Internet Draft
Expiration: Apr. 1999
File: draft-ietf-rap-rsvp-ext-01.txt

Shai Herzog
IPHighway

RSVP Extensions for Policy Control

November 18, 1998

Status of this Memo

Abstract

This memo presents a set of extensions for supporting generic policy based admission control in RSVP. It should be perceived as an extension to the RSVP functional specifications [RSVPSP]

These extensions include the standard format of POLICY_DATA objects, and a description of RSVP's handling of policy events.

This document does not advocate particular policy control mechanisms; however, a Router/Server Policy Protocol description for these extensions can be found in [Fwk, COPS, COPS-RSVP].

Table of Contents

1. Introduction

RSVP, by definition, discriminates between users, by providing some users with better service at the expense of others. Therefore, it is reasonable to expect that RSVP be accompanied by mechanisms for controlling and enforcing access and usage policies. Historically, when RSVP Ver. 1 was developed, the knowledge and understanding of policy issues was in its infancy. As a result, Ver. 1 of the RSVP Functional Specifications [RSVPSP] left a place holder for policy support in the form of POLICY_DATA objects. However, it deliberately refrained from specifying mechanisms, message formats, or providing insight into how policy enforcement should be carried out. This document is intended to fill in this void.

The current RSVP Functional Specification describes the interface to admission (traffic) control that is based "only" on resource availability. In this document we describe a set of extensions to RSVP for supporting policy based admission control as well. The scope of this document is limited to these extensions and does not advocate specific architectures for policy based controls.

For the purpose of this document we define Local Policy Module (LPM) as the policy entity within the RSVP node. This may be fully contained within the RSVP node or may be using an outsourcing mechanism such as described in [Fwk, COPS, COPS-RSVP].

2. Policy Data Object Format

The following replaces section A.13 in [RSVPSP].

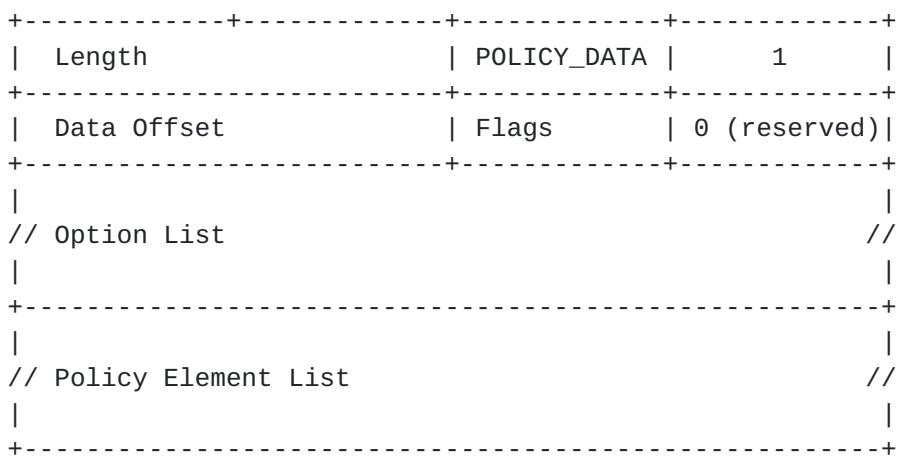
POLICY_DATA objects are carried by RSVP messages and contain policy information. All policy-capable nodes (at any location in the network) can generate, modify, or remove policy objects, even when senders or receivers do not provide, and may not even be aware of policy data objects.

The exchange of POLICY_DATA objects between policy-capable nodes along the data path, supports the generation of consistent end-to-end policies. Furthermore, such policies can be successfully deployed across multiple administrative domains when border nodes manipulate and translate POLICY_DATA objects according to established sets of bilateral agreements.

2.1. Base Format

POLICY_DATA class=14

o Type 1 POLICY_DATA object: Class=14, C-Type=1



Data Offset: 16 bits

The offset in bytes of the data portion (from the first byte of the object header).

Flags: 8 bits

0x01 PCF_Updt
A modified object, don't check against previous one. This is an optimization for systems that attempt to detect unchanged refreshes of POLICY_DATA objects

Reserved: 8 bits

Always 0.

Option List: Variable length

The list of options and their usage is defined in Section 2.2.

Policy Element List: Variable length

The contents of policy elements is opaque to RSVP. See more details in Section 2.3.

2.2. Options

This section describes a set of options that may appear as options in POLICY_DATA objects. All policy options appear as RSVP objects; some use their valid original format while others appear as NULL objects.

2.2.1. Native RSVP Options

The following objects retain the same format specified in [RSVPSP] however, they gain different semantics when used inside POLICY_DATA objects.

FILTER_SPEC object (list)

The set of senders associated with the POLICY_DATA object. If none is provided, the policy information is assumed to be associated with all the flows of the session.

This option is only useful for WF or SE reservation styles, where merged reservations may have originally been intended for different subsets of senders. It can also be used to prevent policy loops in a manner similar to the usage of RSVP s SCOPE object. Using this option may have significant impact on scaling and size of POLICY_DATA objects and therefore should be taken with care.

Originating RSVP_HOP

The RSVP_HOP object identifies the neighbor/peer policy-capable node that constructed the policy object. When policy is enforced at border nodes, peer policy nodes may be several RSVP hops away from each other and the originating RSVP_HOP is the basis for the mechanism that allows them to recognize each other and communicate safely and directly.

If no RSVP_HOP object is present, the policy data is implicitly assumed to have been constructed by the RSVP_HOP indicated in the RSVP message itself (i.e., the neighboring RSVP node is policy-capable).

Destination RSVP_HOP

A second RSVP_HOP object may follow the originating RSVP_HOP object. This second RSVP_HOP identifies the destination policy node. This is used to ensure the POLICY_DATA object is delivered to targeted policy nodes. It may be used to emulate unicast delivery in multicast Path messages. It may also help prevent using a policy object in other parts of the network (replay attack).

On the receiving side, a policy node should ignore any POLCY_DATA that includes a destination RSVP_HOP that doesn t match its own IP address.

INTEGRITY Object

The INTEGRITY object provides guarantees that the object was not compromised. It follows the rules from [MD5], and is calculated over the POLICY_DATA object, the SESSION object, and the message type field

(byte, padded with zero to 32 bit) as if they formed one continuous in-

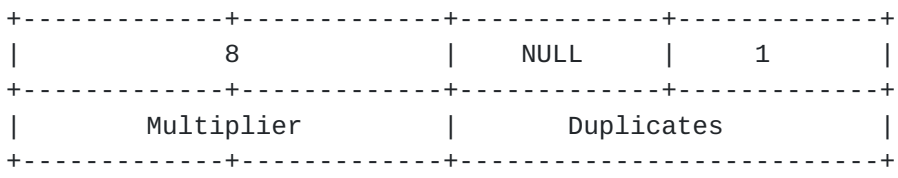
order message. This concatenation is designed to prevent copy and replay attacks of POLICY_DATA objects from other sessions, flows, message types or even other network locations.

2.2.2. Other Options

All options that do not use a valid RSVP object format, should use the NULL RSVP object format with different CType values. This document defines only one such option, however, several other may be considered in future versions. (e.g., Fragmentation, NoChange, etc.).

o Policy Refresh Multiplier

Some policies may have looser timing constraints than RSVP, and therefore may allow for lower refresh frequency. If the Policy Refresh Multiplier option is present, policy is refreshed only once in "Multiplier" RSVP refreshes, for "Duplicates" times.



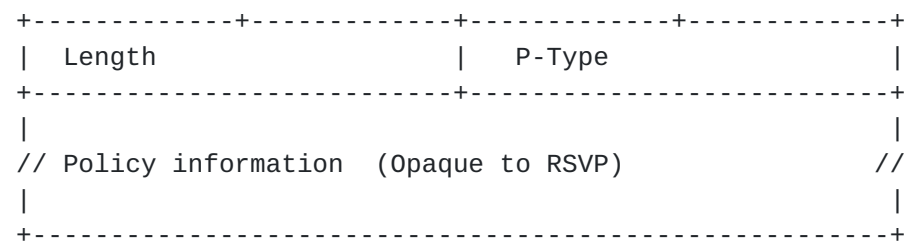
For example, for "Multiplier=16" and "Duplicates=3", the policy should be refreshed on RSVP's refreshes number 1,2,3,16,17,18,...

Note: this option s natural recovery time may be as long as Multiplier times the RSVP refresh period. Hence, it should only be used in conjunction with longer-term policies or topologies that can tolerate longer recovery time.

2.3. Policy Elements

The contents of policy elements is opaque to RSVP and its internal format is only known to the Local Policy Module (LPM). A list of policy elements code points (based on P-type) starting from 0, is registered with IANA. Local, Proprietary, and temporary P-Types can be used from the high end and down (2^16-1 and down).

Policy Elements have the following format:



3. Processing Rules

This sections describes the minimal required policy processing rules for RSVP.

3.1. Basic Signaling

It is generally agreed that policy control should only be enforced for Path, Resv, PathErr, and ResvErr. PathTear and ResvTear and assumed not to require policy control based on two assumptions: First, that MD-5 authentication verifies that the Tear is received from the same node that sent the initial reservation, and second, that it is functionally equivalent to that node holding-off refreshes for this reservation.

3.2. Error Signaling

Policy errors are reported by either ResvErr or PathErr messages with a policy failure error code (specified in [RSVPSP]). Policy error message must include a POLICY_DATA object; the object contains details of the error type and reason in a P-Type specific format.

If a multicast reservation fails due to policy reasons, RSVP should not attempt to discover which reservation caused the failure (as it would do for blockade state). Instead, it should attempt to deliver the policy ResvErr to ALL downstream hops, and have the LPM decide where messages should be sent. This mechanism allows the LPM to limit the error distribution by deciding which "culprit" next-hops should be informed. It also allows the LPM to prevent further distribution of ResvErr or PathErr messages by performing local repair (e.g. substituting the failed POLICY_DATA object with a different one).

3.3. Default Handling

It is generally assumed that policy enforcement (at least in its initial stages) is likely to concentrate on border nodes between autonomous systems. Consequently, policy objects transmitted at one edge of an autonomous cloud may traverse intermediate non-policy-capable RSVP nodes. The minimal requirement from a non-policy-capable RSVP node is to forward POLICY_DATA objects embedded in the appropriate

outgoing messages according to the following rules:

- POLICY_DATA objects are to be forwarded as is, without any modifications.

- Multicast merging (splitting) nodes:

  In the upstream direction:

  When multiple POLICY_DATA objects arrive from downstream, the RSVP node should concatenate all of them and forward them with the outgoing (upstream) message.

  On the downstream direction:

  When a single incoming POLICY_DATA object arrives from upstream, it should be forwarded (copied) to all downstream branches of the multicast tree.

The same rules apply to unrecognized policies (sub-objects) within the POLICY_DATA object. However, since this can only occur in a policy-capable node, it is the responsibility of the LPM and not RSVP.

4. References

[Fwk] R. Yavatkar, D. Pendarakis, R. Guerin. "A Framework for Policy Based Admission Control", Internet-Draft <draft-ietf-rap-framework-00.txt>, November, 1997.

[COPS] Boyle, J., Cohen, R., Durham, D., Herzog, S., Raja,n R., Sastry, A., "The COPS (Common Open Policy Service) Protocol", Internet-Draft <draft-ietf-rap-cops-02.txt>, Aug. 1998.

[RSVPSP] Braden, R., Zhang, L., Berson, S., Herzog, S., and Jamin, S., "Resource Reservation Protocol (RSVP) Version 1 Functional Specification", IETF RFC 2205, Proposed Standard, September 1997.

[MD5] F. Baker. RSVP Cryptographic Authentication" Internet-Draft, <draft-ietf-rsvp-md5-05.txt>, Aug. 1997.

5. Acknowledgments

This document incorporates inputs from Lou Berger, Bob Braden, Deborah Estrin, Roch Guerin, Timothy O'Malley, Dimitrios Pendarakis, Raju Rajan, Scott Shenker, Raj Yavatkar and many others.

6. Author Information

Shai Herzog, IPHighway
Parker Plaza, Suite 1500
400 Kelby St.
Fort-Lee, NJ 07024
(201) 585-0800
herzog@iphighway.com