RSVP Extensions for Policy Control

April 2, 1999

Status of this Memo

  This document is an Internet-Draft and is in full conformance with
  all provisions of Section 10 of RFC2026.

  Internet-Drafts are working documents of the Internet Engineering
  Task Force (IETF), its areas, and its working groups. Note that
  other groups may also distribute working documents as Internet-
  Drafts.

  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."

  The list of current Internet-Drafts can be accessed at
  http://www.ietf.org/ietf/1id-abstracts.txt

  The list of Internet-Draft Shadow Directories can be accessed at
  http://www.ietf.org/shadow.html.

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 [RSVP]

  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 [RAP, 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. Ver. 1 of the
RSVP Functional Specifications [RSVP] left a placeholder for policy
support in the form of POLICY_DATA object.

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 do not differentiate between
Policy Decision Point (PDP) and Local Decision Point (LDPs) as
described in [RAP]. The term PDP should be assumed to include LDP as
well.

2 Policy Data Objects

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.

The following extends section A.13 in [RSVP].

2.1 Base Format

POLICY_DATA class=14

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

```
+-------------+-------------+-------------+-------------+
|  Length                   | POLICY_DATA |      1      |
+---------------------------+-------------+-------------+
|  Data Offset              | 0 (reserved)              |
+---------------------------+-------------+-------------+
|                                                       |
// Option List                                         //
|                                                       |
+-------------------------------------------------------+
|                                                       |
// Policy Element List                                 //
|                                                       |
+-------------------------------------------------------+
```

Data Offset: 16 bits

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

Reserved: 16 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 in
POLICY_DATA objects. All policy options appear as RSVP objects but
their semantic is modified when used as policy data options.

FILTER_SPEC object (list) or SCOPE object

These objects describe 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. These

two types of objects are mutually exclusive, and cannot be mixed.

In Packed FF Resv messages, this FILTER_SPEC option provides
association between a reserved flow and its POLICY_DATA objects.

In WF or SE styles, this option preserves the original
flow/POLICY_DATA association as formed by PDPs, even across RSVP
capable PIN nodes. Such preservation is required since PIN nodes may
change the list of reserved flows on a per-hop basis, irrespective
of legitimate Edge-to-Edge PDP policy considerations.

Last, the SCOPE object should be used to prevent "policy loops" in a
manner similar to the one described in [RSVP], Section 3.4. When PIN
nodes are part of a WF reservation path, the RSVP SCOPE object is
unable to prevent policy loops and the separate policy SCOPE object
is required.

Note: using the SCOPE option may have significant impact on scaling
and size of POLICY_DATA objects.

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 POLICY_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.

Policy Refresh TIME_VALUES (PRT)

The Policy Refresh TIME_VALUES (PRT) option is used to slow policy
refresh frequency for policies that have looser timing constraints
relative to RSVP. If the PRT option is present, policy refreshes can
be withheld as long as at least one refresh is sent before the
policy refresh timer expires. A minimal value for PRT is R; lower
values are assumed to be R (neither error nor warning should be
triggered).

To simplify RSVP processing, time values are not based directly on
the PRT value, but on a Policy Refresh Multiplier N computed as
N=Floor(PRT/R). Refresh and cleanup rules are derived from [RSVP]
Section 3.7 assuming the refresh period for PRT POLICY DATA is R'
computed as R'=N*R. In effect, both the refresh and the state
cleanup are slowed by a factor of N).

The refresh multiplier applies to no-change periodic refreshes only
(rather than updates). For example, a policy being refreshed at time
T, T+N, T+2N,... may encounter a route change detected at T+X. In
this case, the event would force an immediate policy update and
would reset refresh times to T+X, T+X+N, T+X+2N,...

When network nodes restart, RSVP messages between PRT policy
refreshes may be rejected since they arrive without necessary
POLICY_DATA objects. This error situation would clear with the next
periodic policy refresh or with a policy update triggered by ResvErr
or PathErr messages.

This option is especially useful to combine strong (high overhead)
and weak (low overhead) authentication certificates as policy data.
In such schemes the weak certificate can support admitting a
reservation only for a limited time, after which the strong
certificate is required.

This approach may reduce the overhead of POLICY_DATA processing.
Strong certificates could be transmitted less frequently, while weak
certificates are included in every RSVP refresh.

2.3 Policy Elements

The content of policy elements is opaque to RSVP; their internal
format is understood by policy peers e.g. an RSVP Local Decision
Point (LDP) or a Policy Decision Point (PDP) [RAP]. A registry of

policy element codepoints and their meaning is maintained by [IANA-CONSIDERATIONS] (also see Section 4).

Policy Elements have the following format:

```
+-------------+-------------+-------------+-------------+
|  Length                   |      P-Type               |
+---------------------------+---------------------------+
|                                                       |
// Policy information  (Opaque to RSVP)               //
|                                                       |
+-------------------------------------------------------+
```

2.4 Purging Policy State

Policy state expires in the granularity of Policy Elements (POLICY_DATA objects are mere containers and do not expire as such).

Policy elements expire in the exact manner and time as the RSVP state received in the same message (see [RSVP] Section 3.7). PRT controlled state expires N times slower (see Section 2.2).

Only one policy element of a certain P-Type can be active at any given time. Therefore, policy elements are instantaneously replaced when another policy element of the same P-Type is received from the same PDP (previous or next policy RSVP_HOP). An empty policy element of a certain P-Type is used to delete (rather than a replace) all policy state of the same P-Type.

3 Processing Rules

These sections describe the minimal required policy processing rules for RSVP.

3.1 Basic Signaling

This draft mandates enforcing policy control for Path, Resv, PathErr, and ResvErr messages only. PathTear and ResvTear are assumed not to require policy control based on two main presumptions. First, that Integrity verification [MD5] guarantee that the Tear is received from the same node that sent the installed reservation, and second, that it is functionally equivalent to that node holding-off refreshes for this reservation.

3.2 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 policy

ignorant RSVP nodes (PINs). A PIN is required at a minimum to

forward the received POLICY_DATA objects in the appropriate outgoing
messages according to the following rules:

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

o   Multicast merging (splitting) nodes:

     In the upstream direction:

         When multiple POLICY_DATA objects arrive from downstream, the
         RSVP node should concatenate all of them (as a list of the
         original POLICY_DATA objects) 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 PDP and not
RSVP.

3.3 Error Signaling

Policy errors are reported by either ResvErr or PathErr messages
with a policy failure error code in the ERROR_SPEC object. 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
(See Section 2.3).

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 PDP (or LDP)
decide where messages should be sent. This mechanism allows the PDP
to limit the error distribution by deciding which "culprit" next-
hops should be informed. It also allows the PDP 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).

Error codes are described in Appendix A.

4   IANA Considerations

RSVP Policy Elements (P-Types)

Following the policies outlined in [IANA-CONSIDERATIONS],numbers 0-
49151 are allocated as standard policy elements by IETF Consensus
action, numbers in the range 49152-53247 are allocated as vendor
specific (one per vendor) by First Come First Serve, and numbers
53248-65535 are reserved for private use and are not assigned by
IANA.

5   Security Considerations

This draft describes the use of POLICY_DATA objects to carry policy-
related information between RSVP nodes. Two security mechanisms can
be optionally used to ensure the integrity of the carried
information. The first mechanism relies on RSVP integrity [MD5] to
provide a chain of trust when all RSVP nodes are policy capable. The
second mechanism relies on the INTEGRITY object within the
POLICY_DATA object to guarantee integrity between non-neighboring
RSVP PEPs (see Section 2.2).

6 References

[RAP]   Yavatkar, R., et al., "A Framework for Policy Based Admission
         Control",IETF <draft-ietf-rap-framework-02.txt>, Jan., 1999.

[COPS] Boyle, J., Cohen, R., Durham, D., Herzog, S., Raja,n R.,
         Sastry, A., "The COPS (Common Open Policy Service) Protocol",
         IETF <draft-ietf-rap-cops-05.txt>, Jan. 1999.

[COPS-RSVP] Boyle, J., Cohen, R., Durham, D., Herzog, S., Raja,n R.,
         Sastry, A., "COPS Usage for RSVP", IETF <draft-ietf-rap-cops-
         rsvp-04.txt>, Feb. 1999.

[RSVP] Braden, R. ed., "Resource ReSerVation Protocol (RSVP) -
         Functional Specification.", IETF RFC 2205, Proposed Standard,
         Sep. 1997.

[MD5]   Baker, F., Lindell B., Talwar, M. "RSVP Cryptographic
         Authentication" Internet-Draft, <draft-ietf-rsvp-md5-07.txt>,
         Nov. 1998.

[IANA-CONSIDERATIONS] Alvestrand, H. and T. Narten, "Guidelines for
         Writing an IANA Considerations Section in RFCs", RFC 2434,
         October 1998.

7 Acknowledgments

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

8 Author Information

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

Appendix A: Policy Error Codes

  This Appendix extends the list of error codes described in Appendix
  B of [RSVP].

  Note that Policy Element specific errors are reported as described
  in Section 3.3 and cannot be reported through RSVP (using this
  mechanism). However, this mechanism provides a simple, less secure
  mechanism for reporting generic policy errors. Most likely the two
  would be used in concert such that a generic error code is provided
  by RSVP, while Policy Element specific errors are encapsulated in a
  return POLICY_DATA object (as in Section 3.3).

  ERROR_SPEC class = 6

  Error Code = 02: Policy Control failure

  Error Value: 16 bit

  0 = ERR_INFO   : Information reporting
  1 = ERR_WARN   : Warning
  2 = ERR_UNKNOWN : Reason unknown
  3 = ERR_REJECT : Generic Policy Rejection
  4 = ERR_EXCEED : Quota or Accounting violation
  5 = ERR_PREEMPT : Flow was preempted
  6 = ERR_EXPIRED : Previously installed policy expired (not
  refreshed)
  7 = ERR_REPLACED: Previous policy data was replaced & caused
  rejection
  8 = ERR_MERGE   : Policies could not be merged (multicast)
  9 = ERR_PDP   : PDP down or non functioning
  10= ERR_SERVER : Third Party Server (e.g., Kerberos) unavailable
  11= ERR_PD_SYNTX: POLICY_DATA object has bad syntax
  12= ERR_PD_INTGR: POLICY_DATA object failed Integrity Check
  13= ERR_PE_BAD : POLICY_ELEMENT object has bad syntax
  14= ERR_PD_MISS : Mandatory PE Missing (Empty PE is in the PD
  object)
  15= ERR_NO_RSC : PEP Out of resources to handle policies.
  16= ERR_RSVP   : PDP encountered bad RSVP objects or syntax
  17= ERR_SERVICE : Service type was rejected
  18= ERR_STYLE   : Reservation Style was rejected
  19= ERR_FL_SPEC : FlowSpec was rejected (too large)

  Values between 2^15 and 2^16-1 can be used for site and/or vendor
  error values.