Internet Engineering Task Force
INTERNET DRAFT
9-March-2000

Jamie Jason
Intel Coroporation

IPsec Configuration Policy Model
draft-ietf-ipsp-config-policy-model-00.txt

Status of this Memo

Abstract

This document presents an object-oriented model of low-level IPsec policy designed to:

- facilitate agreement about the content and semantics of IPsec policy
- enable derivations of task-specific representations of IPsec policy such as storage schema, distribution representations, and policy specification languages used to configure IPsec-enabled endpoints

The schema described in this document models the IKE phase one parameters as described in [1] and the IKE phase two parameters for the IPsec Domain of Interpretation as described in [2, 3, 4, 5].

Table of Contents

Status of this Memo....1
Abstract....1
Table of Contents....2
1. Introduction....4
2. UML Conventions....4
3. Endpoint Classes....6
3.1. The Class Endpoint....6
3.2. The Class FQDNEndpoint....6
3.3. The Class IPv4Endpoint....6
3.4. The Class IPv6Endpoint....7
4. IPsec Policy Classes....8
4.1. The Class IPsecPolicyList....9
4.2. The Class IPsecPolicy....9
4.3. The Class IPInterface....9
5. IPsec Rule Classes....10
5.1. The Class SecurityAssociationRule....10
6. IPSec Condition Classes....11
6.1. The Class SecurityAssociationCondition....11
6.2. The Class SecurityAssociationConditionExpression....12
7. IPSec Filter Classes....13
7.1. The Class SecurityAssociationFilter....13
7.2. The Class PortFilter....14
7.3. The Class PortRangeFilter....14
7.4. The Class ProtocolFilter....14
7.5. The Class AddressFilter....15
7.6. The Class EndpointFilter....15
7.7. The Class IPv4RangeFilter....15
7.8. The Class IPv6RangeFilter....16
8. IKE and IPsec Action Classes....17
8.1. The Class SecurityAssociationAction....18
8.2. The Class IKEAction....19
8.3. The Class IPsecAction....20
8.4. The Class IPsecTransportAction....20
8.5. The Class IPsecTunnelAction....21
8.6. The Class IPsecBypassAction....21
8.6. The Class IPsecDiscardAction....21
9. IKE and IPsec Proposal Classes....21
9.1. The Class SecurityAssociationProposal....22
9.2. The Class IKEProposal....22
9.3. The Class IPsecProposal....23
9.4. The Class IPsecTransform....24
9.5. The Class ESPTransform....24
9.6. The Class AHTransform....25
9.7. The Class IPCompTransform....25
10. Diffie-Hellman Classes....26
10.1. The Class DiffieHellmanGroup....27

10.2. The Class NewGroupInfo.....27
10.3. The Class NewMODPGroupInfo.....27
10.4. The Class NewECGroupInfo.....27
10.5. The Class NewEC2NGroupInfo.....28
10.6. The Class NewECPGroupInfo.....28

11. Security Considerations.....28
12. Intellectual Property.....28
13. Acknowledgments.....29
14. References.....29
15. Disclaimer.....30
16. Author's Address.....30
17. Full Copyright Statement.....30

1. Introduction

Internet Protocol security (IPsec) policy may assume a variety of forms as it travels from storage to distribution point to decision point. At each step, it needs to be represented in a way that is convenient for the current task. For example, the policy could exist as, but is not limited to:

- a Lightweight Directory Access Protocol (LDAP) [6] schema in a directory
- an on-the-wire representation over a transport protocol like the Common Object Policy Service (COPS) [7]
- a text-based policy specification language [8] suitable for editing by an administrator
- an Extensible Markup Language (XML) document

Each of these task-specific representations should be derived from a canonical representation that precisely specifies the content and semantics of the IPsec policy. The purpose of this document is to abstract IPsec policy into a task-independent representation that is not constrained by any particular task-dependent representation.

This document is organized as follows:

- Section 2 provides a quick introduction to the Unified Modeling Language (UML) graphical notation conventions used in this document.
- Section 3 defines the endpoint class, a utility class that is used as a building block for other classes.
- Section 4 defines the IPsec policy and associated classes.
- Section 5 defines the rule class.
- Section 6 defines the condition and condition expression classes.
- Section 7 defines the filter classes.
- Section 8 defines the IKE and IPsec action classes.
- Section 9 defines the IKE and IPsec proposal classes.
- Section 10 defines the Diffie-Hellman group class.

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 [9].

2. UML Conventions

For this document, a UML static class diagram was chosen as the canonical representation for the IPsec policy model. The reason behind this decision is that UML provides a task-independent way to model systems. A treatise on the graphical notation used in UML is beyond the scope of this paper. However, given the use of ASCII drawing for UML static class diagrams, a description of the notational conventions used in this document is in order:

- Boxes represent classes, with class names in brackets ([]) representing a virtual class. For example, in the action classes diagram, IKEAction is a concrete class while SecurityAssociationAction is a virtual class.
- A line that terminates with a "o" denotes aggregation. Aggregation denotes classes with independent lifetimes. An aggregated object exists independently of the object that references it. For example, in the action classes diagram a SecurityAssocationProposal object exists independently of the SecurityAssociationAction object which references it.
- A line that terminates with an "x" denotes composition. Composition denotes classes with coincident lifetimes. This implies that the lifetime of the contained object is the same as the object that contains it.
- Next to a line appears a multiplicity. Multiplicities indicate the number of objects contained (or referenced) as well as the number of object that can contain (or reference) a particular object. The multiplicity may be:
  - a range in the form "lower bound..upper bound" indicating the minimum and maximum number of objects. For example, in the action classes diagram, an IPsecAction may contain either 0 or 1 DiffieHellmanGroup objects(essentially noting that the DiffieHellmanGroup is optional).
  - a number that indicates the exact number of objects. For example, in the proposal classes diagram, an IKEProposal has 1 and only 1 DiffieHellmanGroup. Using a number is equivalent to number..number.
  - an asterisk indicating any number of objects, including zero. For example, in the action classes diagram, a SecurityAssociationProposal object may be referenced by 0 to n SecurityAssociationAction objects. Using an asterisk is equivalent to 0..n.
  - the letter n indicating from 1 to many. For example, in the action classes diagram, a SecurityAssociationAction references 1 to many SecurityAssociationProposals. Using the letter n is equivalent to 1..n.
- A line that terminates with an arrow (<, , ^, v) denotes generalization (inheritance) with the arrow pointing to the parent class. For example, in the action classes diagram the SecurityAssociationAction class is a generalization of the

IKEAction class (or said another way, the IKEAction class derives from the SecurityAssociationAction class).

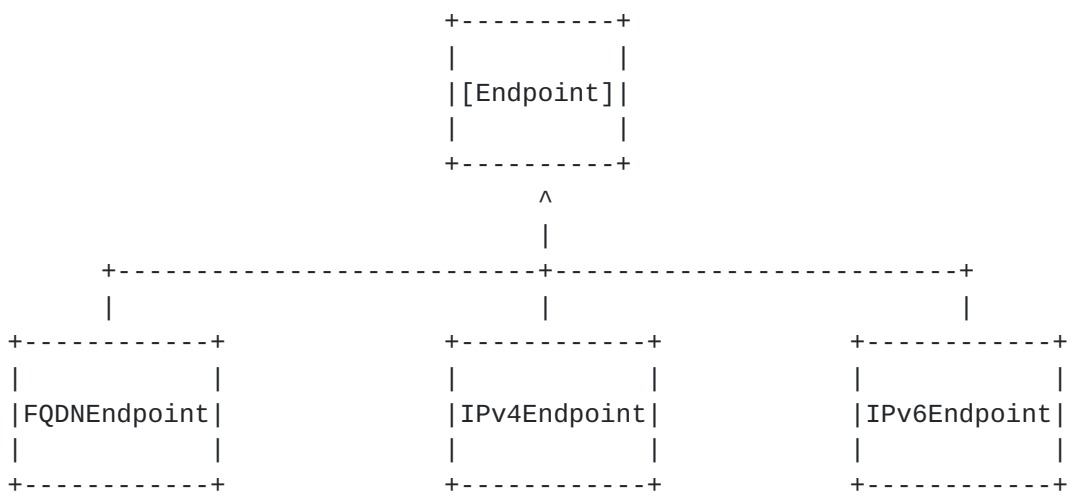
- Occasionally there may be some text, or a reference to some text, enclosed by braces ({}). This indicates a constraint. Constraints are used to constrain the meaning of diagram so that

a diagram does not provide the ability to define something that does not make sense. For example, in the action classes diagram there is a constraint placed upon the DiffieHellmanGroup class such that it is only used if the IPsecAction specifies the user of Perfect Forward Secrecy.

It should be noted that the UML static class diagram presented is a conceptual view of IPsec policy designed to aid in understanding. It does not necessarily get translated class for class into another representation. For example, an LDAP implementation may flatten out the representation to fewer classes (because of the inefficiency of following references).

3. Endpoint Classes

An endpoint is an abstraction used to represent an IP address or hostname. This class is used as a building block in further classes.



3.1. The Class Endpoint

The Endpoint class is used as an abstract base class from which concrete endpoint classes are expected to derive from.

3.2. The Class FQDNEndpoint

The FQDNEndpoint class is used to represent endpoints that can be expressed using a DNS name. It contains the following attribute:

NAME	name
DESCRIPTION	Either a fully-qualified or wild-carded (partially or fully) domain name.
TYPE	string
VALUE	MAY either be fully-qualified (for example,

runner.jf.intel.com) or wild-carded (for example,
*.intel.com).

3.3. The Class IPv4Endpoint

The IPv4Endpoint class is used to represent endpoints that can be expressed using an IPv4 address. It contains the following attribute:

NAME	address
DESCRIPTION	The IPv4 address.
TYPE	unsigned 32-bit integer
VALUE	0x00000000 (i.e., 0.0.0.0) - used to specify any IP address (i.e., a totally wild-carded address or "*"). Any other value specifies an IPv4 address.

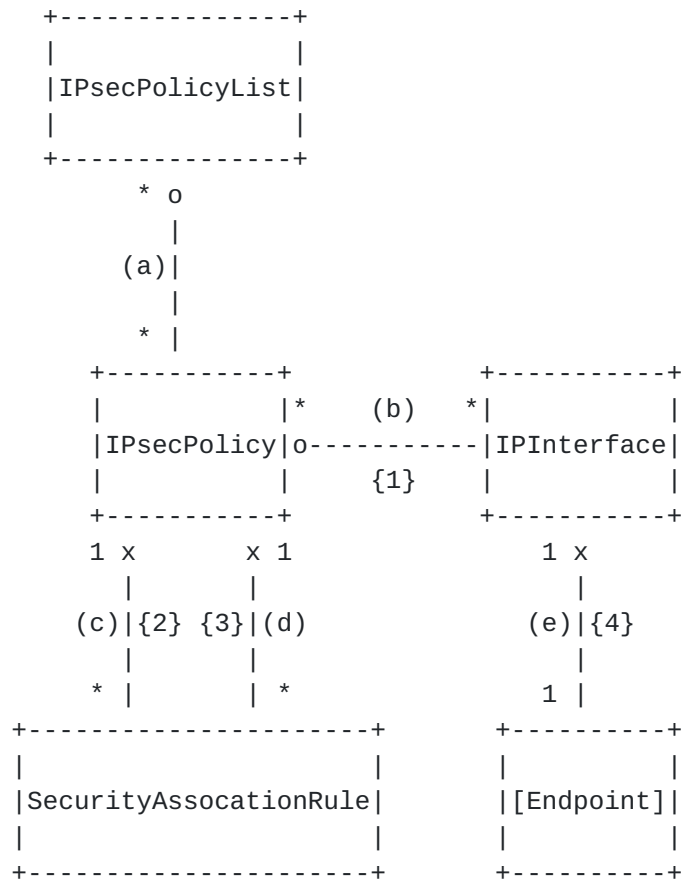
3.4. The Class IPv6Endpoint

The IPv6Endpoint class is used to represent endpoints that can be expressed using an IPv6 address. It contains the following attribute:

NAME	address
DESCRIPTION	The IPv6 address.
TYPE	octet[16]
VALUE	all zero's (i.e., 0:0:0:0:0:0:0:0) - used to specify any IP address (i.e., a totally wild-carded address or "*"). Any other value specifies an IPv6 address.

4. IPsec Policy Classes

The IPsec policy classes represent the set of policies that are contained on a system. In addition, they indicate the active policies as well as associate a policy with a particular interface on a system



(a) Policies
(b) TargetedInterface
(c) IKERules
(d) IPsecRules
(e) Identity

{1} 1. If the policy is marked as enabled, then the IPsecPolicy object MUST reference an IPInterface object.
2. For each interface, there is only one IPsec policy marked as enabled.

{2} IKE rules are ordered and are considered logically ORed. Rule search will stop once a rule that matches the input criteria is found.

{3} IPsec rules are ordered and are considered logically ORed. Rule search will stop once a rule that matches the input

criteria is found.

{4} If the endpoint type is an FQDN, then the DNS name MUST be fully-qualifed (i.e., no wild-card values allowed). If the endpoint type is an IPv4 or IPv6 address, then the address value MUST NOT be the wild-card address.

4.1. The Class IPsecPolicyList

The IPsecPolicyList class is a container for all of the policies on a particular system. It contains the following reference:

NAME	Policies
DESCRIPTION	The policies installed on a particular system. Note that there is a distinction between a policy being installed on a system and actually being actively enforced (see the IPsecPolicy class).

Note: an IPsecPolicyList MAY contain no policies. Additionally, a policy MAY be defined which is not in any policy list. The latter case is only relevant for a management station - in other words, an IPsec policy has been created but it has not yet been targeted to a system.

4.2. The Class IPsecPolicy

The IPsecPolicy class is a container for all of the rules used to enforce the policy. It contains the following attribute/references:

NAME	enabled
DESCRIPTION	Indicates whether or not the policy is enabled (i.e., is actively being enforced). As stated in the constraint {1}, if the policy is enabled, it MUST be associated with a particular interface on the system. This allows for different policies to be enforced on different interfaces.
TYPE	boolean
VALUE	true - policy is currently enabled false - policy is currently disabled

NAME	TargetedInterface
DESCRIPTION	The interface on the system for which this policy is to be enforced. As stated in the constraint {1}, for each interface there is only one policy enabled at any one given time.

NAME	IKERules
DESCRIPTION	The rules which govern when and how to perform IKE phase 1 negotiation. These rules are an ordered list and are logically ORed. When processing the rules, the first rule matched is the one used.

NAME	IPsecRules
DESCRIPTION	The rules which govern when and how to perform IKE phase 2 negotiation. These rules are an ordered list

and are logically ORed. When processing the rules, the first rule matched is the one used.

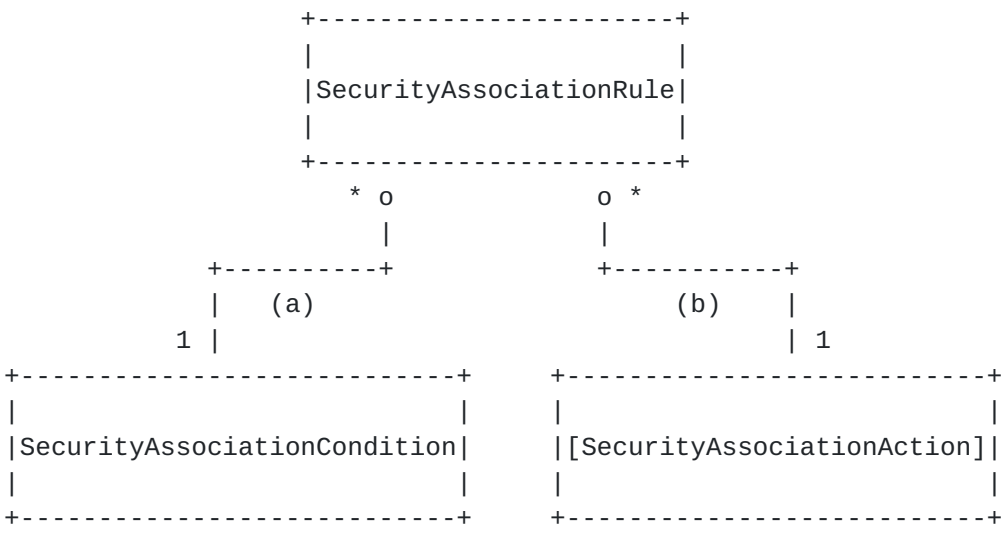
4.3. The Class IPInterface

The IPInterface class is used to represent an interface on the system. It contains the following reference:

NAME	Identity
DESCRIPTION	Indicates the IP address or DNS name assigned to the interface. No wild-card values are allowed for the endpoint object.

5. IPsec Rule Classes

The IPsec rule class is used to associate a condition with the action which is to be performed when the condition evaluates to true.



(a) Condition
(b) Action

5.1. The Class SecurityAssociationRule

The SecurityAssociationRule class is used to associate a condition with the IKE/IPsec action information that is to be used during the negotiation. It contains the following attribute/references:

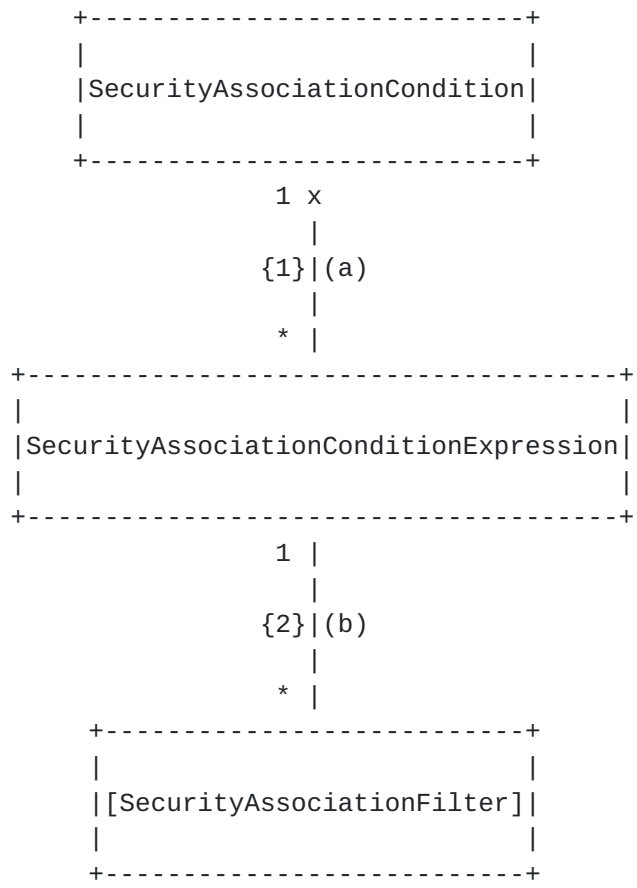
NAME	enabled
DESCRIPTION	Indicates whether or not the rule is enabled.
TYPE	boolean
VALUE	true - rule is currently enabled false - rule is currently disabled

NAME	Condition
DESCRIPTION	The condition, when evaluated against the given input, that MUST evaluate to true in order for the associated action to be performed.

```
   NAME         Action
   DESCRIPTION  The security association negotiation parameters to use
                when the associated condition evaluates to true.
```

6. IPSec Condition Classes

The condition class is used to determine when the associated IKE or IPsec action is to be performed.



(a) Expressions
(b) Filters

{1} If using disjunctive normal form (DNF), each expression is logically ORed. If using conjunctive normal form (CNF), each expression is logically ANDed.
{2} If using DNF, each filter is logically ANDed. If using CNF, each filter is logically ORed.

6.1. The Class SecurityAssociationCondition

The SecurityAssociationCondition class specifies the criteria that is applied to the input information to determine if a particular condition is met. It contains the following attributes/references:

NAME negated
DESCRIPTION Indicates whether or not the result of the rule

evaluation is to be negated.

TYPE	boolean
VALUE	true - condition evaluation result is to be negated

false - condition evaluation result is not to be negated

NAME	useDNF
DESCRIPTION	Indicates whether or not the rule is specified in DNF or CNF form.
TYPE	boolean
VALUE	true - condition is expressed as DNF. The expressions within the condition are logically ORed. The filters within an expression are logically ANDed. false - condition is expressed as CNF. The expressions within the condition are logically ANDed. The filters within an expression are logically ORed.

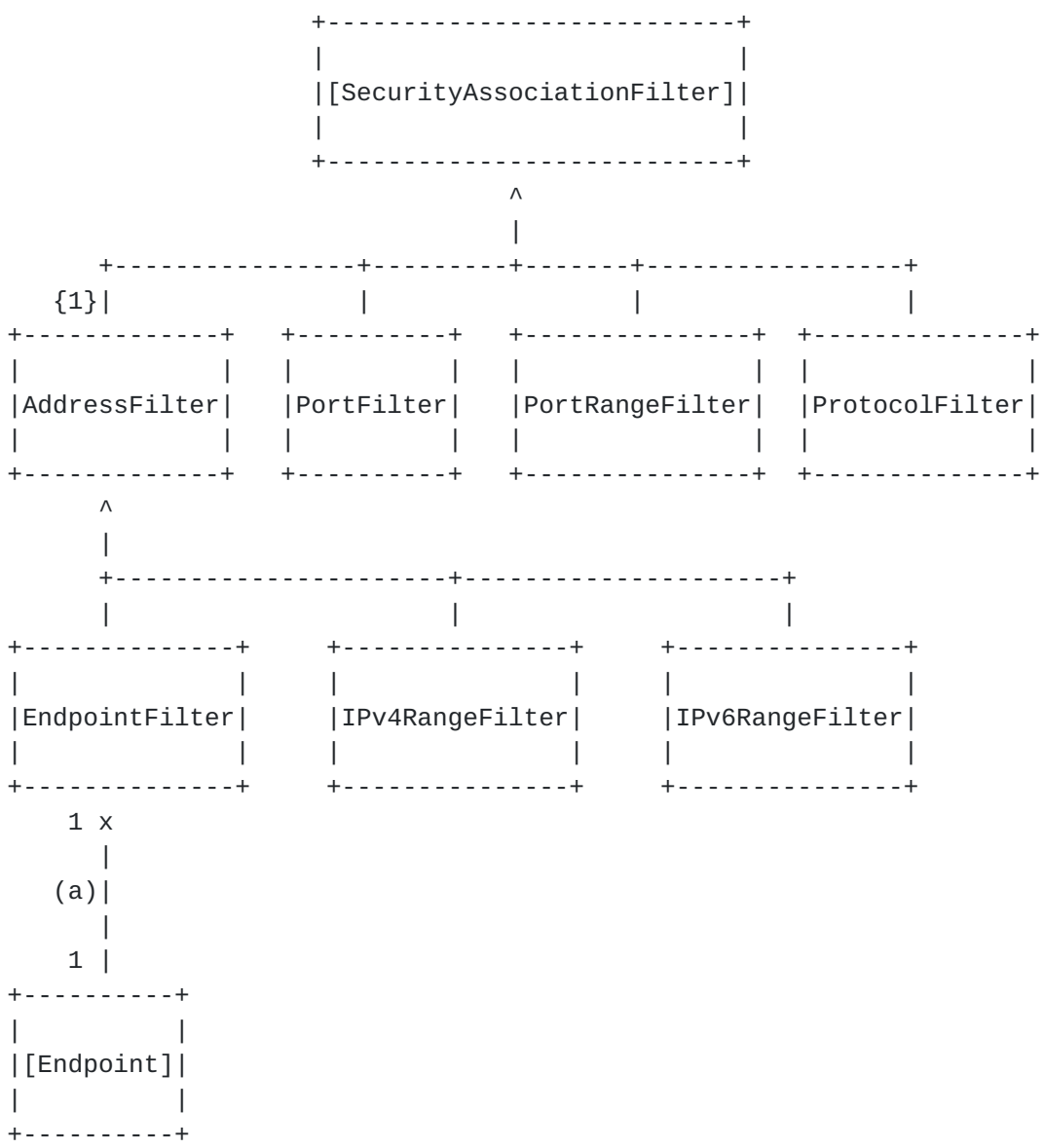
6.2. The Class SecurityAssociationConditionExpression

The SecurityAssociationConditionExpression class is used to combine several filters, which together constitute one logical expression. It contains the following reference:

NAME	Filters
DESCRIPTION	The set of filters, which combined, are used to represent the expression. When using DNF, these filters are logically ANDed. When using CNF, these filters are logically ORed.

7. IPSec Filter Classes

The filter classes are used to specify individual criteria which MUST be met before a condition will evaluate to true.



(a) Identity

{1} When the rule is for an IKE phase one negotiation, the AddressFilter is the only type of filter allowed.

7.1. The Class SecurityAssociationFilter

The SecurityAssociationFilter class is used as an abstract base class from which all concrete filter class are expected to derive

from. It contains the following attribute:

NAME	negated
DESCRIPTION	Indicates whether or not the result of the filter evaluation is to be negated.

TYPE	boolean
VALUE	true - filter evaluation is to be negated false - filter evaluation is not to be negated

7.2. The Class PortFilter

The PortFilter class specifies a filter that is based upon a single port value. It contains the following attributes:

NAME	applyToSource
DESCRIPTION	Indicates whether or not the port specified is to be interpreted as a source port or a destination port.
TYPE	boolean
VALUE	true - the port specified is to be interpreted as a source port false - the port specified is to be interpreted as a destination port

NAME	port
DESCRIPTION	Specifies the port value.
TYPE	unsigned 16-bit integer
VALUE	0 - wild-card port (i.e., any port matches). Any other value specifies a specific port.

7.3. The Class PortRangeFilter

The PortRangeFilter class specifies a filter that is based upon a range of port values. The port range is to be interpreted as inclusive. It contains the following attributes:

NAME	applyToSource
DESCRIPTION	Indicates whether or not the port specified is to be interpreted as a source port range or a destination port range.
TYPE	boolean
VALUE	true - the port range specified is to be interpreted as a source port range false - the port range specified is to be interpreted as a destination port range

NAME	firstPort
DESCRIPTION	Specifies the first port in the range.
TYPE	unsigned 16-bit integer

NAME	lastPort
DESCRIPTION	Specifies the last port in the range.
TYPE	unsigned 16-bit integer
VALUE	The lastPort attribute value MUST be greater than or equal to the firstPort attribute value.

7.4. The Class ProtocolFilter

The ProtocolFilter class specifies a filter that is based upon the IP protocol. It contains the following attribute:

NAME	protocol
DESCRIPTION	Specifies the IP protocol value.
TYPE	unsigned 8-bit integer
VALUE	0 - wild-card protocol (i.e., any protocol). Any other value specifies a specific protocol.

Note: if using DNF, it does not make sense to use a PortFilter or PortRangeFilter when using a ProtocolFilter that is not either UDP or TCP.

7.5. The Class AddressFilter

The AddressFilter class is used to represent filters which use a system's address or DNS name as a filter. It is used as an abstract base class from which specific address-based filters will be derived. The address filters are always used to specify the address/hostname of the destination machine. The reason is that the association of a policy with a particular interface implies the source address/hostname - one could look at the policy to interface mapping as another type of filter.

Note: for IKE rules, these are the only filter type allowed.

7.6. The Class EndpointFilter

The EndpointFilter class is used to represent a filter that specifies an individual interface on one system. It is used to specify an FQDN, an IPv4 address, or an IPv6 address. It contains the following reference:

NAME	Identity
DESCRIPTION	Specifies the FQDN or IP address to use for the filter. The value MAY be wild-carded (see the Endpoint class description).

7.7. The Class IPv4RangeFilter

The IPv4RangeFilter is used to represent a filter that specifies a range of IPv4 address. The range is to be interpreted as inclusive. It contains the following attributes:

NAME	firstAddress
DESCRIPTION	Specifies the first address in the range.
TYPE	unsigned 32-bit value

NAME	lastAddress

DESCRIPTION	Specifies the last address in the range.
TYPE	unsigned 32-bit value
VALUE	The lastAddress attribute value MUST be greater than or equal to the firstAddress attribute value.

7.8. The Class IPv6RangeFilter

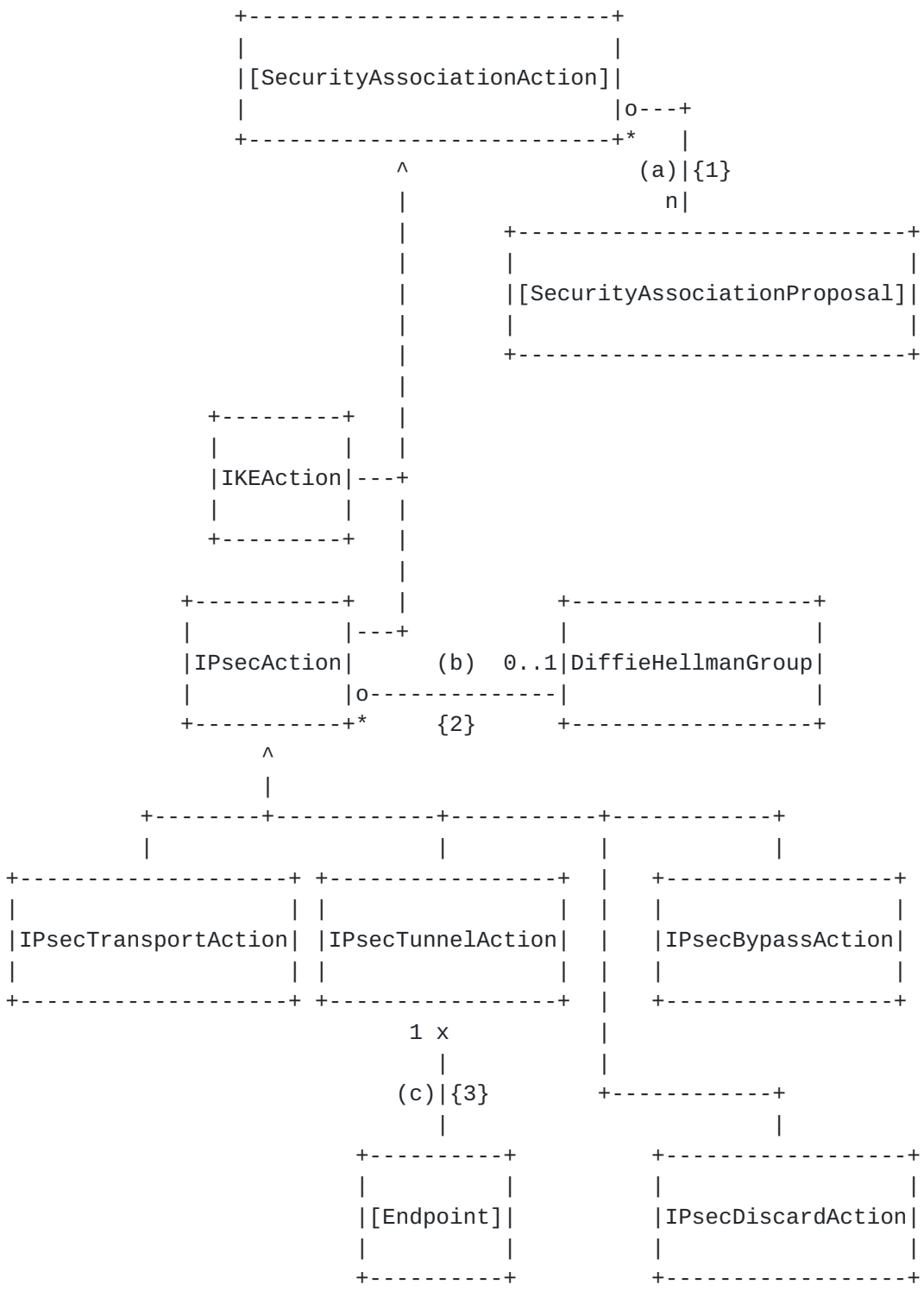
The IPv6RangeFilter is used to represent a filter that specifies a range of IPv6 address. The range is to be interpreted as inclusive. It contains the following attributes:

NAME	firstAddress
DESCRIPTION	Specifies the first address in the range.
TYPE	octet[16]

NAME	lastAddress
DESCRIPTION	Specifies the last address in the range.
TYPE	octet[16]
VALUE	The lastAddress attribute value MUST be greater than or equal to the firstAddress attribute value.

8. IKE and IPsec Action Classes

An action is a set of proposals combined with the security association level information that is used to protect a particular flow.



(a) Proposals
(b) IPsecGroup
(c) RemoteGateway

{1} 1. For an IKEAction object, these MUST be IKEPropsal objects. For an IPsecAction object, these MUST be IPsec Action objects.
2. SecurityAssociationProposal objects are ordered from most preferable to least preferable and are logically ORed. The mechanism by which ordering is accomplished is dependent upon the specific derivation of the IPsec schema.

{2} If not using Perfect Forward Secrecy (PFS), then the DiffieHellmanGroup object either does not exist or is ignored. Otherwise (PFS is used) if the DiffieHellmanGroup object is not present, then the Diffie-Hellman Group from Phase 1 will be used for Phase 2. Otherwise, use the DiffieHellmanGroup object.

{3} If the endpoint type is an FQDN, then the DNS name MUST be fully-qualifed (i.e., no wild-card values allowed).
If the endpoint type is an IPv4 or IPv6 address, then the address MUST NOT be the wild-card value.

8.1. The Class SecurityAssociationAction

The SecurityAssociationAction class contains the parameters that are common between the IKE and IPsec action classes. It contains the following attributes/references:

NAME	refreshThresholdSeconds
DESCRIPTION	Specifies the percentage of expiration (in other words, the refresh threshold) of an established SA's seconds lifetime at which to begin renegotiation of the SA.
TYPE	integer
VALUE	Valid values are in the range 1 to 100 inclusive. A value of 100 means that renegotiation does not occur until the seconds lifetime value has expired.

refreshThresholdSeconds is not a negotiated parameter.

NAME	refreshThresholdKilobytes
DESCRIPTION	Specifies the percentage of expiration of an established SA's kilobyte lifetime at which to begin renegotiation of the SA.
TYPE	integer
VALUE	Valid values are in the range 1 to 100 inclusive. A value of 100 means that renegotiation does not occur until the kilobyte lifetime value has expired.

refreshThresholdKilobytes is not a negotiated parameter.

NAME	minLifetimeSeconds
DESCRIPTION	Specifies the minimum SA seconds lifetime that will be accepted from a peer while negotiating an SA based upon

this action. The purpose of this value is to prevent denial-of-service attacks in which a peer can select an arbitrarily low seconds lifetime, causing the IKE server to perform renegotiations with correspondingly expensive Diffie-Hellman calculations.

TYPE unsigned 32-bit integer
VALUE 0 - indicates that there is no minimum lifetime enforced.
Any other value specifies a required minimum seconds lifetime.

minLifetimeSeconds is not a negotiated parameter.

NAME minLifetimeKilobytes
DESCRIPTION Specifies the minimum kilobyte lifetime that will be accepted from a negotiating peer while negotiating an SA based upon this action. The purpose of this value is to prevent denial-of-service attacks in which a peer can select an arbitrarily low kilobyte lifetime, causing the IKE server to perform renegotiations with correspondingly expensive Diffie-Hellman calculations.
TYPE unsigned 32-bit integer
VALUE 0 - indicates that there is no minimum lifetime enforced.
Any other value specifies a required minimum kilobyte lifetime.

minLifetimeKilobytes is not a negotiated parameter.

NAME trafficIdleTime
DESCRIPTION Specifies the amount of time in seconds an SA, negotiated using the containing action object, may remain idle (in other words, no traffic protected by the SA) before it is deleted.
TYPE unsigned 32-bit integer
VALUE 0 - there is no idle time detection. In other words, the expiration of the SA is solely dependent upon the expiration of one of the lifetime values.
Any other value specifies the number of seconds the SA may remain idle before it can be forcibly expired.

trafficIdleTime is not a negotiated parameter.

NAME Proposals
DESCRIPTION Specifies a logically ORed set of proposals, ORDERED from most preferable to least prefereable, which are used during negotiation of the SA. If the action is an IKEAction, then the set will contain IKEProposal objects. If the action is an IPsecAction, then the set will contain IPsecProposal objects. A SecurityAssociationAction object will reference one to many SecurityAssociationProposal objects. A SecurityAssociationProposal object MAY be referenced by

zero to many SecurityAssociationAction objects. See section 9 for a description of the SecurityAssociationProposal and derived classes.

8.2. The Class IKEAction

IKEAction is a specialization of the SecurityAssociationAction class and specifies the parameters unique to an IKE action. It contains the following attributes:

NAME	exchangeMode
DESCRIPTION	Specifies the negotiation mode that the IKE server will use for phase one.
TYPE	unsigned 16-bit integer
VALUE	1 - base mode 2 - main mode 4 - aggressive mode

NAME	refreshThresholdDerivedKeys
DESCRIPTION	Specifies the percentage of expiration of an established IKE SA's derived keys lifetime at which to begin renegotiation of the SA.
TYPE	integer
VALUE	Valid values are in the range 1 to 100 inclusive. A value of 100 means that renegotiation does not occur until the derived key lifetime value has expired.

refreshThresholdDerivedKeys is not a negotiated parameter.

8.3. The Class IPsecAction

IPsecAction is a specialization of the SecurityAssociationAction class and specifies the parameters unique to an IPsec action. It contains the following attributes/references:

NAME	usePfs
DESCRIPTION	Specifies whether or not PFS should be used when negotiating the phase two IPsec SA.
TYPE	boolean
VALUE	true false

NAME	IPsecGroup
DESCRIPTION	If PFS should be used during IKE phase two, this specifies the Diffie-Hellman group to use. The DiffieHellmanGroup class is described in section 10.
DEFAULT	Since an IPsecAction object MAY optionally contain a IPsecGroup object, absence of one when using PFS indicates that the IKE phase two negotiation should use the same Diffie-Hellman group that was agreed upon during the IKE phase one negotiation.

8.4. The Class IPsecTransportAction

IPsecTransportAction is a specialization of IPsecAction, but does not add any attributes. It is used to signify that the phase two action will be for the negotiation of an IPsec transport mode SA.

8.5. The Class IPsecTunnelAction

IPsecTunnelAction is a specialization of IPsecAction that is used to signify that the phase two action will be for the negotiation of an IPsec tunnel mode SA. It contains the following reference:

NAME	RemoteGateway
DESCRIPTION	The identity of the point where the tunnel terminates on the remote gateway.

Note: since a particular IPsec policy is directly associated with a particular interface in the system, the local gateway identity can be implicitly determined from this information.

8.6. The Class IPsecBypassAction

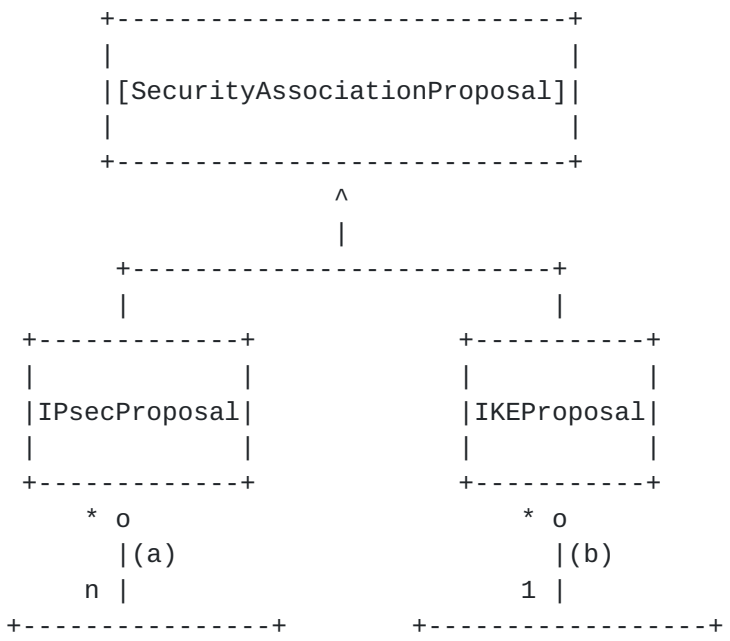
IPsecBypassAction is a specialization of IPsecAction, but does not add any attributes. It is used to signify that the traffic is to be allowed to pass in the clear.

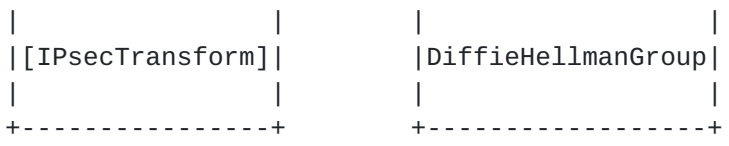
8.6. The Class IPsecDiscardAction

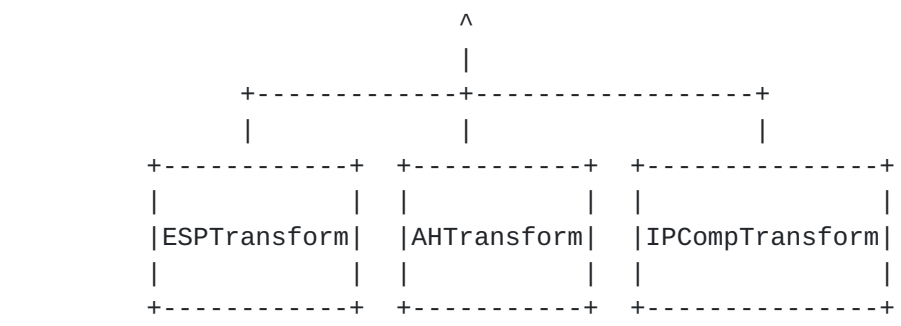
IPsecDiscardAction is a specialization of IPsecAction, but does not add any attributes. It is used to signify that the traffic should be denied.

9. IKE and IPsec Proposal Classes

A proposal contains the security parameters that will be used during the IKE phase one and two negotiations.




[IPsecTransform]
DiffieHellmanGroup



(a) Transforms
(b) IkeDhGroup

9.1. The Class SecurityAssociationProposal

The SecurityAssociationProposal class contains the parameters that are common between the IKE and IPsec proposal classes. It contains the following attributes:

NAME	lifetimeSeconds
DESCRIPTION	Specifies the seconds lifetime for this particular proposal. This value is used when sending this proposal to the negotiating peer. Additionally, it may be used, possibly in conjunction with the minimum seconds lifetime value, when selecting a proposal from the negotiating peer.
TYPE	unsigned 32-bit integer
VALUE	0 - indicates that the lifetime value defaults to 8 hours (28,800 seconds).

NAME	lifetimeKilobytes
DESCRIPTION	Specifies the kilobyte lifetime for this particular proposal. This value is used when sending this proposal to the negotiating peer. Additionally, it may be used, possibly in conjunction with the minimum kilobyte lifetime value, when selecting a proposal from the negotiating peer.
TYPE	unsigned 32-bit integer
VALUE	0 - indicates that there is no kilobyte lifetime.

9.2. The Class IKEProposal

IKEProposal is a specialization of the SecurityAssociationProposal class and specifies the parameters unique to the IKE proposal. It contains the following attributes/references:

NAME	cipherAlgorithm
DESCRIPTION	Specifies the encryption algorithm the IKE server will propose.

```
   TYPE         unsigned 16-bit integer
   VALUE        1 - DES-CBC
                2 - IDEA-CBC
                3 - Blowfish-CBC
```

```
                4 - RC5-R16-B64-CBC
                5 - 3DES-CBC
                6 - CAST-CBC

   NAME         hashAlgorithm
   DESCRIPTION  Specifies the hash algorithm the IKE server will
                propose.
   TYPE         unsigned 16-bit integer
   VALUE        1 - MD5
                2 - SHA-1
                3 - Tiger

   NAME         authenticationMethod
   DESCRIPTION  Specifies the authentication method the IKE server will
                propose.
   TYPE         unsigned 16-bit integer
   VALUE        1 - Preshared Key
                2 - DSS Signatures
                3 - RSA Signatures
                4 - RSA Encryption
                5 - Revised RSA Encryption
                6 - El-Gamal Encryption
                7 - Revised El-Gamal Encyrption
                65001 - Kerberos

   NAME         lifetimeDerivedKeys
   DESCRIPTION  Specifies the number of times the IKE phase one key may
                be used to derive an IKE phase two key.
   TYPE         unsigned 32-bit integer
   VALUE        0 - indicates that the number of times a IKE phase one
                key may be used to derive an IKE phase two key is
                limited by the seconds and/or kilobyte lifetimes.

   lifetimeDerivedKeys is not a negotiated parameter.  Although this
   value is not negotiated, it is included with the proposal
   information as the value is dependent upon the strength of the
   security parameters in the proposal.

   NAME         prfAlgorithm
   DESCRIPTION  Specifies the Psuedo-Random Function (PRF) the IKE
                server will propose.
   TYPE         unsigned 16-bit integer
   VALUE        At this time, there are no negotiable PRFs defined.

   NAME         IkeDhGroup
   DESCRIPTION  Specifies the Diffie-Hellman group that the IKE server
                will propose.  The DiffieHellmanGroup class is defined
                in section 10.
```

9.3. The Class IPsecProposal

IPsecProposal is a specialization of the SecurityAssociationProposal class and specifies the parameters unique to the IPsec proposal. It contains the following reference:

NAME	Transforms
DESCRIPTION	Specifies a set of IPsecTransform objects that represent the Encapsulating Security Payload (ESP), Authentication Header (AH), and IP Payload Compression Protocol (IPComp) parameters that are to used to create an IPsec proposal. Transforms of the same type are to be grouped together and logically ORed and the order of the transforms of the same type MUST be preserved. The transform groups are to be logically ANDed. For example, if the proposal had the following set of transforms {ESP=DES,AH=MD5,ESP=3-DES,ESP=RC5,AH=SHA-1}, the proposal would be ((ESP = DES or 3-DES or RC5) and (AH = MD5 or SHA-1)). An IPsecProposal object MAY reference one to many IPsecTransform objects. An IPsecTransform object MAY be referenced by zero to many IPsecProposal objects.

9.4. The Class IPsecTransform

The IPsecTransform class contains no properties and exists only for the purpose of modeling the is-a-kind-of relationship for IPsec transforms. For example, an ESPTransform is a kind of IPsecTransform.

9.5. The Class ESPTransform

ESPTransform is a specialization of an IPsecTransform. It specifies the parameters for one IPSec ESP transform within an IPsec proposal. It contains the following attributes:

NAME	integrityTransformId
DESCRIPTION	Specifies the ESP integrity algorithm to propose.
TYPE	unsigned 16-bit integer
VALUE	1 - HMAC MD5 2 - HMAC SHA-1 3 - HMAC DES 4 - KPDK

NAME	cipherTransformId
DESCRIPTION	Specifies the ESP cipher/encryption algorithm to propose.
TYPE	unsigned 16-bit integer
VALUE	1 - DES IV64 2 - DES

3 - 3-DES
4 - RC5
5 - IDEA
6 - CAST
7 - Blowfish

```
                8 - 3-IDEA
                9 - DES IV32
                10 - RC4
                11 - NULL

   NAME         cipherKeyRounds
   DESCRIPTION  Specifies the number of key rounds for the ESP cipher
                algorithm specified by the attribute cipherTransformId.
   TYPE         unsigned 16-bit integer
   VALUE        At this time, there are no cipher key rounds defined
                for any IPsec ESP algorithms.

   NAME         cipherKeyLength
   DESCRIPTION  Specifies the length of the ESP cipher key, in bits.
                If cipherTansformId specifies a cipher with a fixed-
                length key, cipherKeyLength is ignored.
   TYPE         unsigned 16-bit integer
   VALUE        0 - the cipher algorithm specified by the
                cipherTransformId attribute implies the key length.
                Any other value specifies a key length, in bits.
```

9.6. The Class AHTransform

AHTransform is a specialization of an IPsecTransform. It specifies the parameters for one AH transform within an IPsec proposal. It contains the following property:

```
   NAME         transformId
   DESCRIPTION  Specifies the AH hash algorithm to propose.
   TYPE         unsigned 16-bit integer
   VALUE        2 - MD5
                3 - SHA-1
                4 - DES
```

9.7. The Class IPCompTransform

IPCompTransform is a specialization of an IPsecTransform. It specifies the parameters for one IPComp transform within an IPsec proposal. It contains the following properties:

```
   NAME         algorithm
   DESCRIPTION  Specifies the IPComp compression algorithm to propose.
   TYPE         unsigned 16-bit integer
   VALUE        1 - OUI (privateAlgorithm MUST contain a valid value)
                2 - Deflate
                3 - LZS

   NAME         dictionarySize
   DESCRIPTION  Specifies the dictionary size for the compression
```

algorithm.

TYPE unsigned 16-bit integer

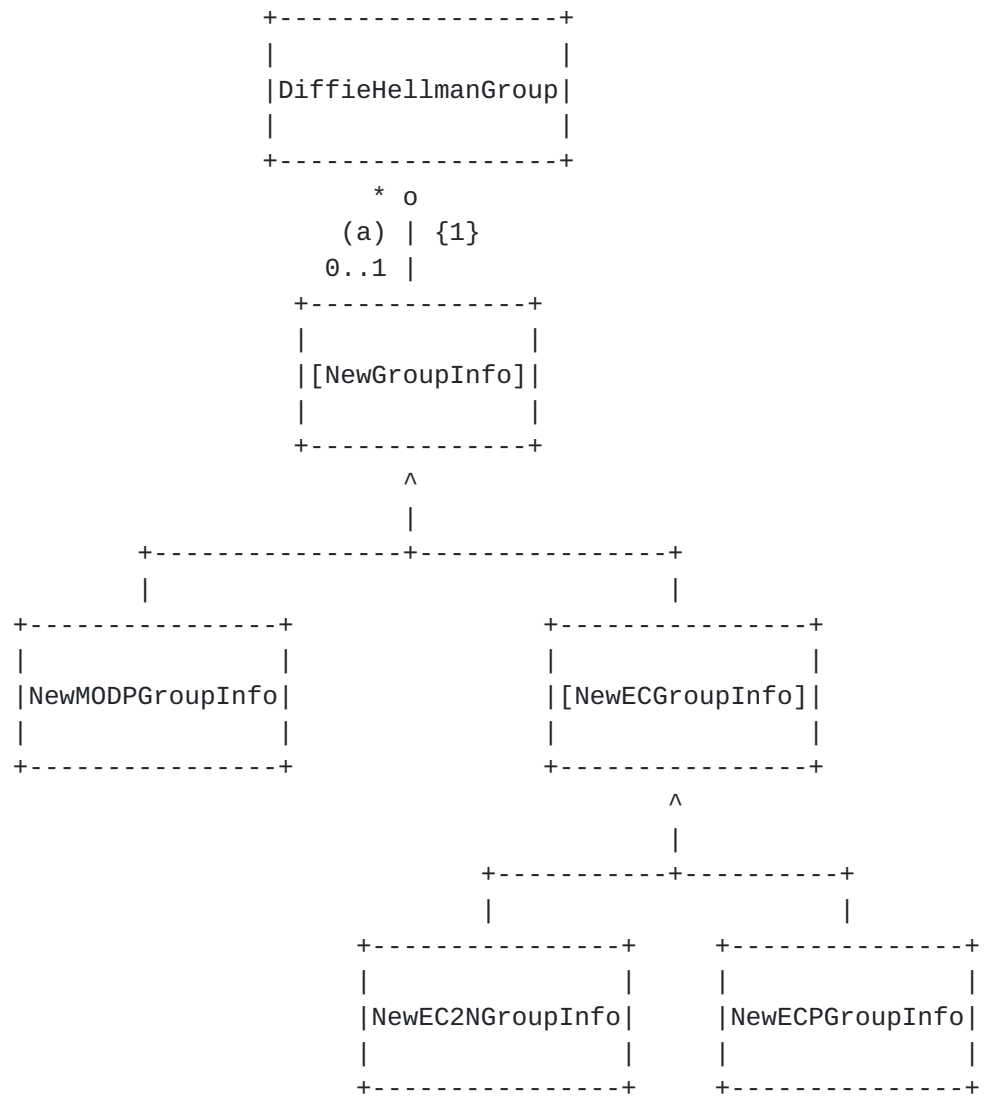
VALUE 0 - the compression algorithm specified by the algorithm attribute dictates the dictionary size.

Any other value is to be interpreted in the context of the compression algorithm.

NAME	privateAlgorithm
DESCRIPTION	If the algorithm attribute specifies the use of a proprietary compression transform (OUI = 1), then this specifies the specific vendor algorithm that will be used. Otherwise, this value is ignored.
TYPE	unsigned 32-bit integer

10. Diffie-Hellman Classes

The Diffie-Hellman classes are used to define the Diffie-Hellman attributes that are used during phase one (and possibly phase two) of the IKE negotiation.



(a) ExplicitGroupInfo

{1} If the Diffie-Hellman Group is a well-known group (or previously agreed upon private group), then the NewGroupInfo object doesn't exist (or is ignored).

10.1. The Class DiffieHellmanGroup

DiffieHellmanGroup describes the specific Diffie-Hellman Group that will be proposed. It contains the following properties:

NAME	groupDescription
DESCRIPTION	Specifies the Diffie-Hellman Group to propose.
TYPE	unsigned 16-bit integer
VALUE	1 - 768-bit MODP group 2 - 1024-bit MODP group 3 - EC2N group on GP[2^155] 4 - EC2N group on GP[2^185] 5 - 1536-bit MODP group

NAME	ExplicitGroupInfo
DESCRIPTION	Specifies Diffie-Hellman Group information if groupDescription is not one of the well-known values or a previously agreed upon private group. If groupDescription is one of the well-known values or a previously agreed upon private group, the NewGroupInfo object will not exist or it is ignored.

10.2. The Class NewGroupInfo

NewGroupInfo is the abstract base class for the concrete new group information classes. The specific derived class implies the group type value.

10.3. The Class NewMODPGroupInfo

NewMODPGroupInfo specifies the Diffie-Hellman group information for a MODP group that is proposed during new group mode. It contains the following properties:

NAME	prime
DESCRIPTION	Specifies the prime for the MODP group.
TYPE	byte string

NAME	generator
DESCRIPTION	Specifies the generator for the MODP group.
TYPE	byte string

10.4. The Class NewECGroupInfo

NewECGroupInfo is an abstract class that specifies the Diffie-Hellman group information for an elliptic curve group that is proposed during new group mode. It contains the following properties:

```
   NAME         polynomial
   DESCRIPTION  Specifies the polynomial for the elliptic curve group.
   TYPE         byte string
```

```
NAME         fieldSize
DESCRIPTION  Specifies the field size for the elliptic curve group.
TYPE         unsigned 32-bit integer

NAME         order
DESCRIPTION  Specifies the order for the elliptic curve group.
TYPE         unsigned 32-bit integer

NAME         generatorOne
DESCRIPTION  Specifies generator one for the elliptic curve group.
TYPE         byte string

NAME         generatorTwo
DESCRIPTION  Specifies generator two for the elliptic curve group.
TYPE         byte string

NAME         curveA
DESCRIPTION  Specifies curve A for the elliptic curve group.
TYPE         byte string

NAME         curveB
DESCRIPTION  Specifies curve B for the elliptic curve group.
TYPE         byte string
```

10.5. The Class NewEC2NGroupInfo

NewEC2NGroupInfo is a class that represents a new EC2N group. It contains no properties and exists only to imply the group type.

10.6. The Class NewECPGroupInfo

NewECPGroupInfo is a class that represents a new ECP group. It contains no properties and exists only to imply the group type.

11. Security Considerations

This document describes a schema for IPsec policy. It does not detail security requirements for storage or delivery of said schema. Storage and delivery security requirements should be detailed in a comprehensive security policy architecture document.

12. Intellectual Property

The IETF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it

has made any effort to identify any such rights. Information on the IETF's procedures with respect to rights in standards-track and standards-related documentation can be found in BCP-11.

Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

13. Acknowledgments

The author would like to thank Mike Jeronimo, Ylian Saint-Hilaire, Vic Lortz, and William Dixon for their contributions to this IPsec policy model.

Additionally, this draft would not have been possible without the preceding IPsec schema drafts. For that, thanks go out to Rob Adams, Partha Bhattacharya, William Dixon, Roy Pereira, and Raju Rajan.

14. References

[1] Harkins, D., and D. Carrel, "The Internet Key Exchange (IKE)", RFC 2409, November 1998.

[2] Shacham, A., and R. Monsour, R. Pereira, M. Thomas, "IP Payload Compression Protocol (IPComp)", RFC 2393, August 1998.

[3] Kent, S., and R. Atkinson, "IP Encapsulating Security Payload (ESP)", RFC 2406, November 1998.

[4] Kent, S., and R. Atkinson, "IP Authentication Header", RFC 2402, November 1998.

[5] Piper, D., "The Internet IP Security Domain of Interpretation for ISAKMP", RFC 2407, November 1998.

[6] Wahl, M., and T. Howes, S. Kille, "Lightweight Directory Access Protocol (v3)", RFC 2251, December 1997.

[7] Boyle, J., and R. Cohen, D. Durham, S. Herzog, R. Rajan, A. Sastry, "The COPS (Common Open Policy Service) Protocol", RFC 2748, January 2000. Internet-Draft work in progress.

[8] Condell, M., and C. Lynn, J. Zao, "Security Policy Specification Language", draft-ietf-ipsec-spsl-01.txt, July 1999. Internet-Draft

work in progress.

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

15. Disclaimer

The views and specification herein are those of the authors and are not necessarily those of their employer. The authors and their employer specifically disclaim responsibility for any problems arising from correct or incorrect implementation or use of this specification.

16. Author's Address

Jamie Jason
Intel Corporation
MS JF3-206
2111 NE 25th Ave.
Hillsboro, OR 97124
Phone: +1-503-264-9531
Fax: +1-503-264-9428
E-Mail: jamie.jason@intel.com

17. Full Copyright Statement