolicyClauseDetail
                 association class that provides the semantics of
                 this aggregation. This defines the object class
                 that this instance-identifier points to.";
        }
        description
            "This is an association class, and defines the semantics of
             the SUPAHasPolicyClause aggregation. The attributes and
             relationships of this class can be used to define which
             SUPAPolicyTarget objects can be attached to which
             particular set of SUPAPolicyStructure objects.
             Every SUPAPolicyStructure object instance MUST aggregate
             at least one SUPAPolicyClause object instance. However,
             the converse is NOT true. For example, a SUPAPolicyClause
             could be instantiated and then stored for later use in a
             policy repository.";
    }

    container supa-policy-clause-detail-container {
        description
            "This is a container to collect all object instances of
             type SUPAPolicyClauseDetail.";
        list supa-policy-clause-detail-list {
            key supa-policy-ID;
            uses supa-has-policy-clause-detail;
            description
                "This is a list of all supa-policy-clause-detail
                 objects.";
        }
    }

    identity SUPA-HAS-POLICY-EXEC-ACTION-ASSOC {
        description
            "The identity corresponding to a
             SUPAHasPolExecFailActionToTake association class
             object instance.";
    }

    grouping supa-has-policy-exec-action-detail {
        leaf supa-policy-ID {
            type string;
            description
                "This is a globally unique ID for this association
                 instance in the overall policy system.";
        }
        leaf entity-class {
```

```
            type identityref {
                base SUPA-HAS-POLICY-EXEC-ACTION-ASSOC;
            }
```

```
            default SUPA-HAS-POLICY-EXEC-ACTION-ASSOC;
            description
                "The identifier of the class of this assocation.";
        }
        leaf supa-policy-structure-action-src-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  POLICY-STRUCTURE-TYPE)";
            description
                "This associates the SUPAPolicyStructure object
                 instance participating in a
                 SUPAHasPolExecFailActionToTake association to the
                 SUPAHasPolExecFailActionToTakeDetail association
                 class that provides the semantics of this
                 aggregation. This defines the object class that
                 this instance-identifier points to.";
        }
        leaf supa-policy-structure-action-dst-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  POLICY-STRUCTURE-TYPE)";
            description
                "This associates a SUPAPolicyAction object
                 instance participating in a
                 SUPAHasPolExecFailActionToTake association to the
                 SUPAHasPolExecFailActionToTakeDetail association
                 class that provides the semantics of this
                 aggregation. This defines the object class that
                 this instance-identifier points to.";
        }
        leaf supa-policy-exec-fail-take-action-encoding {
            type policy-data-type-id-encoding-list;
            description
                "This defines how to find the set of SUPA Policy
                 Action objects contained in each element of the
                 supa-policy-exec-fail-take-action-name attribute
                 object.";
        }
        leaf-list supa-policy-exec-fail-take-action-name {
            type string;
            description
                "This identifies the set of SUPA Policy Actions to take
                 if the SUPAPolicyStructure object that owns this
                 association failed to execute properly. The
                 interpretation of this string attribute is defined by
                 the supa-policy-exec-fail-take-action-encoding class
                 attribute.";
        }
```

```
        description
            "This is an association class, and defines the semantics of
             the SUPAHasPolExecFailTakeAction association. The
             attributes and relationships of this class can be used to
             determine which SUPA Policy Action objects are executed in
             response to a failure of the SUPAPolicyStructure object
             instance that owns this association.";
    }

    container supa-policy-exec-fail-take-action-detail-container {
        description
            "This is a container to collect all object instances of
             type SUPAPolExecFailActionToTakeDetail.";
        list supa-policy-exec-fail-take-action-detail-list {
            key supa-policy-ID;
            uses supa-has-policy-exec-action-detail;
            description
                "This is a list of all
                 supa-has-policy-exec-action-detail objects.";
        }
    }
}

<CODE ENDS>
```

6. IANA Considerations

No IANA considerations exist for this document.

7. Security Considerations

TBD

8. Acknowledgments

This document has benefited from reviews, suggestions, comments and proposed text provided by the following members, listed in alphabetical order: Qin Wu.

9. References

This section defines normative and informative references for this document.

9.1. Normative References

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

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

[RFC6991] Schoenwaelder, J., "Common YANG Data Types", RFC 6991, July 2013.

9.2. Informative References

[1] Strassner, J., Halpern, J., Coleman, J., "Generic Policy Information Model for Simplified Use of Policy Abstractions (SUPA)", draft-strassner-supa-generic-policy-info-model-05 March 21, 2016

Authors' Addresses

Joel Halpern
Ericsson
P. O. Box 6049
Leesburg, VA 20178
Email: joel.halpern@ericsson.com

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