

TSVWG
Internet-Draft
Intended status: Informational
Expires: August 28, 2007

F. Le Faucheur
Cisco
J. Manner
University of Helsinki
D. Wing
Cisco
A. Guillou
Neuf
February 24, 2007

RSVP Proxy Approaches
draft-ietf-tsvwg-rsvp-proxy-approaches-00.txt

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with [Section 6 of BCP 79](#).

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>.

This Internet-Draft will expire on August 28, 2007.

Copyright Notice

Copyright (C) The IETF Trust (2007).

Abstract

RSVP signaling can be used to make end-to-end resource reservations in an IP network in order to guarantee the QoS required by certain flows. With conventional RSVP, both the data sender and receiver of

a given flow take part in RSVP signaling. Yet, there are many use cases where resource reservation is required, but the receiver, the sender, or both, is not RSVP-capable. This document presents RSVP Proxy behaviors allowing RSVP routers to perform RSVP signaling on behalf of a receiver or a sender that is not RSVP-capable. This allows resource reservations to be established on parts of the end-to-end path. This document reviews conceptual approaches for deploying RSVP Proxies and discusses how those can synchronize RSVP reservations with application requirements. This document also points out where extensions to RSVP (or to other protocols) may be needed for deployment of a given RSVP Proxy approach. However, such extensions are outside the scope of this document. Finally, practical use cases for RSVP Proxy are described.

Table of Contents

1.	Introduction	3
2.	RSVP Proxy Behaviors	4
2.1.	RSVP Receiver Proxy	4
2.2.	RSVP Sender Proxy	5
3.	Terminology	6
4.	RSVP Proxy Approaches	7
4.1.	Path-Triggered Receiver Proxy	7
4.2.	Path-Triggered Sender Proxy for Reverse Direction	10
4.3.	Inspection-Triggered Proxy	13
4.4.	STUN-Triggered Proxy	15
4.5.	Application-Signaling-Triggered On-Path Proxy	17
4.6.	Application-Signaling-Triggered Off-Path Source Proxy	21
4.7.	RSVP-Signaling-Triggered Proxy	23
4.8.	Other Approaches	24
5.	Security Considerations	24
6.	IANA Considerations	25
7.	Acknowledgments	25
8.	Informative References	25
Appendix A.	Use Cases for RSVP Proxies	27
A.1.	RSVP-based VoD CAC in Broadband Aggregation Networks	27
A.2.	RSVP-based Voice/Video CAC in Enterprise WAN	31
A.3.	RSVP-based Voice CAC in TSP Domain	33
A.4.	RSVP Proxies for Mobile Access Networks	34
	Authors' Addresses	35
	Intellectual Property and Copyright Statements	37

1. Introduction

Guaranteed QoS for some applications with tight QoS requirements may be achieved by reserving resources in each node on the end-to-end path. The main IETF protocol for these resource reservations is RSVP, specified in [[RFC2205](#)]. RSVP does not require that all intermediate nodes support RSVP, however it assumes that both the sender and the receiver of the data flow support RSVP. There are environments where it would be useful to be able to reserve resources for a flow on at least a subset of the flow path even when the sender or the receiver (or both) is not RSVP capable.

Since the data sender or receiver may be unaware of RSVP, there are two scenarios. In the first case, an entity in the network must operate on behalf of the data sender, generate an RSVP Path message, and eventually receive, process and sink a Resv message. We refer to this entity as the RSVP Sender Proxy. In the latter case, an entity in the network must receive a Path message sent by a data sender (or by an RSVP Sender Proxy), and reply to it with a Resv message on behalf of the data receiver(s). We refer to this entity as the RSVP Receiver Proxy.

The flow sender and receiver generally have at least some (if not full) awareness of the application producing or consuming that flow. Hence, the sender and receiver are in a natural position to synchronize the establishment, maintenance and tear down of the RSVP reservation with the application requirements and to determine the characteristics of the reservation (bandwidth, QoS service,...) which best match the application requirements. For example, before completing the establishment of a multimedia session, the endpoints may decide to establish RSVP reservations for the corresponding flows. Similarly, when the multimedia session is torn down, the endpoints may decide to tear down the corresponding RSVP reservations. For example, [[RFC3312](#)] discusses how RSVP reservations can be very tightly synchronised by SIP endpoints with SIP session control and SIP signaling.

When RSVP reservation establishment, maintenance and tear down is to be handled by RSVP Proxy devices on behalf of an RSVP sender or receiver, a key challenge for the RSVP proxy is to determine when the RSVP reservations need to be established, maintained and torn down and to determine what are the characteristics (bandwidth, QoS Service,...) of the required RSVP reservations matching the application requirements. We refer to this problem as the synchronization of RSVP reservations with application level requirements.

The IETF Next Steps in Signaling (NSIS) working group is designing,

as one their many charter items, a new QoS signaling protocol. This scheme already includes the notion of proxy operation, and terminating QoS signaling on nodes that are not the actual data senders or receivers. This is the same concept as the proxy operation for RSVP discussed in this document. One difference though is that the NSIS framework does not consider multicast resource reservations, which RSVP provides today.

The next two sections introduce the notion of RSVP Sender Proxy and RSVP receiver Proxy. The following section defines useful terminology. The subsequent section then presents several fundamental RSVP Proxy approaches insisting on how they achieve the synchronization of RSVP reservations with application level requirements. [Appendix A](#) includes more detailed use cases for the proxies in various deployment environments.

2. RSVP Proxy Behaviors

This section discusses the two types of proxies; the RSVP Sender Proxy operating on behalf of data senders, and the RSVP Receiver Proxy operating for data receivers. The concepts presented in this document are not meant to replace the standard RSVP and end-to-end RSVP reservations are still expected to be used whenever possible. However, RSVP Proxies are intended to facilitate RSVP deployment where end-to-end RSVP signaling is not possible.

2.1. RSVP Receiver Proxy

With conventional RSVP operations, RSVP reservations are controlled by receivers of data. After a data sender has sent an RSVP Path message towards the intended recipient(s), each recipient that requires a reservation generates a Resv message. If, however, a data receiver is not running the RSVP protocol, the last hop RSVP router will still send the Path message to the data receiver, which will silently drop this message as an IP packet with an unknown protocol number.

In order for reservations to be made in such a scenario, one of the RSVP routers on the data path must somehow know that the data receiver will not be participating in the resource reservation signaling. This RSVP router should, thus, perform RSVP Receiver Proxy functionality on behalf of the data receiver. This is illustrated in the figure below. Various mechanisms by which the RSVP proxy router can gain the required information are discussed later in the document.

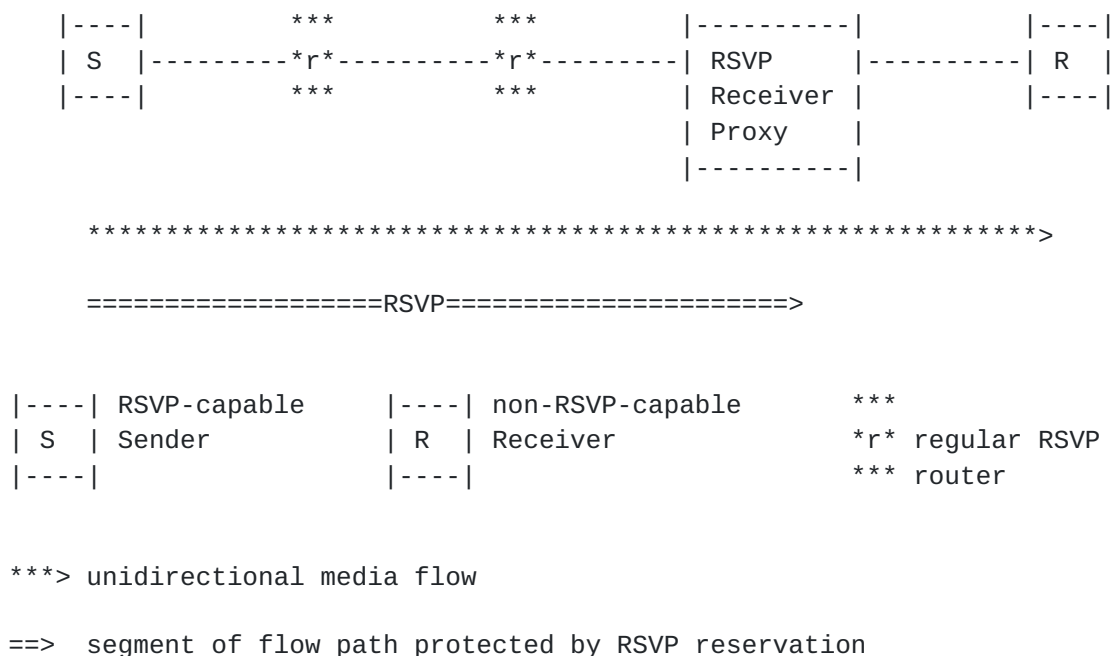


Figure 1: RSVP Receiver Proxy

2.2. RSVP Sender Proxy

With conventional RSVP operations, If a data sender is not running the RSVP protocol, a resource reservation can not be set up; a data receiver can not alone reserve resources without Path messages first being sent. Thus, even if the data receiver is running RSVP, it still needs some node on the data path to send a Path message towards the data receiver.

In that case, in a similar manner to the RSVP Receiver Proxy described before, an RSVP node on the data path must somehow know that it should generate a Path message for setting up a resource reservation. This case is more complex than the Receiver Proxy, since the RSVP Sender Proxy must be able to generate all the information in the Path message (such as the Sender TSpec) without the benefit of having previously seen any RSVP messages. An RSVP Receiver Proxy, by contrast only needs to formulate an appropriate Resv message in response to an incoming Path message. Mechanisms to operate an RSVP Sender Proxy are discussed later in this document.

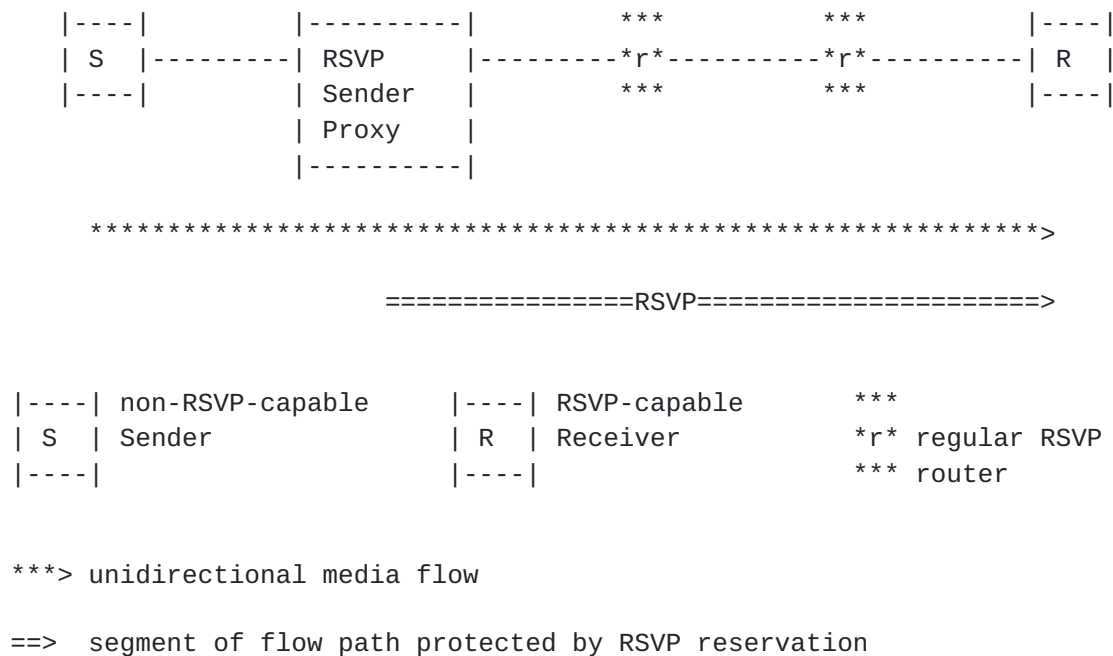


Figure 2: RSVP Sender Proxy

3. Terminology

On-Path: located on the datapath of the actual flow of data from the application (regardless of where it is located on the application level signaling path)

Off-Path: not On-Path

RSVP-capable (or RSVP-aware): which supports the RSVP protocol as per [\[RFC2205\]](#)

RSVP Receiver Proxy: an RSVP capable router performing, on behalf of a receiver, the RSVP operations which would normally be performed by an RSVP-capable receiver if end-to-end RSVP signaling was used. Note that while RSVP is used upstream of the RSVP Receiver Proxy, RSVP is not used downstream of the RSVP Receiver Proxy.

RSVP Sender Proxy: an RSVP capable router performing, on behalf of a sender, the RSVP operations which would normally be performed by an RSVP-capable sender if end-to-end RSVP signaling was used. Note that while RSVP is used downstream of the RSVP Sender Proxy, RSVP is not used upstream of the RSVP Sender Proxy.

Regular RSVP Router: an RSVP-capable router which is not behaving as

a RSVP Receiver Proxy nor as a RSVP Sender Proxy.

Note that the roles of RSVP Receiver Proxy, RSVP Sender Proxy, Regular RSVP Router are all relative to one unidirectional flow. A given router may act as the RSVP Receiver Proxy for a flow, as the RSVP Sender Proxy for another flow and as a Regular RSVP router for yet another flow.

Application level signaling: signaling between entities operating above the IP layer and which are aware of the QoS requirements for actual media flows. SIP and RTSP are examples of application level signaling protocol. RSVP is clearly not an application level signaling.

4. RSVP Proxy Approaches

This section specifies several fundamental RSVP Proxy approaches.

4.1. Path-Triggered Receiver Proxy

In this approach, it is assumed that the sender is RSVP capable and takes full care of the synchronisation between application requirements and RSVP reservations. With this approach, the RSVP Receiver Proxy uses the RSVP Path messages generated by the sender as the cue for establishing the RSVP reservation on behalf of the receiver. The RSVP Receiver Proxy is effectively acting as a slave making reservations (on behalf of the receiver) under the sender's control. This changes somewhat the usual RSVP reservation model where reservations are normally controlled by receivers. Such a change greatly facilitates operations in the scenario of interest here, which is where the receiver is not RSVP capable. Indeed it allows the RSVP Receiver Proxy to remain application unaware by taking advantage of the application awareness and RSVP awareness of the sender.

With the Path-Triggered RSVP Receiver Proxy approach, the RSVP router may be configured to use receipt of a regular RSVP Path message as the trigger for RSVP Receiver Proxy behavior.

On receipt of the RSVP Path message, the RSVP Receiver Proxy:

1. establishes the RSVP Path state as per regular RSVP processing
2. identifies the downstream interface towards the receiver
3. sinks the Path message

4. behaves as if a Resv message (whose details are discussed below) was received on the downstream interface. This includes performing admission control on the downstream interface, establishing a Resv state (in case of successful admission control) and forwarding the Resv message upstream, sending periodic refreshes of the Resv message and tearing down the reservation if the Path state is torn down.

In order to build the Resv message, the RSVP Receiver Proxy can take into account information received in the Path message. For example, the RSVP Receiver Proxy may compose a FLOWSPEC object for the Resv message which mirrors the SENDER_TSPEC object in the received Path message.

Operation of the Path-Triggered Receiver Proxy in the case of a successful reservation is illustrated in the Figure below.

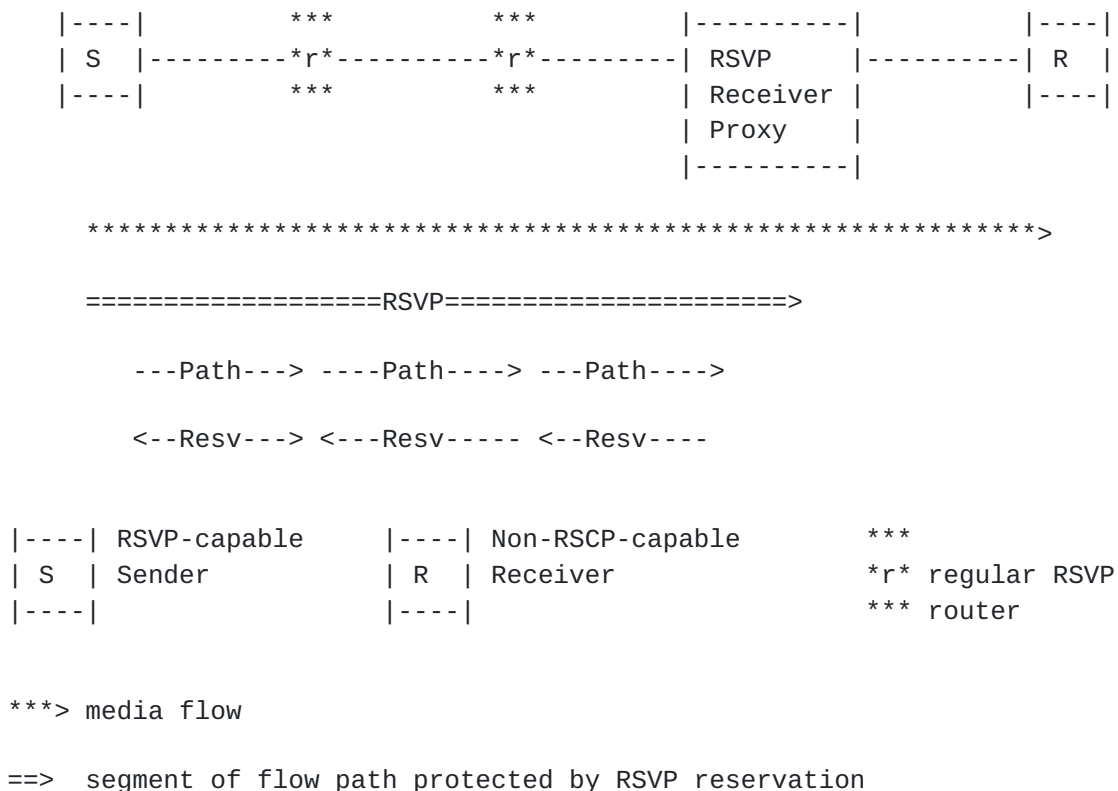
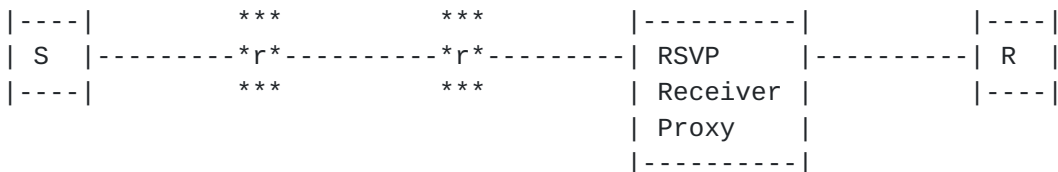


Figure 3: Path-Triggered RSVP Receiver Proxy

In case the reservation establishment is rejected (for example because of an admission control failure on a regular RSVP router on the path between the RSVP-capable sender and the RSVP Receiver

Proxy), a ResvErr message will be generated as per conventional RSVP operations and will travel downstream towards the RSVP Receiver Proxy. While this ensures that the RSVP Receiver Proxy is aware of the reservation failure, conventional RSVP procedures do not cater for notification of the sender of the reservation failure. Operation of the Path-Triggered RSVP Receiver Proxy in the case of an admission control failure is illustrated in the Figure below.



*****>

=====RSVP=====>

---Path---> ---Path---> ---Path--->

<---Resv----- <---Resv-----

---ResvErr---> --ResvErr--->

----	RSVP-capable	----	Non-RSCP-capable	***
S	Sender	R	Receiver	*r* regular RSVP
----		----		*** router

***> media flow

=> segment of flow path protected by RSVP reservation

Figure 4: Path-Triggered RSVP Receiver Proxy with Failure

Since, as explained above, in this scenario involving the RSVP Receiver Proxy, synchronisation between application and RSVP reservation is generally performed by the sender, notifying the sender of reservation failure is needed.

[[I-D.ietf-tsvwg-rsvp-proxy-proto](#)] specifies RSVP extensions allowing such sender notification in case of reservation failure.

4.2. Path-Triggered Sender Proxy for Reverse Direction

In this approach, it is assumed that one endpoint is RSVP capable and takes full care of the synchronisation between application requirements and RSVP reservations. This endpoint is the sender for one flow direction (which we refer to as the "forward" direction) and is the receiver for the flow in the opposite direction (which we refer to as the "reverse" direction).

With the Path-Triggered Sender Proxy for Reverse Direction approach, the RSVP Proxy uses the RSVP signaling generated by the sender as the cue for initiating RSVP signaling for the reservation in the reverse direction. Thus, the RSVP Proxy is effectively acting as a Sender Proxy for the reverse direction under the control of the sender for the forward direction. Note that this assumes a degree of symmetry for the two directions of the flow (as is currently typical for IP telephony, for example).

This is illustrated in the Figure below.

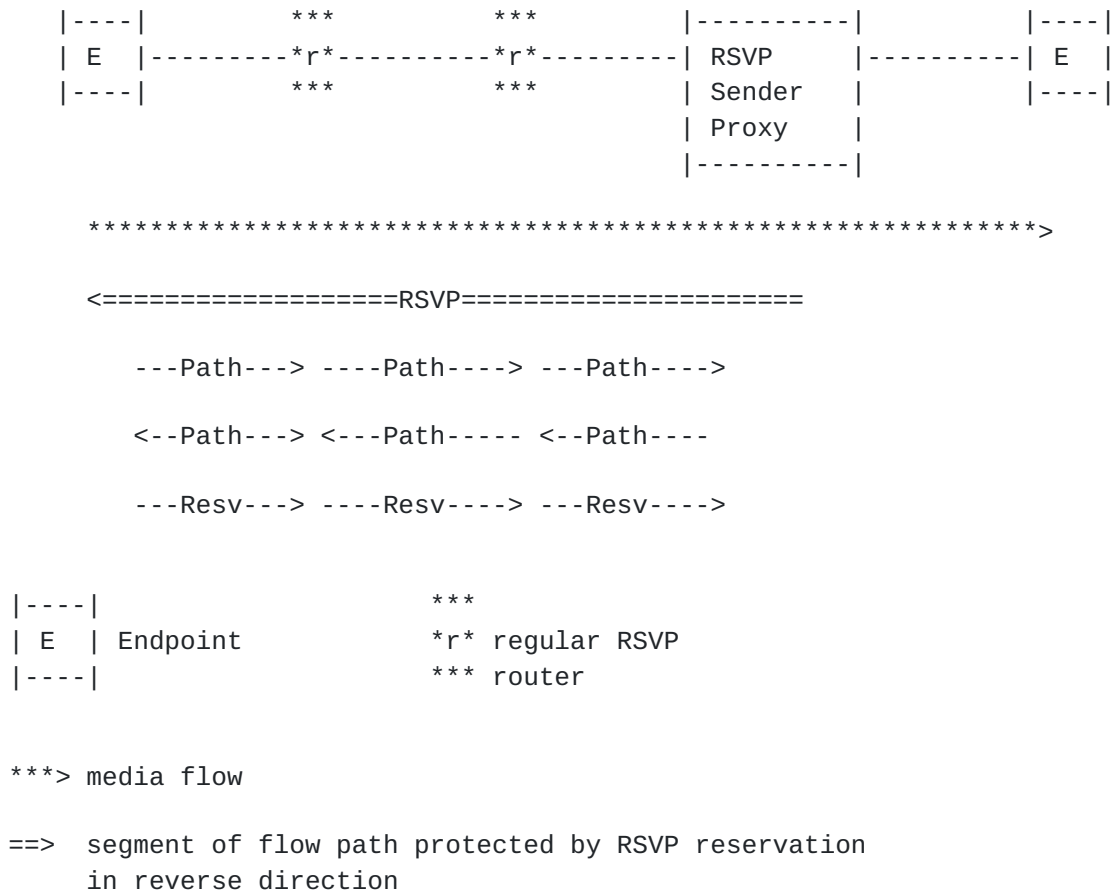


Figure 5: Path-Triggered Sender Proxy for Reverse Direction

Of course, the RSVP Proxy may simultaneously (and typically will) also act as the Path-Triggered Receiver Proxy for the forward direction, as defined in [Section 4.1](#). Such an approach is most useful in situations involving RSVP reservations in both directions for symmetric flows. This is illustrated in the Figure below

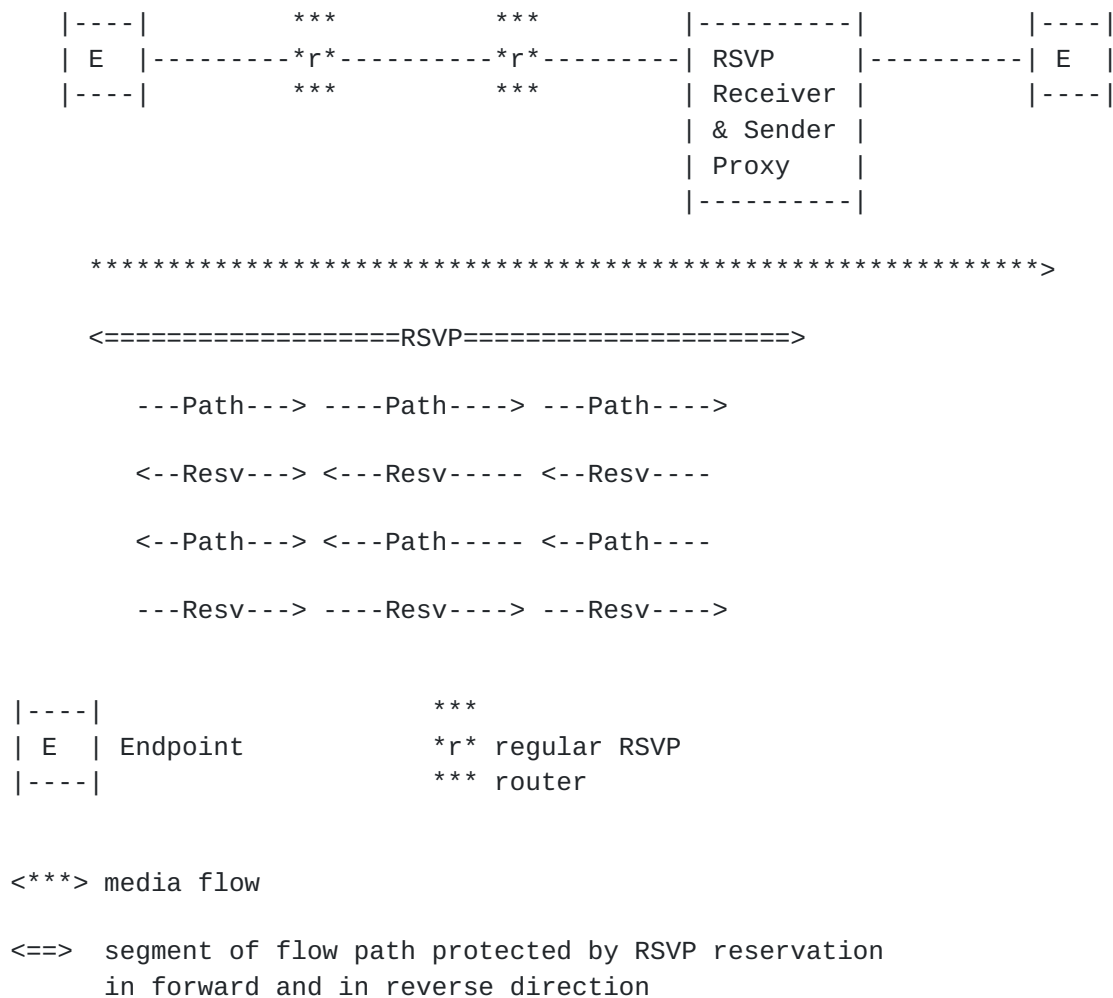


Figure 6: Path Triggered Receiver & Sender Proxy

With the Path-Triggered Sender Proxy for Reverse Direction approach, the RSVP router may be configurable to use receipt of a regular RSVP Path message as the trigger for Sender Proxy for Reverse Direction behavior.

On receipt of the RSVP Path message for the forward direction, the RSVP Sender Receiver Proxy :

1. sinks the Path message
2. behaves as if a Path message for reverse direction (whose details are discussed below) had been received by the Sender Proxy. This includes establishing the corresponding Path state, forwarding the Path message downstream, sending periodic refreshes of the Path message and tearing down the Path in reverse direction when

the Path state in forward direction is torn down.

In order to build the Path message for the reverse direction, the RSVP Sender Proxy can take into account information in the received Path message for the forward direction. For example, the RSVP Sender Proxy may mirror the SENDER_TSPEC object in the received Path message.

We observe that this approach does not require any extensions to the existing RSVP protocol.

4.3. Inspection-Triggered Proxy

In this approach, it is assumed that the RSVP Proxy device is on the datapath of "packets of interest", that it can inspect such packets on the fly as they transit through it, and that it can infer information from these packets of interest to determine what RSVP reservations need to be established, when and with what characteristics (possibly also using some configured information).

One example of "packets of interest" could be application level signaling. An RSVP Proxy device capable of inspecting SIP signaling for multimedia session or RTSP signaling for Video streaming, can obtain from such signaling information about when a multimedia session is up or when a Video is going to be streamed. It can also identify the addresses and ports of senders and receivers and can determine the bandwidth of the corresponding flows. Thus, such an RSVP Proxy device can determine all necessary information to synchronise RSVP reservations to application requirements. This is illustrated in the Figure below.

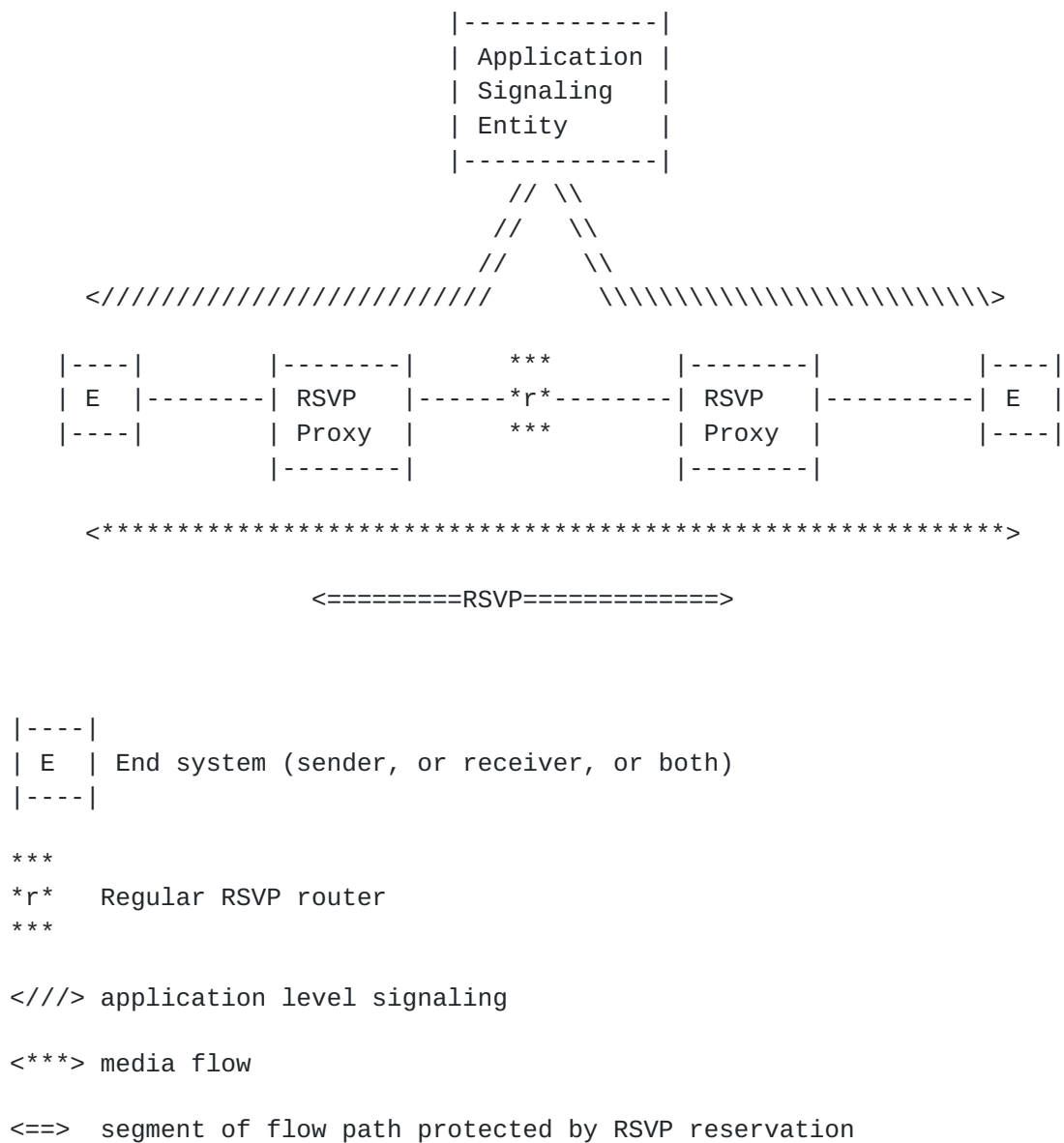


Figure 7: Inspection-Triggered RSVP Proxy

Another example of "packets of interest" could be packets belonging to the application flow itself (e.g. media packets). An RSVP Proxy device capable of detecting the transit of packets from a particular flow, can attempt to establish a reservation corresponding to that flow. Characteristics of the reservation may be derived from configuration, flow measurement or a combination of those.

Note however, that in case of reservation failure, the inspection-triggered RSVP Proxy does not have a direct mechanism for notifying the application (since it is not participating itself actively in

application signaling) so that the application takes appropriate action (for example terminate the corresponding session). To mitigate this problem, the inspection-triggered RSVP Proxy may mark differently the DSCP of flows for which an RSVP reservation has been successfully proxied from the flows for which a reservation is not in place. In some situations, the Inspection-Triggered Proxy might be able to modify the "packets of interest" (e.g. application signaling messages) to convey some hint to applications that the corresponding flows cannot be guaranteed by RSVP reservations.

With the inspection-triggered Proxy approach, the RSVP Receiver Proxy is effectively required to attempt to build application awareness by traffic inspection and then is somewhat limited in the actions it can take in case of reservation failure. However, this may be a useful approach in some environments. Note also that this approach does not require any change to the RSVP protocol.

With the "Inspection-Triggered" RSVP Proxy approach, the RSVP router may be configurable to use and interpret some specific "packets of interest" as the trigger for RSVP Receiver Proxy behavior.

4.4. STUN-Triggered Proxy

In this approach, the RSVP Proxy entity takes advantage of the application awareness provided by the STUN signaling to synchronise RSVP reservations with application requirements. The STUN signaling is sent from endpoint to endpoint. This is illustrated in the figure below.

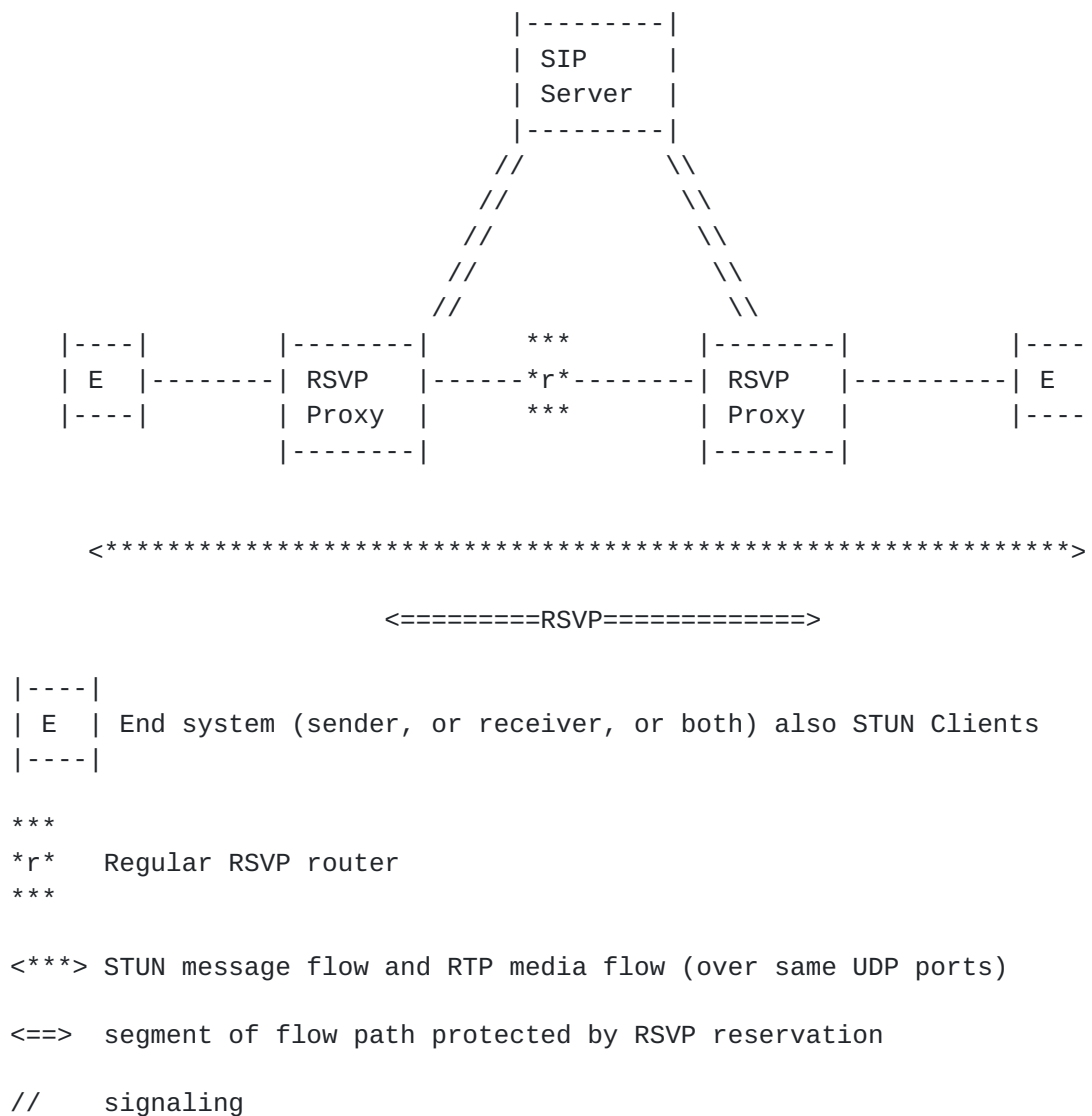


Figure 8: STUN-Triggered Proxy

In this approach, a STUN [[I-D.ietf-behave-rfc3489bis](#)] message triggers the RSVP proxy. Using a STUN message eases the RSVP proxy agent's computational burden because it need only look for STUN messages rather than maintain state of all flows. More importantly, the STUN message can also include a STUN attribute describing the bandwidth and describing the application requesting the flow, which allows the RSVP proxy agent to authorize an appropriately-sized reservation for each flow.

For unicast flows, [[I-D.ietf-mmusic-ice](#)] is an already widely-adopted emerging standard for NAT traversal. For our purposes, we are not interested in its NAT traversal capabilities. Rather, ICE's useful

characteristic is its connectivity check -- the exchange of STUN Binding Request messages between hosts to verify connectivity (see Connectivity Check Usage in [[I-D.ietf-behave-rfc3489bis](#)]). By including new STUN attributes in those messages an RSVP proxy agent could perform its functions more effectively. Additionally, the RSVP proxy agent can inform endpoints of an RSVP reservation failure by dropping the ICE connectivity check message. This provides very RSVP-like call admission control and signaling to the endpoints, without implementing RSVP on the endpoints.

For multicast flows (or certain kinds of unicast flows that don't or can't use ICE), a STUN Indication message [[I-D.ietf-behave-rfc3489bis](#)] could be used for similar purposes. The STUN Indication message is not acknowledged by its receiver and has the same scalability as the underlying multicast flow itself.

The corresponding extensions to ICE and STUN for such a STUN-triggered RSVP Proxy approach are beyond the scope of this document. They may be defined in the future in a separate document.

[4.5.](#) Application-Signaling-Triggered On-Path Proxy

In this approach, it is assumed that an entity involved in the application level signaling controls an RSVP Proxy device which is located in the datapath of the application flows (i.e. "on-path"). In this case, the RSVP Proxy entity does not attempt to understand the application reservation requirements, but instead is instructed by the entity participating in application level signaling to establish, maintain and tear down reservations as needed by the application flows. In other words, with this approach, the solution for synchronising RSVP signaling with application-level signaling is to rely on an application-level signaling entity which controls an RSVP Proxy function that sits in the flow datapath.

In some instantiations, the application-level signaling entity may be collocated with the on-path RSVP Proxy device. The figure below illustrates such an environment in the case where the application-level signaling protocol is SIP.

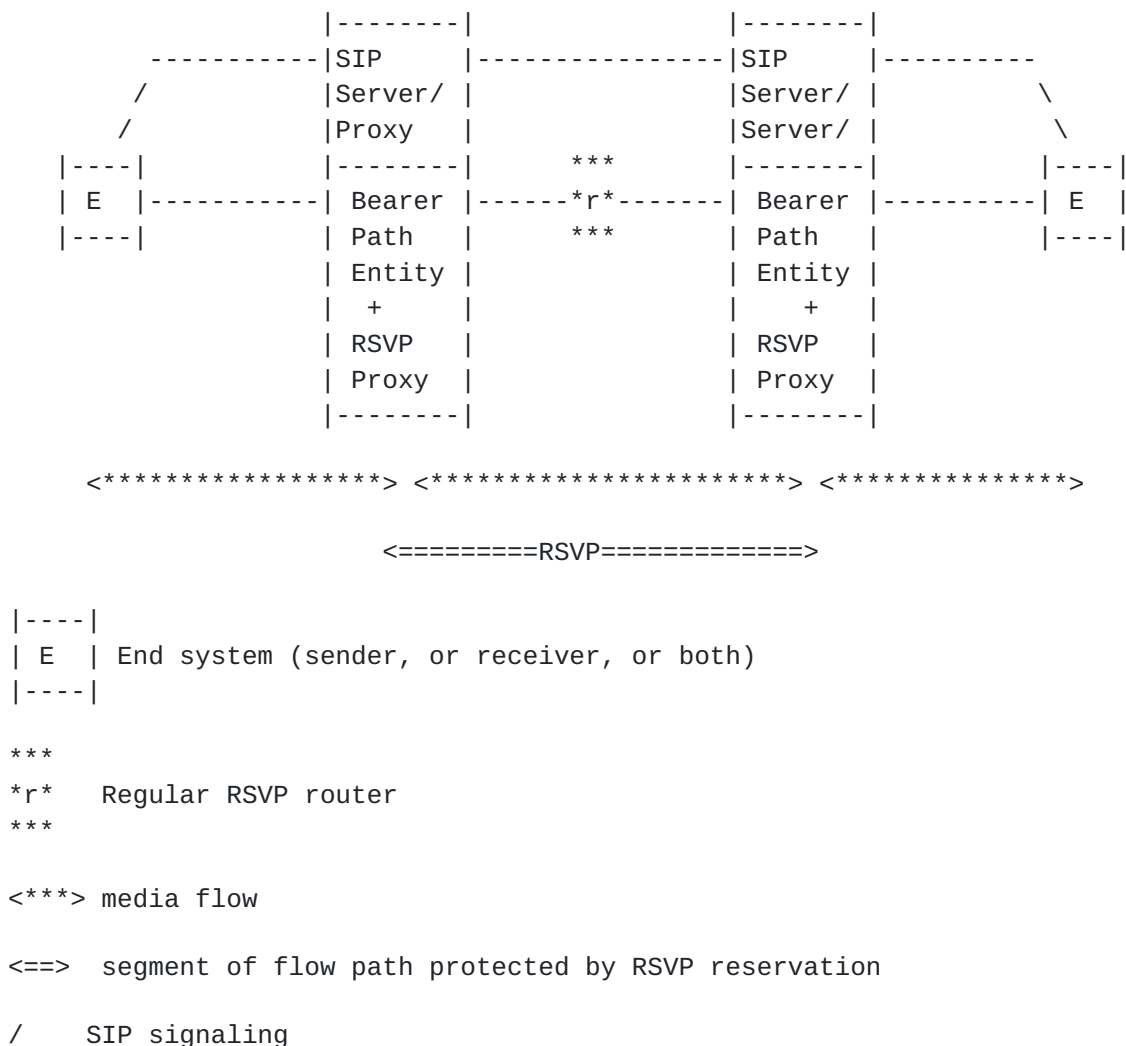
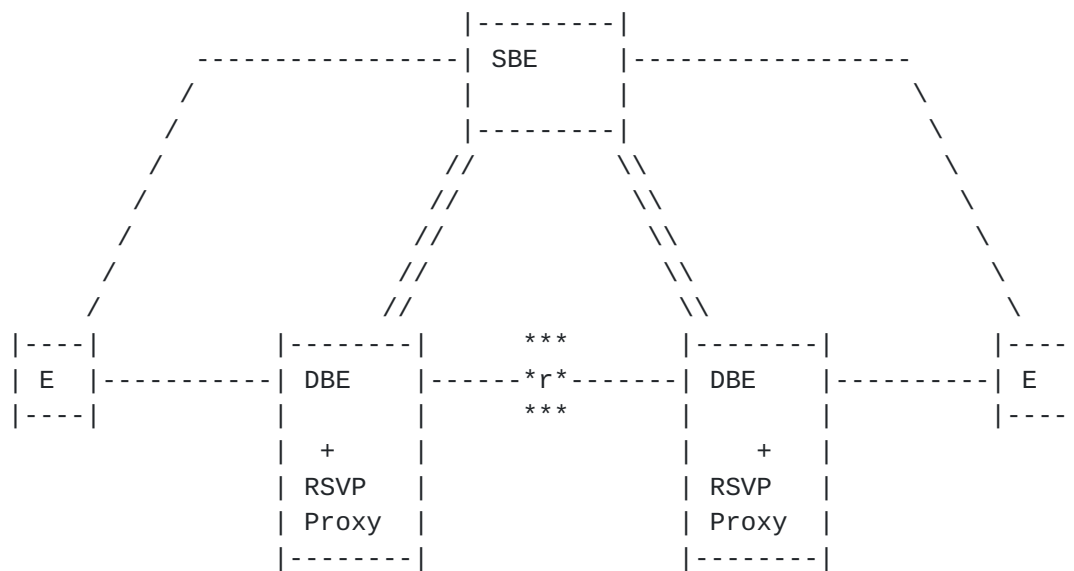


Figure 9: Application-Signaling-Triggered On-Path Proxy

This RSVP Proxy approach does not require any protocol extensions. We also observe that when multiple sessions are to be established between two given such RSVP Proxy, the RSVP Proxy have the option to establish aggregate RSVP reservations (as defined in ([[RFC3175](#)] or [[I-D.ietf-tsvwg-rsvp-ipsec](#)])) for a group of sessions, instead of establishing one RSVP reservation per session.

Consider an environment involving decomposed Session Border Controllers (SBCs). The SBC function may be broken up into a Signaling Border Element (SBE) and Datapath Border Elements (DBEs). The DBEs are remotely controlled by the SBE. This may be achieved using a protocol like [[RFC3525](#)]. Where admission control and bandwidth reservation is required between the SBEs for QoS guarantees of the sessions, the SBE could implement RSVP Proxy functionality.

In that case, the application-level signaling entity (the SBE) is remotely located from the on-path RSVP Proxy devices (located in the DBEs). Such an environment is illustrated in the Figure below.



<*****> <*****> <*****>

<=====RSVP=====>

```
|----|
| E | End system (sender, or receiver, or both)
|----|
```

r Regular RSVP router

SBE Signaling Border Element

DBE Datapath Border Element

SBE + DBE = decomposed Session Border Controller (decomposed SBC)

<***> media flow

<==> segment of flow path protected by RSVP reservation

/ SIP signaling

// control interface between the SBE and DBE

Figure 10: Remote Signaling Border Element

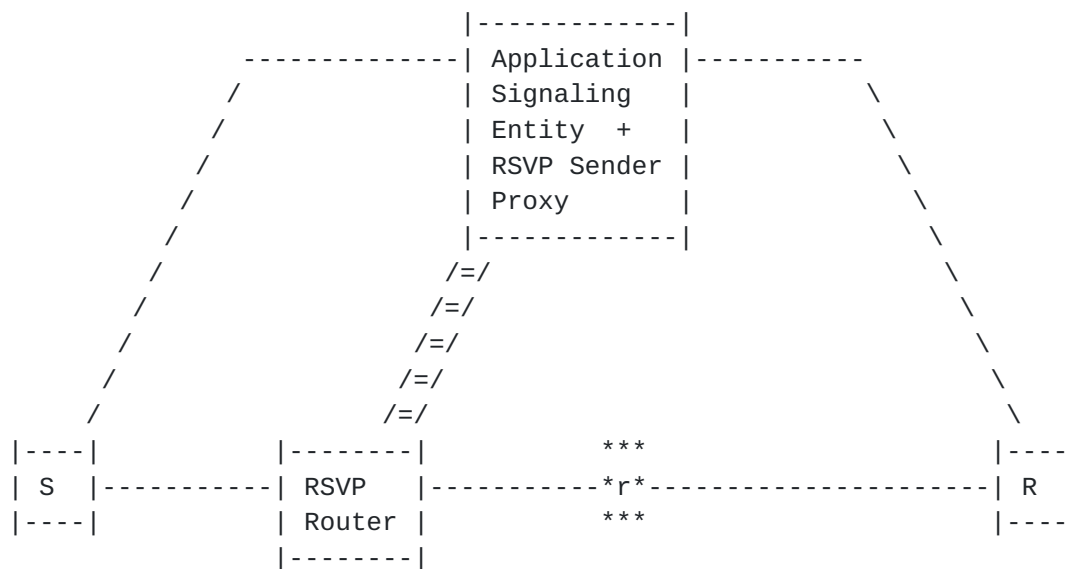
This RSVP Proxy approach does not require any extension to the RSVP protocol. However, it may require extensions to the protocol (e.g. that may be based on [RFC3525](#)) used by the application level signaling entity to control the remote on-path RSVP Proxy entities.

4.6. Application-Signaling-Triggered Off-Path Source Proxy

In this approach, it is assumed that an entity involved in the application level signaling also behaves as the RSVP Source Proxy device. However, since such an application level signaling entity is generally not on the datapath of the actual application flows, the RSVP messages need to be logically "tunnelled" between the off-path RSVP Source Proxy and a router in the datapath and upstream of the segment of the path to be protected by RSVP reservations. This is to ensure that the RSVP messages follow the exact same path as the flow they protect (as required by RSVP operations) on the segment of the end-to-end path which is to be subject to RSVP reservations.

With this approach, the solution for synchronising RSVP signaling with application-level signaling is to co-locate the RSVP Proxy function with a (typically) off-path application-level signaling entity and then "tunnel" the RSVP signaling towards the appropriate router in the datapath.

The figure below illustrates such an environment.



*****>

====RSVP====>

----		----	***
S	Sender	R	Receiver
----		----	

*** regular RSVP
*** router

<***> media flow

=> segment of flow path protected by RSVP reservation
in forward direction

/ Application level signaling

/*/ GRE-tunnelled RSVP (Path messages)

Figure 11: Application-Signaling-Triggered Off-Path Source Proxy

With the "Application-Triggered Off-Path Source Proxy" approach, the RSVP Source Proxy :

- o generates a Path message on behalf of the sender, corresponding to the reservation needed by the application and maintains the corresponding Path state. The Path message built by the Source Proxy is exactly the same as would be built by the actual sender (if it was RSVP capable), with one single exception which is that the RSVP Sender Proxy put its own IP address as the RSVP Previous Hop. In particular, it is recommended that the source address of

the Path message built by the RSVP Source Proxy be set to the IP address of the sender (not of the Sender Proxy). This helps ensuring that, in the presence of non-RSVP routers and of load-balancing in the network where the load-balancing algorithm takes into account the source IP address, the Path message generated by the Sender Proxy follows the exact same path that the actual stream sourced by the sender.

- o encapsulates the Path message into a GRE tunnel whose destination address is an RSVP Router sitting on the datapath for the flow (and upstream of the segment which requires QoS guarantees via RSVP reservation).
- o processes the corresponding received RSVP messages (including Resv messages) as per regular RSVP.
- o synchronises the RSVP reservation state with application level requirements and signaling.

Note that since the Off-Path Source Proxy encodes its own IP address as the RSVP PHOP in the Path message, the RSVP Router terminating the GRE tunnel naturally addresses all the RSVP messages travelling upstream hop-by-hop (such as Resv messages) to the Off-Path Source Proxy (without having to encapsulate those in a reverse-direction GRE tunnel to the Off-Path Proxy).

This RSVP Proxy approach does not require any extension to the RSVP protocol. It only requires tunneling of the downstream end-to-end routed RSVP messages (in particular the Path messages) in a GRE tunnel.

4.7. RSVP-Signaling-Triggered Proxy

An RSVP proxy can also be triggered and controlled through extended RSVP signaling from the remote end that is RSVP-capable (and supports these RSVP extensions for Proxy control). For example, an RSVP capable sender could send a new or extended RSVP message explicitly requesting an RSVP Proxy device on the path towards the receiver to behave as an RSVP Receiver Proxy and also to trigger a reverse direction reservation thus also behaving as a sender proxy. The new or extended RSVP message sent by the sender could also include attributes (e.g. bandwidth) for the reservations to be signaled by the RSVP Proxy.

The challenges in these explicit signaling schemes are:

- o How does the proxy differentiate between reservation requests that must be proxied, from requests that should go end-to-end?

- o How does the node sending the explicit messages know where the proxy is located, e.g., an IP address of the proxy that should reply to the signaling?
- o How are sender and receiver proxy operations differentiated?

An example of such a mechanism is presented in [\[I-D.manner-tsvwg-rsvp-proxy-sig\]](#). This scheme is primarily targeted to local access network reservations whereby an end host can request resource reservations for both incoming and outgoing flows only over the access network. This may be useful in environments where the access network is typically the bottleneck while the core is comparatively over-provisioned, as may be the case with a number of radio access technologies. In this proposal, messages targeted to the proxy are flagged with one bit in all RSVP message. Similarly, all messages sent by the proxy back are marked. The use of one bit allows differentiating between proxied and end-to-end reservations. For triggering an RSVP receiver proxy, the sender of the data sends a PATH message which is marked with the mentioned one bit. The receiver proxy is located on the signaling and data path, eventually gets the PATH message, and replies back with a RESV. A node triggers an RSVP sender proxy with a new PATH_REQUEST message, which instructs the proxy to send a PATH messages towards the triggering node. The node then replies back with a RESV. More details can be found in [\[I-D.manner-tsvwg-rsvp-proxy-sig\]](#).

Such RSVP-Signaling-Triggered Proxy approaches require RSVP signaling extensions which are outside the scope of the present document, however they can provide more flexibility in the control of the Proxy behavior (e.g. control of reverse reservation parameters).

[4.8. Other Approaches](#)

In some cases, having a full RSVP implementation running on an end host can be seen to produce excessive overhead. In end-hosts that are low in processing power and functionality, having an RSVP daemon run and take care of the signaling may introduce unnecessary overhead. One article [\[Kars01\]](#) proposes to create a remote API so that the daemon would in fact run on the end-host's default router and the end-host application would send its requests to that daemon. Thus, we can have deployments, where an end host uses some proprietary simple protocol to communicate with its pre-defined RSVP router - a form of RSVP proxy.

[5. Security Considerations](#)

In the environments concerned by this document, RSVP messages are

used to control resource reservations on a segment of the end-to-end path of flows. To ensure the integrity of the associated reservation and admission control mechanisms, the mechanisms defined in [\[RFC2747\]](#) and [\[RFC3097\]](#) can be used. Those protect RSVP messages integrity hop-by-hop and provide node authentication, thereby protecting against corruption and spoofing of RSVP messages.

An important issue regarding the various types of proxy functionality is authorization: which node is authorized to send messages on behalf of the data sender or receiver, and how is the authorization verified? [RFC 3520](#) [\[RFC3520\]](#) presents a mechanism to include authorization information within RSVP signaling messages. Subsequent versions of this document will discuss in more details how such mechanisms can be used to address security of RSVP Proxy approaches.

6. IANA Considerations

This document does not make any request to IANA registration.

7. Acknowledgments

This document benefited from earlier work on the concept of RSVP Proxy including the one documented by Silvano Gai, Dinesh Dutt, Nitsan Elfassy and Yoram Bernet. It also benefited from discussions with Pratik Bose, Chris Christou and Michael Davenport.

8. Informative References

[I-D.ietf-behave-rfc3489bis]

Rosenberg, J., "Simple Traversal Underneath Network Address Translators (NAT) (STUN)", [draft-ietf-behave-rfc3489bis-05](#) (work in progress), October 2006.

[I-D.ietf-mmusic-ice]

Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Methodology for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", [draft-ietf-mmusic-ice-13](#) (work in progress), January 2007.

[I-D.ietf-tsvwg-rsvp-ipsec]

Le Faucheur, L., "Generic Aggregate Resource ReSerVation Protocol (RSVP) Reservations", February 2007.

[I-D.ietf-tsvwg-rsvp-proxy-proto]

Le Faucheur, L., "RSVP Extensions For Path-Triggered RSVP Receiver Proxy", February 2007.

[I-D.manner-tsvwg-rsvp-proxy-sig]

Manner, J., "Localized RSVP for Controlling RSVP Proxies", October 2006.

[Kars01] Karsten, M., "Experimental Extensions to RSVP -- Remote Client and One-Pass Signalling", IWQoS Karlsruhe, Germany, 2006.

[RFC1633] Braden, B., Clark, D., and S. Shenker, "Integrated Services in the Internet Architecture: an Overview", [RFC 1633](#), June 1994.

[RFC2205] Braden, B., Zhang, L., Berson, S., Herzog, S., and S. Jamin, "Resource ReSeRVation Protocol (RSVP) -- Version 1 Functional Specification", [RFC 2205](#), September 1997.

[RFC2210] Wroclawski, J., "The Use of RSVP with IETF Integrated Services", [RFC 2210](#), September 1997.

[RFC2475] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., and W. Weiss, "An Architecture for Differentiated Services", [RFC 2475](#), December 1998.

[RFC2747] Baker, F., Lindell, B., and M. Talwar, "RSVP Cryptographic Authentication", [RFC 2747](#), January 2000.

[RFC2961] Berger, L., Gan, D., Swallow, G., Pan, P., Tommasi, F., and S. Molendini, "RSVP Refresh Overhead Reduction Extensions", [RFC 2961](#), April 2001.

[RFC3097] Braden, R. and L. Zhang, "RSVP Cryptographic Authentication -- Updated Message Type Value", [RFC 3097](#), April 2001.

[RFC3175] Baker, F., Iturralde, C., Le Faucheur, F., and B. Davie, "Aggregation of RSVP for IPv4 and IPv6 Reservations", [RFC 3175](#), September 2001.

[RFC3312] Camarillo, G., Marshall, W., and J. Rosenberg, "Integration of Resource Management and Session Initiation Protocol (SIP)", [RFC 3312](#), October 2002.

[RFC3520] Hamer, L-N., Gage, B., Kosinski, B., and H. Shieh, "Session Authorization Policy Element", [RFC 3520](#), April 2003.

[RFC3525] Groves, C., Pantaleo, M., Anderson, T., and T. Taylor,
"Gateway Control Protocol Version 1", [RFC 3525](#), June 2003.

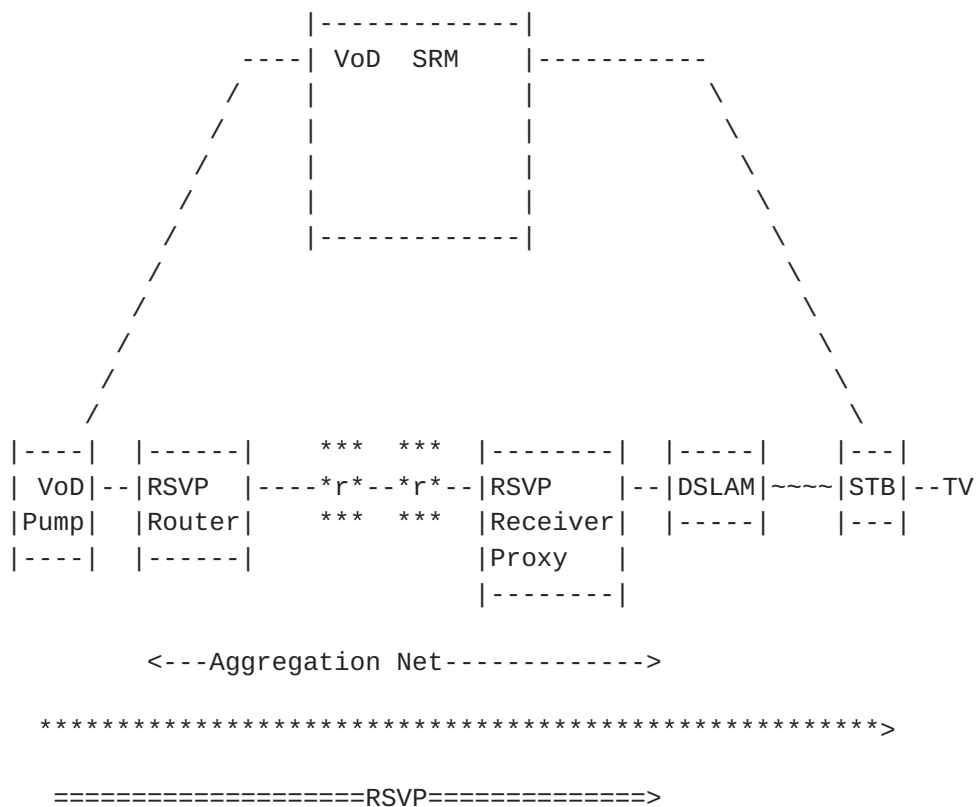
[Appendix A](#). Use Cases for RSVP Proxies

[A.1](#). RSVP-based VoD CAC in Broadband Aggregation Networks

As broadband services for residential are becoming more and more prevalent, next generation aggregation networks are being deployed in order to aggregate traffic from broadband users (whether attached via Digital Subscriber Line technology aka DSL, Fiber To The Home/Curb aka FTTx, Cable or other broadband access technology) and service providers core network or service delivery platforms. Video on Demand (VoD) services which may be offered to broadband users present significant capacity planning challenges for the aggregation network because each VoD stream requires significant dedicated sustained bandwidth (typically 2-4 Mb/s in Standard Definition TV and 8-12 Mb/s in High Definition TV), the VoD codec algorithms are very sensitive to packet loss and the load resulting from such services is very hard to predict (e.g. it can vary very suddenly with block-buster titles made available as well as with commercial offerings). As a result, transport of VoD streams on the aggregation network usually translate into a strong requirement for admission control, so that the quality of established VoD sessions can be protected at all times by rejecting the excessive session attempts during unpredictable peaks, during link or node failures, or combination of those factors.

RSVP can be used in the aggregation network for admission control of the VoD sessions. However, since Customer Premises equipment such as Set Top Boxes (which behave as the receiver for VoD streams) often do not yet support RSVP, the last IP hop in the aggregation network can behave as an RSVP Receiver Proxy. This way, RSVP can be used between VoD Pumps and the Last IP hop in the Aggregation network to perform accurate admission control of VoD streams over the resources set aside for VoD in the aggregation network (typically a certain percentage of the bandwidth of any link). As VoD streams are unidirectional, a simple "Path-Triggered" RSVP Receiver Proxy (as described in [Section 4.1](#)) is all that is required in this use case.

The Figure below illustrates operation of RSVP-based admission control of VoD sessions in an Aggregation network involving RSVP support on the VoD Pump (the senders) and RSVP Receiver Proxy on the last IP hop of the aggregation network. All the customer premises equipment remain RSVP unaware.



SRM Systems Resource Manager

```

***                |---|
*r* regular RSVP   |STB| Set Top Box
*** router         |---|

```

<***> media flow

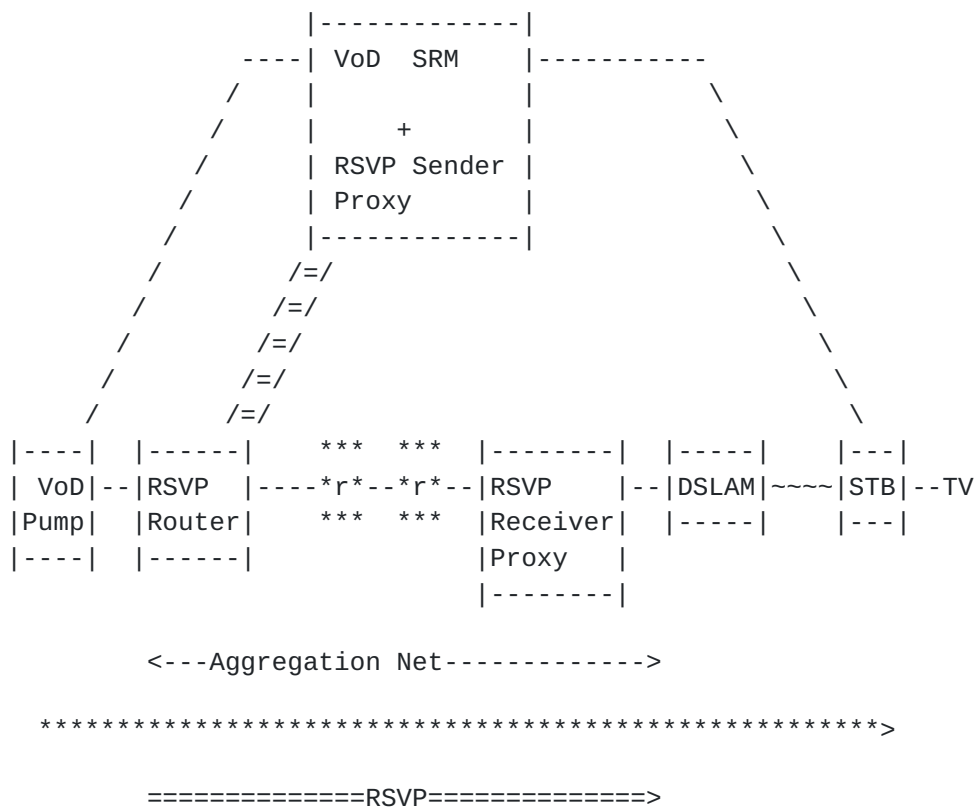
==> segment of flow path protected by RSVP reservation
in forward direction

/ VoD Application level signaling

Figure 12: VoD Use Case with Receiver Proxy

In the case where the VoD Pumps are not RSVP-capable, an Application-Signaling-triggered Off-Path Source Proxy (as described in [Section 4.6](#)) can also be implemented on the VoD Controller or Systems Resource Manager (SRM) devices typically involved in VoD deployments. The Figure below illustrates operation of RSVP-based admission control of VoD sessions in an Aggregation network involving such

Application-Signaling-triggered Off-Path Source Proxy on the SRM and RSVP Receiver Proxy on the Last IP hop of the aggregation network. All the customer premises equipment, as well as the VoD pumps, remain RSVP unaware.



SRM Systems Resource Manager

```

***          |---|
*r* regular RSVP   |STB| Set Top Box
*** router        |---|

```

<***> media flow

==> segment of flow path protected by RSVP reservation
in forward direction

/ VoD Application level signaling

/*/ GRE-tunnelled RSVP (Path messages)

Figure 13: VoD Use Case with Receiver Proxy and SRM-based Sender Proxy

The RSVP Proxy entities specified in this document play a significant role here since they allow immediate deployment of an RSVP-based admission control solution for VoD without requiring any upgrade to the huge installed base of non-RSVP-capable customer premises

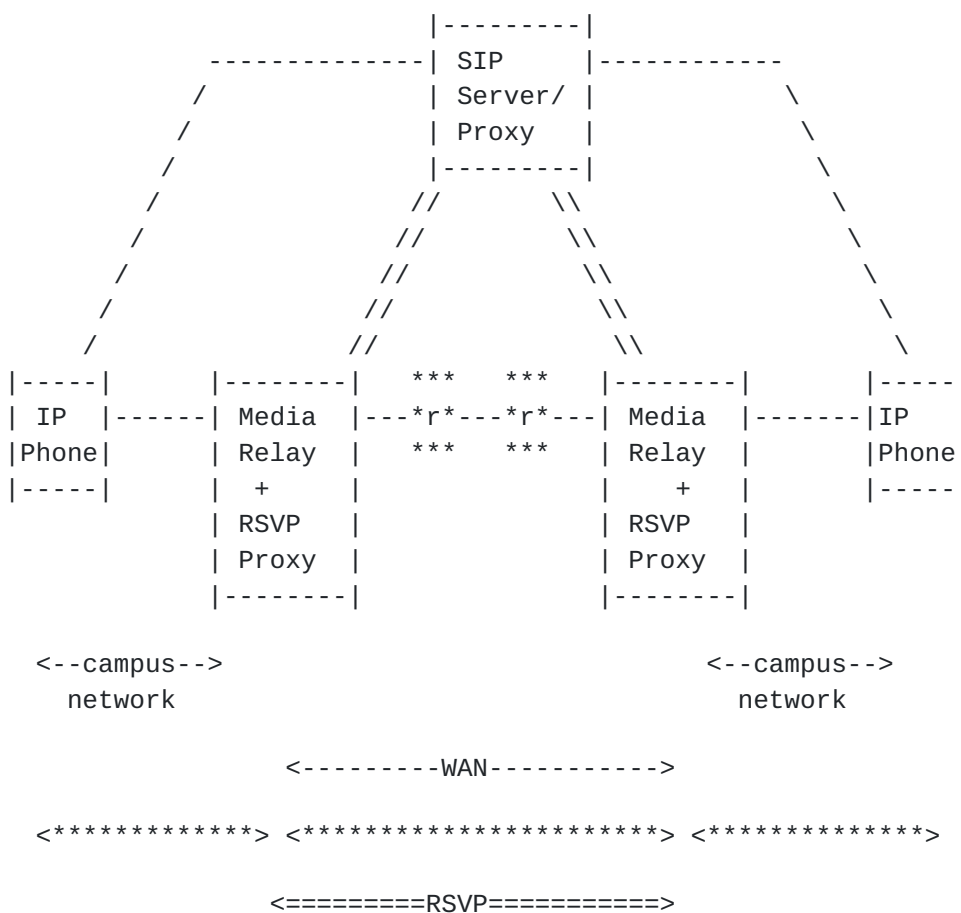
equipment. In one mode described above, they also avoid upgrade of non-RSVP-capable VoD pumps. In turn, this means that the benefits of on-path admission control can be offered to VoD services over broadband aggregation networks. Those include accurate bandwidth accounting regardless of topology (hub-and-spoke, ring, mesh, star, arbitrary combinations) and dynamic adjustment to any change in topology (such as failure, routing change, additional links...).

[A.2.](#) RSVP-based Voice/Video CAC in Enterprise WAN

More and more enterprises are migrating their telephony and videoconferencing applications onto IP. When doing so, there is a need for retaining admission control capabilities of existing TDM-based systems to ensure the QoS of these applications is maintained even when transiting through the enterprise's Wide Area Network (WAN). Since many of the endpoints already deployed (such as IP Phones or Videoconferencing terminals) are not RSVP capable, RSVP Proxy approaches are very useful by allowing deployment of an RSVP-based admission control solution over the WAN without requiring upgrade of the existing terminals.

A common deployment architecture for such environments involves Application-Signaling-Triggered On-Path RSVP Proxy as defined in [Section 4.5](#). Routers sitting at the edges of the WAN network behave as Media Relay in the datapath. For example, such a Media Relay router on the WAN Edge may terminate a call-leg from the calling IP phone and relay it to another call-leg setup on the WAN side towards another Media Relay router on the egress side of the WAN towards the called IP phone. Finally that egress Media Relay router may terminate the call leg from the ingress Media Relay router and relay it onto a call-leg setup to the called IP Phone. The Media Relay routers setup, maintain and tear down the call-legs on the WAN segment under the control of the SIP Server/Proxy. They also establish, maintain and tear-down RSVP reservations over the WAN segment for these call-legs also under the control of the SIP Server/Proxy. The SIP Server/Proxy synchronises the RSVP reservation status with the status of end-to-end calls. For example, the called IP phone will only be instructed to play a ring tone if the RSVP reservations for the corresponding WAN call leg has been successfully established.

This architecture allowing RSVP-based admission control of voice and video on the Enterprise WAN is illustrated in the Figure below.



r Regular RSVP router

<***> media flow

<==> segment of flow path protected by RSVP reservation

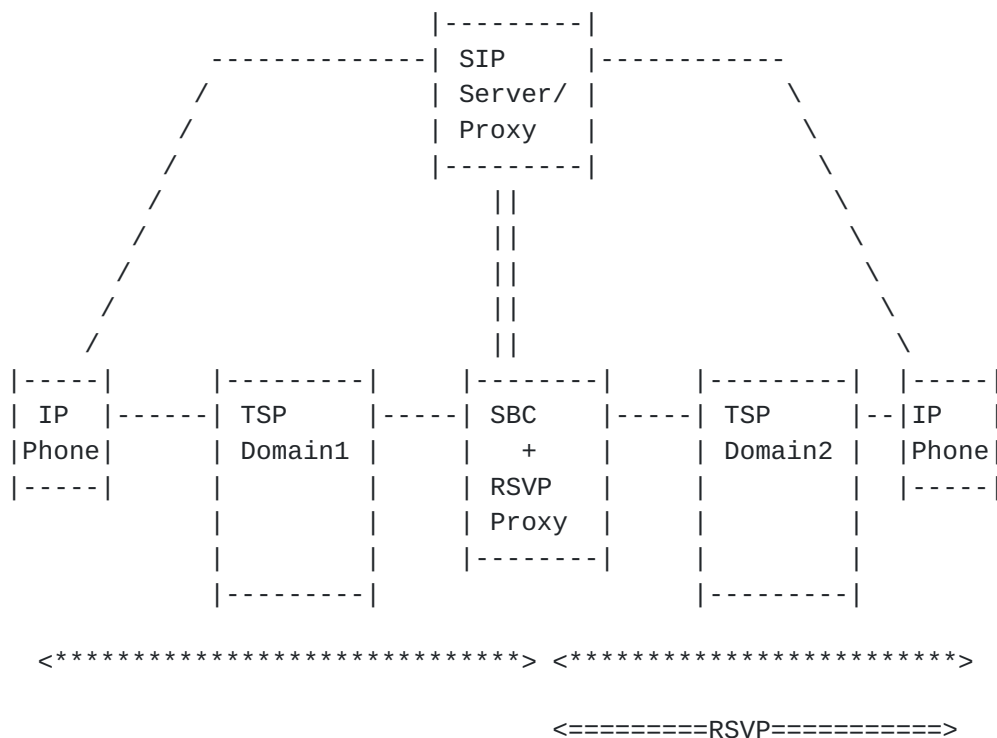
/ SIP signaling

// control interface between the SIP Server/Proxy and
Media Relay/RSVP Proxy

Figure 14: CAC on Enterprise WAN Use Case

A.3. RSVP-based Voice CAC in TSP Domain

Let us consider an environment involving multiple Telephony Service Providers (TSPs). Those may be interconnected through Session Border Controllers (SBC) which are on-path i.e. on the datapath of the voice media streams. The SBCs may be remotely controlled by a SIP Server/Proxy. Support of RSVP Proxy on one side of the SBC may be used to perform RSVP-based admission control through one of the TSP Domain, even if it is not used end-to-end (and in particular when another TSP domain remains entirely non-RSVP-aware). This relies on the Application-Signaling-Triggered On-Path RSVP Proxy presented in [Section 4.5](#). This is illustrated in the Figure below.



<***> media flow

<==> segment of flow path protected by RSVP reservation

/ SIP signaling

|| control interface between the SIP Server/Proxy and
SBC/RSVP Proxy

Figure 15: Voice CAC in TSP Domain

A.4. RSVP Proxies for Mobile Access Networks

Mobile access networks are increasingly based on IP technology. This implies that, on the network layer, all traffic, both traditional data and streamed data like audio or video, is transmitted as packets. Increasingly popular multimedia applications would benefit from better than best-effort service from the network, a forwarding service with strict Quality of Service (QoS) with guaranteed minimum bandwidth and bounded delay. Other applications, such as electronic commerce, network control and management, and remote login applications, would also benefit from a differentiated treatment.

The IETF has two main models for providing differentiated treatment

of packets in routers. The Integrated Services (IntServ) model [RFC1633] together with the Resource Reservation Protocol (RSVP) [RFC2205] [RFC2210] [RFC2961] provides per-flow guaranteed end-to-end transmission service. The Differentiated Services (DiffServ) framework [RFC2475] provides non-s signaled flow differentiation that usually provides, but does not guarantee, proper transmission service.

However, these architectures have weaknesses, for example, RSVP requires support from both communication end points, and the protocol may have potential performance issues in mobile environments. DiffServ can only provide statistical guarantees and is not well suited for dynamic environments.

Let us consider a scenario, where a fixed network correspondent node (CN) would be sending a multimedia stream to an end host behind a wireless link. If the correspondent node does not support RSVP it cannot signal its traffic characteristics to the network and request specific forwarding services. Likewise, if the correspondent node is not able to mark its traffic with a proper DiffServ Code Point (DSCP) to trigger service differentiation, the multimedia stream will get only best-effort service which may result in poor visual and audio quality in the receiving application. Even if the connecting wired network is over-provisioned, an end host would still benefit from local resource reservations, especially in wireless access networks, where the bottleneck resource is most probably the wireless link.

RSVP proxies would be a very beneficial solution to this problem. It would allow distinguishing local network reservations from the end-to-end reservations. The end host does not need to know the access network topology or the nodes that will reserve the local resources. The access network would do resource reservations for both incoming and outgoing flows based on certain criterion, e.g., filters based on application protocols. Another option is that the mobile end host makes an explicit reservation that identifies the intention and the access network will find the correct local access network node(s) to respond to the reservation. RSVP proxies would, thus, allow resource reservation over the segment which is the most likely bottleneck, the wireless connectivity. If the wireless access network uses a local mobility management mechanism, where the IP address of the mobile node does not change during handover, RSVP reservations would follow the mobile node movement.

Authors' Addresses

Francois Le Faucheur
Cisco Systems
Greenside, 400 Avenue de Roumanille
Sophia Antipolis 06410
France

Phone: +33 4 97 23 26 19
Email: flefauch@cisco.com

Jukka Manner
University of Helsinki
P.O. Box 68
University of Helsinki FIN-00014 University of Helsinki
Finland

Phone: +358 9 191 51298
Email: jmanner@cs.helsinki.fi
URI: <http://www.cs.helsinki.fi/u/jmanner/>

Dan Wing
Cisco Systems
170 West Tasman Drive
San Jose, CA 95134
United States

Email: dwing@cisco.com

Allan Guillou
Neuf Cegetel
40-42 Quai du Point du Jour
Boulogne-Billancourt, 92659
France

Email: allan.guillou@neufcegetel.fr

Full Copyright Statement

Copyright (C) The IETF Trust (2007).

This document is subject to the rights, licenses and restrictions contained in [BCP 78](#), and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY, THE IETF TRUST AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights 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; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in [BCP 78](#) and [BCP 79](#).

Copies of IPR disclosures made to the IETF Secretariat 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 on-line IPR repository at <http://www.ietf.org/ipr>.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).

