Network Working Group
Request for Comments: 4230
Category: Informational

H. Tschofenig
Siemens
R. Graveman
RFG Security

December 2005

RSVP Security Properties

Status of This Memo

Copyright Notice

Abstract

This document summarizes the security properties of RSVP. The goal of this analysis is to benefit from previous work done on RSVP and to capture knowledge about past activities.

Table of Contents

1. Introduction

As the work of the NSIS working group began, concerns about security and its implications for the design of a signaling protocol were raised. In order to understand the security properties and available options of RSVP, a number of documents have to be read. This document summarizes the security properties of RSVP and is part of the overall process of analyzing other signaling protocols and learning from their design considerations. This document should also provide a starting point for further discussions.

The content of this document is organized as follows. Section 2 introduces the terminology used throughout the document. Section 3 provides an overview of the security mechanisms provided by RSVP including the INTEGRITY object, a description of the identity representation within the POLICY_DATA object (i.e., user authentication), and the RSVP Integrity Handshake mechanism. Section 4 provides a more detailed discussion of the mechanisms used and tries to describe in detail the mechanisms provided. Several miscellaneous issues are covered in Section 5.

RSVP also supports multicast, but this document does not address security aspects for supporting multicast QoS signaling. Multicast is currently outside the scope of the NSIS working group.

Although a variation of RSVP, namely RSVP-TE, is used in the context of MPLS to distribute labels for a label switched path, its usage is different from the usage scenarios envisioned for NSIS. Hence, this document does not address RSVP-TE or its security properties.

2. Terminology and Architectural Assumptions

This section describes some important terms and explains some architectural assumptions.

- Chain-of-Trust:

  The security mechanisms supported by RSVP [1] heavily rely on optional hop-by-hop protection, using the built-in INTEGRITY object. Hop-by-hop security with the INTEGRITY object inside the RSVP message thereby refers to the protection between RSVP-supporting network elements. Additionally, there is the notion of policy-aware nodes that understand the POLICY_DATA element within the RSVP message. Because this element also includes an INTEGRITY object, there is an additional hop-by-hop security mechanism that provides security between policy-aware nodes. Policy-ignorant nodes are not affected by the inclusion of this object in the POLICY_DATA element, because they do not try to interpret it.

To protect signaling messages that are possibly modified by each RSVP router along the path, it must be assumed that each incoming request is authenticated, integrity protected, and replay protected. This provides protection against bogus messages injected by unauthorized nodes. Furthermore, each RSVP-aware router is assumed to behave in the expected manner. Outgoing messages transmitted to the next-hop network element receive new protection according to RSVP security processing.

Using the mechanisms described above, a chain-of-trust is created whereby a signaling message that is transmitted by router A via router B and received by router C is supposed to be secure if routers A and B and routers B and C share security associations and all routers behave as expected. Hence, router C trusts router A although router C does not have a direct security association with router A. We can therefore conclude that the protection achieved with this hop-by-hop security for the chain-of-trust is no better than the weakest link in the chain.

If one router is malicious (for example, because an adversary has control over this router), then it can arbitrarily modify messages, cause unexpected behavior, and mount a number of attacks that are not limited to QoS signaling. Additionally, it must be mentioned that some protocols demand more protection than others (which depends, in part, on which nodes are executing these protocols). For example, edge devices, where end-users are attached, may be more likely to be attacked in comparison with the more secure core network of a service provider. In some cases, a network service provider may choose not to use the RSVP-provided security mechanisms inside the core network because a different security protection is deployed.

Section 6 of [2] mentions the term chain-of-trust in the context of RSVP integrity protection. In Section 6 of [14] the same term is used in the context of user authentication with the INTEGRITY object inside the POLICY_DATA element. Unfortunately, the term is not explained in detail and the assumptions behind it are not clearly specified.

- Host and User Authentication:

  The presence of RSVP protection and a separate user identity representation leads to the fact that both user-identity and host-identity are used for RSVP protection. Therefore, user-based security and host-based security are covered separately, because of the different authentication mechanisms provided. To avoid confusion about the different concepts, Section 3.4 describes the concept of user authentication in more detail.

- Key Management:

  It is assumed that most of the security associations required for the protection of RSVP signaling messages are already available, and hence key management was done in advance. There is, however, an exception with respect to support for Kerberos. Using Kerberos, an entity is able to distribute a session key used for RSVP signaling protection.

- RSVP INTEGRITY and POLICY_DATA INTEGRITY Objects:

  RSVP uses an INTEGRITY object in two places in a message. The first is in the RSVP message itself and covers the entire RSVP message as defined in [1]. The second is included in the POLICY_DATA object and defined in [2]. To differentiate the two objects by their scope of protection, the two terms RSVP INTEGRITY and POLICY_DATA INTEGRITY object are used, respectively. The data structure of the two objects, however, is the same.

- Hop versus Peer:

  In the past, the terminology for nodes addressed by RSVP has been discussed considerably. In particular, two favorite terms have been used: hop and peer. This document uses the term hop, which is different from an IP hop. Two neighboring RSVP nodes communicating with each other are not necessarily neighboring IP nodes (i.e., they may be more than one IP hop away).

3. Overview

This section describes the security mechanisms provided by RSVP. Although use of IPsec is mentioned in Section 10 of [1], the other security mechanisms primarily envisioned for RSVP are described.

3.1. The RSVP INTEGRITY Object

The RSVP INTEGRITY object is the major component of RSVP security protection. This object is used to provide integrity and replay protection for the content of the signaling message between two RSVP participating routers or between an RSVP router and host. Furthermore, the RSVP INTEGRITY object provides data origin authentication. The attributes of the object are briefly described:

- Flags field:

  The Handshake Flag is the only defined flag. It is used to synchronize sequence numbers if the communication gets out of sync (e.g., it allows a restarting host to recover the most

recent sequence number). Setting this flag to one indicates that the sender is willing to respond to an Integrity Challenge message. This flag can therefore be seen as a negotiation capability transmitted within each INTEGRITY object.

- Key Identifier:

  The Key Identifier selects the key used for verification of the Keyed Message Digest field and, hence, must be unique for the sender. It has a fixed 48-bit length. The generation of this Key Identifier field is mostly a decision of the local host. [1] describes this field as a combination of an address, sending interface, and key number. We assume that the Key Identifier is simply a (keyed) hash value computed over a number of fields, with the requirement to be unique if more than one security association is used in parallel between two hosts (e.g., as is the case with security associations having overlapping lifetimes). A receiving system uniquely identifies a security association based on the Key Identifier and the sender's IP address. The sender's IP address may be obtained from the RSVP_HOP object or from the source IP address of the packet if the RSVP_HOP object is not present. The sender uses the outgoing interface to determine which security association to use. The term "outgoing interface" may be confusing. The sender selects the security association based on the receiver's IP address (i.e., the address of the next RSVP-capable router). The process of determining which node is the next RSVP-capable router is not further specified and is likely to be statically configured.

- Sequence Number:

  The sequence number used by the INTEGRITY object is 64 bits in length, and the starting value can be selected arbitrarily. The length of the sequence number field was chosen to avoid exhaustion during the lifetime of a security association as stated in Section 3 of [1]. In order for the receiver to distinguish between a new and a replayed message, the sequence number must be monotonically incremented (modulo 2^64) for each message. We assume that the first sequence number seen (i.e., the starting sequence number) is stored somewhere. The modulo-operation is required because the starting sequence number may be an arbitrary number. The receiver therefore only accepts packets with a sequence number larger (modulo 2^64) than the previous packet. As explained in [1] this process is started by handshaking and agreeing on an initial sequence number. If no such handshaking is available then the initial sequence number must be part of the establishment of the security association.

The generation and storage of sequence numbers is an important step in preventing replay attacks and is largely determined by the capabilities of the system in the presence of system crashes, failures, and restarts. Section 3 of [1] explains some of the most important considerations. However, the description of how the receiver distinguishes proper from improper sequence numbers is incomplete: it implicitly assumes that gaps large enough to cause the sequence number to wrap around cannot occur.

If delivery in order were guaranteed, the following procedure would work: the receiver keeps track of the first sequence number received, INIT-SEQ, and the most recent sequence number received, LAST-SEQ, for each key identifier in a security association. When the first message is received, set INIT-SEQ = LAST-SEQ = value received and accept. When a subsequent message is received, if its sequence number is strictly between LAST-SEQ and INIT-SEQ, (modulo 2^64), accept and update LAST-SEQ with the value just received. If it is between INIT-SEQ and LAST-SEQ, inclusive, (modulo 2^64), reject and leave the value of LAST-SEQ unchanged. Because delivery in order is not guaranteed, the above rules need to be combined with a method of allowing a fixed sized window in the neighborhood of LAST-SEQ for out-of-order delivery, for example, as described in Appendix C of [3].

- Keyed Message Digest:

  The Keyed Message Digest is a security mechanism built into RSVP that used to provide integrity protection of a signaling message (including its sequence number). Prior to computing the value for the Keyed Message Digest field, the Keyed Message Digest field itself must be set to zero and a keyed hash computed over the entire RSVP packet. The Keyed Message Digest field is variable in length but must be a multiple of four octets. If HMAC-MD5 is used, then the output value is 16 bytes long. The keyed hash function HMAC-MD5 [4] is required for an RSVP implementation, as noted in Section 1 of [1]. Hash algorithms other than MD5 [5], like SHA-1 [15], may also be supported.

  The key used for computing this Keyed Message Digest may be obtained from the pre-shared secret, which is either manually distributed or the result of a key management protocol. No key management protocol, however, is specified to create the desired security associations. Also, no guidelines for key length are given. It should be recommended that HMAC-MD5 keys be 128 bits and SHA-1 keys 160 bits, as in IPsec AH [16] and ESP [17].

3.2. Security Associations

Different attributes are stored for security associations of sending and receiving systems (i.e., unidirectional security associations). The sending system needs to maintain the following attributes in such a security association [1]:

- Authentication algorithm and algorithm mode
- Key
- Key Lifetime
- Sending Interface
- Latest sequence number (received with this key identifier)

The receiving system has to store the following fields:

- Authentication algorithm and algorithm mode
- Key
- Key Lifetime
- Source address of the sending system
- List of last n sequence numbers (received with this key identifier)

Note that the security associations need to have additional fields to indicate their state. It is necessary to have overlapping lifetimes of security associations to avoid interrupting an ongoing communication because of expired security associations. During such a period of overlapping lifetime it is necessary to authenticate with either one or both active keys. As mentioned in [1], a sender and a receiver may have multiple active keys simultaneously. If more than one algorithm is supported, then the algorithm used must be specified for a security association.

3.3. RSVP Key Management Assumptions

RFC 2205 [6] assumes that security associations are already available. An implementation must support manual key distribution as noted in Section 5.2 of [1]. Manual key distribution, however, has different requirements for key storage; a simple plaintext ASCII file may be sufficient in some cases. If multiple security associations with different lifetimes need to be supported at the same time, then

a key engine would be more appropriate. Further security requirements listed in Section 5.2 of [1] are the following:

- The manual deletion of security associations must be supported.
- The key storage should persist during a system restart.
- Each key must be assigned a specific lifetime and a specific Key Identifier.

3.4. Identity Representation

In addition to host-based authentication with the INTEGRITY object inside the RSVP message, user-based authentication is available as introduced in [2]. Section 2 of [7] states that "Providing policy based admission control mechanism based on user identities or application is one of the prime requirements." To identify the user or the application, a policy element called AUTH_DATA, which is contained in the POLICY_DATA object, is created by the RSVP daemon at the user's host and transmitted inside the RSVP message. The structure of the POLICY_DATA element is described in [2]. Network nodes acting as policy decision points (PDPs) then use the information contained in the AUTH_DATA element to authenticate the user and to allow policy-based admission control to be executed. As mentioned in [7], the policy element is processed and the PDP replaces the old element with a new one for forwarding to the next hop router.

A detailed description of the POLICY_DATA element can be found in [2]. The attributes contained in the authentication data policy element AUTH_DATA, which is defined in [7], are briefly explained in this Section. Figure 1 shows the abstract structure of the RSVP message with its security-relevant objects and the scope of protection. The RSVP INTEGRITY object (outer object) covers the entire RSVP message, whereas the POLICY_DATA INTEGRITY object only covers objects within the POLICY_DATA element.

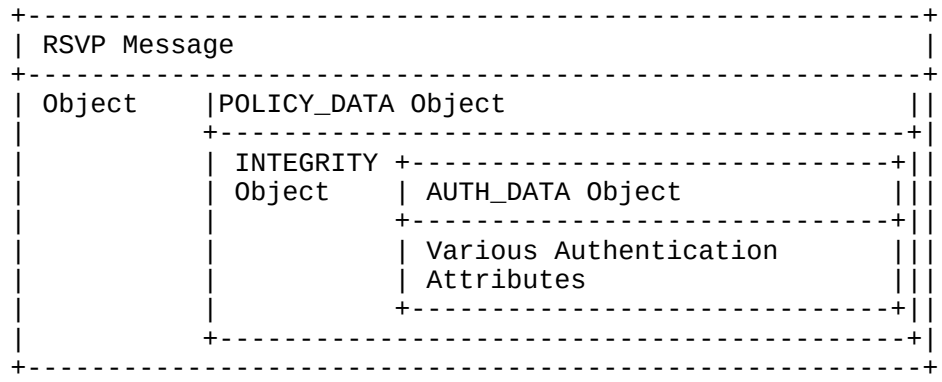

Figure 1: Security Relevant Objects and Elements within the RSVP Message.

The AUTH_DATA object contains information for identifying users and applications together with credentials for those identities. The main purpose of these identities seems to be usage for policy-based admission control and not authentication and key management. As noted in Section 6.1 of [7], an RSVP message may contain more than one POLICY_DATA object and each of them may contain more than one AUTH_DATA object. As indicated in Figure 1 and in [7], one AUTH_DATA object may contain more than one authentication attribute. A typical configuration for Kerberos-based user authentication includes at least the Policy Locator and an attribute containing the Kerberos session ticket.

Successful user authentication is the basis for executing policy-based admission control. Additionally, other information such as time-of-day, application type, location information, group membership, etc. may be relevant to the implementation of an access control policy.

The following attributes are defined for use in the AUTH_DATA object:

- Policy Locator
  - ASCII_DN
  - UNICODE_DN
  - ASCII_DN_ENCRYPT
  - UNICODE_DN_ENCRYPT

The policy locator string is an X.500 distinguished name (DN) used to locate user or application-specific policy information. The four types of X.500 DNs are listed above. The first two types are the ASCII and the Unicode representation of the user or application DN identity. The two "encrypted" distinguished name types are either encrypted with the Kerberos session key or with the private key of the user's digital certificate (i.e., digitally signed). The term "encrypted together with a digital signature" is easy to misconceive. If user identity confidentiality is provided, then the policy locator has to be encrypted with the public key of the recipient. How to obtain this public key is not described in the document. This detail may be specified in a concrete architecture in which RSVP is used.

o Credentials

Two cryptographic credentials are currently defined for a user: authentication with Kerberos V5 [8], and authentication with the help of digital signatures based on X.509 [18] and PGP [19]. The following list contains all defined credential types currently available and defined in [7]:

Credential Type	Description
ASCII_ID	User or application identity encoded as an ASCII string
UNICODE_ID	User or application identity encoded as a Unicode string
KERBEROS_TKT	Kerberos V5 session ticket
X509_V3_CERT	X.509 V3 certificate
PGP_CERT	PGP certificate

Figure 2: Credentials Supported in RSVP.

The first two credentials contain only a plaintext string, and therefore they do not provide cryptographic user authentication. These plaintext strings may be used to identify applications, that are included for policy-based admission control. Note that these plain-text identifiers may, however, be protected if either the RSVP INTEGRITY or the

INTEGRITY object of the POLICY_DATA element is present. Note that the two INTEGRITY objects can terminate at different entities depending on the network structure. The digital signature may also provide protection of application identifiers. A protected application identity (and the entire content of the POLICY_DATA element) cannot be modified as long as no policy-ignorant nodes are encountered in between.

A Kerberos session ticket, as previously mentioned, is the ticket of a Kerberos AP_REQ message [8] without the Authenticator. Normally, the AP_REQ message is used by a client to authenticate to a server. The INTEGRITY object (e.g., of the POLICY_DATA element) provides the functionality of the Kerberos Authenticator, namely protecting against replay and showing that the user was able to retrieve the session key following the Kerberos protocol. This is, however, only the case if the Kerberos session was used for the keyed message digest field of the INTEGRITY object. Section 7 of [1] discusses some issues for establishment of keys for the INTEGRITY object. The establishment of the security association for the RSVP INTEGRITY object with the inclusion of the Kerberos Ticket within the AUTH_DATA element may be complicated by the fact that the ticket can be decrypted by node B, whereas the RSVP INTEGRITY object terminates at a different host C.

The Kerberos session ticket contains, among many other fields, the session key. The Policy Locator may also be encrypted with the same session key. The protocol steps that need to be executed to obtain such a Kerberos service ticket are not described in [7] and may involve several roundtrips, depending on many Kerberos-related factors. As an optimization, the Kerberos ticket does not need to be included in every RSVP message, as described in Section 7.1 of [1]. Thus, the receiver must store the received service ticket. If the lifetime of the ticket has expired, then a new service ticket must be sent. If the receiver lost its state information (because of a crash or restart) then it may transmit an Integrity Challenge message to force the sender to re-transmit a new service ticket.

If either the X.509 V3 or the PGP certificate is included in the policy element, then a digital signature must be added. The digital signature computed over the entire AUTH_DATA object provides authentication and integrity protection. The SubType of the digital signature authentication attribute is set to zero before computing the digital signature. Whether or not a guarantee of freshness with replay protection (either

timestamps or sequence numbers) is provided by the digital signature is an open issue as discussed in Section 4.3.

- o Digital Signature

  The digital signature computed over the contents of the AUTH_DATA object must be the last attribute. The algorithm used to compute the digital signature depends on the authentication mode listed in the credential. This is only partially true, because, for example, PGP again allows different algorithms to be used for computing a digital signature. The algorithm identifier used for computing the digital signature is not included in the certificate itself. The algorithm identifier included in the certificate only serves the purpose of allowing the verification of the signature computed by the certificate authority (except for the case of self-signed certificates).

- o Policy Error Object

  The Policy Error Object is used in the case of a failure of policy-based admission control or other credential verification. Currently available error messages allow notification if the credentials are expired (EXPIRED_CREDENTIALS), if the authorization process disallowed the resource request (INSUFFICIENT_PRIVILEGES), or if the given set of credentials is not supported (UNSUPPORTED_CREDENTIAL_TYPE). The last error message returned by the network allows the user's host to discover the type of credentials supported. Particularly for mobile environments this might be quite inefficient. Furthermore, it is unlikely that a user supports different types of credentials. The purpose of the error message IDENTITY_CHANGED is unclear. Also, the protection of the error message is not discussed in [7].

3.5. RSVP Integrity Handshake

The Integrity Handshake protocol was designed to allow a crashed or restarted host to obtain the latest valid challenge value stored at the receiving host. Due to the absence of key management, it must be guaranteed that two messages do not use the same sequence number with the same key. A host stores the latest sequence number of a cryptographically verified message. An adversary can replay eavesdropped packets if the crashed host has lost its sequence numbers. A signaling message from the real sender with a new sequence number would therefore allow the crashed host to update the sequence number field and prevent further replays. Hence, if there

is a steady flow of RSVP-protected messages between the two hosts, an attacker may find it difficult to inject old messages, because new, authenticated messages with higher sequence numbers arrive and get stored immediately.

The following description explains the details of an RSVP Integrity Handshake that is started by Node A after recovering from a synchronization failure:

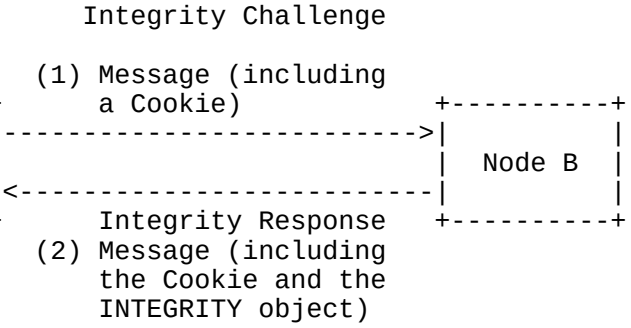

Figure 3: RSVP Integrity Handshake.

The details of the messages are as follows:

CHALLENGE:=(Key Identifier, Challenge Cookie)

Integrity Challenge Message:=(Common Header, CHALLENGE)

Integrity Response Message:=(Common Header, INTEGRITY, CHALLENGE)

The "Challenge Cookie" is suggested to be a MD5 hash of a local secret and a timestamp [1].

The Integrity Challenge message is not protected with an INTEGRITY object as shown in the protocol flow above. As explained in Section 10 of [1] this was done to avoid problems in situations where both communicating parties do not have a valid starting sequence number.

Using the RSVP Integrity Handshake protocol is recommended although it is not mandatory (because it may not be needed in all network environments).

4. Detailed Security Property Discussion

This section describes the protection of the RSVP-provided mechanisms for authentication, authorization, integrity and replay protection individually, user identity confidentiality, and confidentiality of the signaling messages,

4.1. Network Topology

This paragraph shows the basic interfaces in a simple RSVP network architecture. The architecture below assumes that there is only a single domain and that the two routers are RSVP- and policy-aware. These assumptions are relaxed in the individual paragraphs, as necessary. Layer 2 devices between the clients and their corresponding first-hop routers are not shown. Other network elements like a Kerberos Key Distribution Center and, for example, an LDAP server from which the PDP retrieves its policies are also omitted. The security of various interfaces to the individual servers (KDC, PDP, etc.) depends very much on the security policy of a specific network service provider.

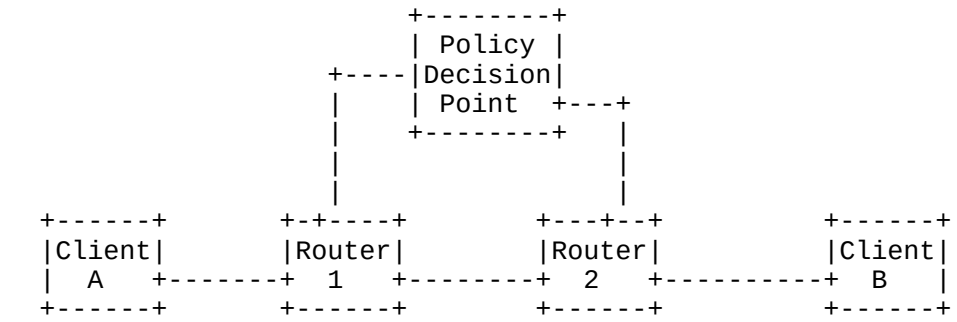

Figure 4: Simple RSVP Architecture.

4.2. Host/Router

When considering authentication in RSVP, it is important to make a distinction between user and host authentication of the signaling messages. The host is authenticated using the RSVP INTEGRITY object, whereas credentials inside the AUTH_DATA object can be used to authenticate the user. In this section, the focus is on host authentication, whereas the next section covers user authentication.

(1) Authentication

The term "host authentication" is used above, because the selection of the security association is bound to the host's IP

address, as mentioned in Section 3.1 and Section 3.2. Depending on the key management protocol used to create this security association and the identity used, it is also possible to bind a user identity to this security association. Because the key management protocol is not specified, it is difficult to evaluate this part, and hence we speak about data-origin authentication based on the host's identity for RSVP INTEGRITY objects. The fact that the host identity is used for selecting the security association has already been described in Section 3.1.

Data-origin authentication is provided with a keyed hash value computed over the entire RSVP message, excluding the keyed message digest field itself. The security association used between the user's host and the first-hop router is, as previously mentioned, not established by RSVP, and it must therefore be available before signaling is started.

* Kerberos for the RSVP INTEGRITY object

  As described in Section 7 of [1], Kerberos may be used to create the key for the RSVP INTEGRITY object. How to learn the principal name (and realm information) of the other node is outside the scope of [1]. [20] describes a way to distribute principal and realm information via DNS, which can be used for this purpose (assuming that the FQDN or the IP address of the other node for which this information is desired is known). All that is required is to encapsulate the Kerberos ticket inside the policy element. It is furthermore mentioned that Kerberos tickets with expired lifetime must not be used, and the initiator is responsible for requesting and exchanging a new service ticket before expiration.

  RSVP multicast processing in combination with Kerberos involves additional considerations. Section 7 of [1] states that in the multicast case all receivers must share a single key with the Kerberos Authentication Server (i.e., a single principal used for all receivers). From a personal discussion with Rodney Hess, it seems that there is currently no other solution available in the context of Kerberos. Multicast handling therefore leaves some open questions in this context.

  In the case where one entity crashed, the established security association is lost and therefore the other node must retransmit the service ticket. The crashed entity can use an Integrity Challenge message to request a new Kerberos ticket to be retransmitted by the other node. If a node receives such a request, then a reply message must be returned.

(2) Integrity protection

Integrity protection between the user's host and the first-hop router is based on the RSVP INTEGRITY object. HMAC-MD5 is preferred, although other keyed hash functions may also be used within the RSVP INTEGRITY object. In any case, both communicating entities must have a security association that indicates the algorithm to use. This may, however, be difficult, because no negotiation protocol is defined to agree on a specific algorithm. Hence, if RSVP is used in a mobile environment, it is likely that HMAC-MD5 is the only usable algorithm for the RSVP INTEGRITY object. Only in local environments may it be useful to switch to a different keyed hash algorithm. The other possible alternative is that every implementation support the most important keyed hash algorithms. e.g., MD5, SHA-1, RIPEMD-160, etc. HMAC-MD5 was chosen mainly because of its performance characteristics. The weaknesses of MD5 [21] are known and were initially described in [22]. Other algorithms like SHA-1 [15] and RIPEMD-160 [21] have stronger security properties.

(3) Replay Protection

The main mechanism used for replay protection in RSVP is based on sequence numbers, whereby the sequence number is included in the RSVP INTEGRITY object. The properties of this sequence number mechanism are described in Section 3.1 of [1]. The fact that the receiver stores a list of sequence numbers is an indicator for a window mechanism. This somehow conflicts with the requirement that the receiver only has to store the highest number given in Section 3 of [1]. We assume that this is an oversight. Section 4.2 of [1] gives a few comments about the out-of-order delivery and the ability of an implementation to specify the replay window. Appendix C of [3] describes a window mechanism for handling out-of-sequence delivery.

(4) Integrity Handshake

The mechanism of the Integrity Handshake is explained in Section 3.5. The Cookie value is suggested to be a hash of a local secret and a timestamp. The Cookie value is not verified by the receiver. The mechanism used by the Integrity Handshake is a simple Challenge/Response message, which assumes that the key shared between the two hosts survives the crash. If, however, the security association is dynamically created, then this assumption may not be true.

In Section 10 of [1], the authors note that an adversary can create a faked Integrity Handshake message that includes challenge cookies. Subsequently, it could store the received response and later try to replay these responses while a responder recovers from a crash or restart. If this replayed Integrity Response value is valid and has a lower sequence number than actually used, then this value is stored at the recovering host. In order for this attack to be successful, the adversary must either have collected a large number of challenge/response value pairs or have "discovered" the cookie generation mechanism (for example by knowing the local secret). The collection of Challenge/Response pairs is even more difficult, because they depend on the Cookie value, the sequence number included in the response message, and the shared key used by the INTEGRITY object.

(5) Confidentiality

Confidentiality is not considered to be a security requirement for RSVP. Hence, it is not supported by RSVP, except as described in paragraph d) of Section 4.3. This assumption may not hold, however, for enterprises or carriers who want to protect billing data, network usage patterns, or network configurations, in addition to users' identities, from eavesdropping and traffic analysis. Confidentiality may also help make certain other attacks more difficult. For example, the PathErr attack described in Section 5.2 is harder to carry out if the attacker cannot observe the Path message to which the PathErr corresponds.

(6) Authorization

The task of authorization consists of two subcategories: network access authorization and RSVP request authorization. Access authorization is provided when a node is authenticated to the network, e.g., using EAP [23] in combination with AAA protocols (for example, RADIUS [24] or DIAMETER [9]). Issues related to network access authentication and authorization are outside the scope of RSVP.

The second authorization refers to RSVP itself. Depending on the network configuration:

* the router either forwards the received RSVP request to the policy decision point (e.g., using COPS [10] and [11]) to request that an admission control procedure be executed, or

* the router supports the functionality of a PDP and, therefore, there is no need to forward the request, or

* the router may already be configured with the appropriate policy information to decide locally whether to grant this request.

Based on the result of the admission control, the request may be granted or rejected. Information about the resource-requesting entity must be available to provide policy-based admission control.

(7) Performance

The computation of the keyed message digest for an RSVP INTEGRITY object does not represent a performance problem. The protection of signaling messages is usually not a problem, because these messages are transmitted at a low rate. Even a high volume of messages does not cause performance problems for an RSVP router due to the efficiency of the keyed message digest routine.

Dynamic key management, which is computationally more demanding, is more important for scalability. Because RSVP does not specify a particular key exchange protocol, it is difficult to estimate the effort needed to create the required security associations. Furthermore, the number of key exchanges to be triggered depends on security policy issues like lifetime of a security association, required security properties of the key exchange protocol, authentication mode used by the key exchange protocol, etc. In a stationary environment with a single administrative domain, manual security association establishment may be acceptable and may provide the best performance characteristics. In a mobile environment, asymmetric authentication methods are likely to be used with a key exchange protocol, and some sort of public key or certificate verification needs to be supported.

4.3. User to PEP/PDP

As noted in the previous section, RSVP supports both user-based and host-based authentication. Using RSVP, a user may authenticate to the first hop router or to the PDP as specified in [1], depending on the infrastructure provided by the network domain or the architecture used (e.g., the integration of RSVP and Kerberos V5 into the Windows 2000 Operating System [25]). Another architecture in which RSVP is tightly integrated is the one specified by the PacketCable organization. The interested reader is referred to [26] for a discussion of their security architecture.

(1) Authentication

When a user sends an RSVP PATH or RESV message, this message may include some information to authenticate the user. [7] describes how user and application information is embedded into the RSVP message (AUTH_DATA object) and how to protect it. A router receiving such a message can use this information to authenticate the client and forward the user or application information to the policy decision point (PDP). Optionally, the PDP itself can authenticate the user, which is described in the next section. To be able to authenticate the user, to verify the integrity, and to check for replays, the entire POLICY_DATA element has to be forwarded from the router to the PDP (e.g., by including the element into a COPS message). It is assumed, although not clearly specified in [7], that the INTEGRITY object within the POLICY_DATA element is sent to the PDP along with all other attributes.

* Certificate Verification

  Using the policy element as described in [7], it is not possible to provide a certificate revocation list or other information to prove the validity of the certificate inside the policy element. A specific mechanism for certificate verification is not discussed in [7] and hence a number of them can be used for this purpose. For certificate verification, the network element (a router or the policy decision point) that has to authenticate the user could frequently download certificate revocation lists or use a protocol like the Online Certificate Status Protocol (OCSP) [27] and the Simple Certificate Validation Protocol (SCVP) [28] to determine the current status of a digital certificate.

* User Authentication to the PDP

  This alternative authentication procedure uses the PDP to authenticate the user instead of the first-hop router. In Section 4.2.1 of [7], the choice is given for the user to obtain a session ticket either for the next hop router or for the PDP. As noted in the same section, the identity of the PDP or the next hop router is statically configured or dynamically retrieved. Subsequently, user authentication to the PDP is considered.

* Kerberos-based Authentication to the PDP

  If Kerberos is used to authenticate the user, then a session ticket for the PDP must be requested first. A user who roams

between different routers in the same administrative domain does not need to request a new service ticket, because the same PDP is likely to be used by most or all first-hop routers within the same administrative domain. This is different from the case in which a session ticket for a router has to be obtained and authentication to a router is required. The router therefore plays a passive role of simply forwarding the request to the PDP and executing the policy decision returned by the PDP. Appendix B describes one example of user-to-PDP authentication.

User authentication with the policy element provides only unilateral authentication, whereby the client authenticates to the router or to the PDP. If an RSVP message is sent to the user's host and public-key-based authentication is not used, then the message does not contain a certificate and digital signature. Hence, no mutual authentication can be assumed. In case of Kerberos, mutual authentication may be accomplished if the PDP or the router transmits a policy element with an INTEGRITY object computed with the session key retrieved from the Kerberos ticket, or if the Kerberos ticket included in the policy element is also used for the RSVP INTEGRITY object as described in Section 4.2. This procedure only works if a previous message was transmitted from the end host to the network and such key is already established. Reference [7] does not discuss this issue, and therefore there is no particular requirement for transmitting network-specific credentials back to the end-user's host.

(2) Integrity Protection

Integrity protection is applied separately to the RSVP message and the POLICY_DATA element, as shown in Figure 1. In case of a policy-ignorant node along the path, the RSVP INTEGRITY object and the INTEGRITY object inside the policy element terminate at different nodes. Basically, the same is true for the user credentials if they are verified at the policy decision point instead of the first hop router.

* Kerberos

If Kerberos is used to authenticate the user to the first hop router, then the session key included in the Kerberos ticket may be used to compute the INTEGRITY object of the policy element. It is the keyed message digest that provides the authentication. The existence of the Kerberos service ticket inside the AUTH_DATA object does not provide authentication or a guarantee of freshness for the receiving host.

Authentication and guarantee of freshness are provided by the keyed hash value of the INTEGRITY object inside the POLICY_DATA element. This shows that the user actively participated in the Kerberos protocol and was able to obtain the session key to compute the keyed message digest. The Authenticator used in the Kerberos V5 protocol provides similar functionality, but replay protection is based on timestamps (or on a sequence number if the optional seq-number field inside the Authenticator is used for KRB_PRIV/KRB_SAFE messages as described in Section 5.3.2 of [8]).

* Digital Signature

If public-key-based authentication is provided, then user authentication is accomplished with a digital signature. As explained in Section 3.3.3 of [7], the DIGITAL_SIGNATURE attribute must be the last attribute in the AUTH_DATA object, and the digital signature covers the entire AUTH_DATA object. In the case of PGP, which hash algorithm and public key algorithm are used for the digital signature computation is described in [19]. In the case of X.509 credentials, the situation is more complex because different mechanisms like CMS [29] or PKCS#7 [30] may be used for digitally signing the message element. X.509 only provides the standard for the certificate layout, which seems to provide insufficient information for this purpose. Therefore, X.509 certificates are supported, for example, by CMS or PKCS#7. [7], however, does not make any statements about the usage of CMS or PKCS#7. Currently, there is no support for CMS or for PKCS#7 [7], which provides more than just public-key-based authentication (e.g., CRL distribution, key transport, key agreement, etc.). Furthermore, the use of PGP in RSVP is vaguely defined, because there are different versions of PGP (including OpenPGP [19]), and no indication is given as to which should be used.

Supporting public-key-based mechanisms in RSVP might increase the risks of denial-of-service attacks. The large processing, memory, and bandwidth requirements should also be considered. Fragmentation might also be an issue here.

If the INTEGRITY object is not included in the POLICY_DATA element or not sent to the PDP, then we have to make the following observations:

For the digital signature case, only the replay protection provided by the digital signature algorithm can be used. It is not clear, however, whether this usage was anticipated or not. Hence, we might assume that replay

protection is based on the availability of the RSVP INTEGRITY object used with a security association that is established by other means.

Including only the Kerberos session ticket is insufficient, because freshness is not provided (because the Kerberos Authenticator is missing). Obviously there is no guarantee that the user actually followed the Kerberos protocol and was able to decrypt the received TGS_REP (or, in rare cases, the AS_REP if a session ticket is requested with the initial AS_REQ).

(3) Replay Protection

Figure 5 shows the interfaces relevant for replay protection of signaling messages in a more complicated architecture. In this case, the client uses the policy data element with PEP2, because PEP1 is not policy-aware. The interfaces between the client and PEP1 and between PEP1 and PEP2 are protected with the RSVP INTEGRITY object. The link between the PEP2 and the PDP is protected, for example, by using the COPS built-in INTEGRITY object. The dotted line between the Client and the PDP indicates the protection provided by the AUTH_DATA element, which has no RSVP INTEGRITY object included.

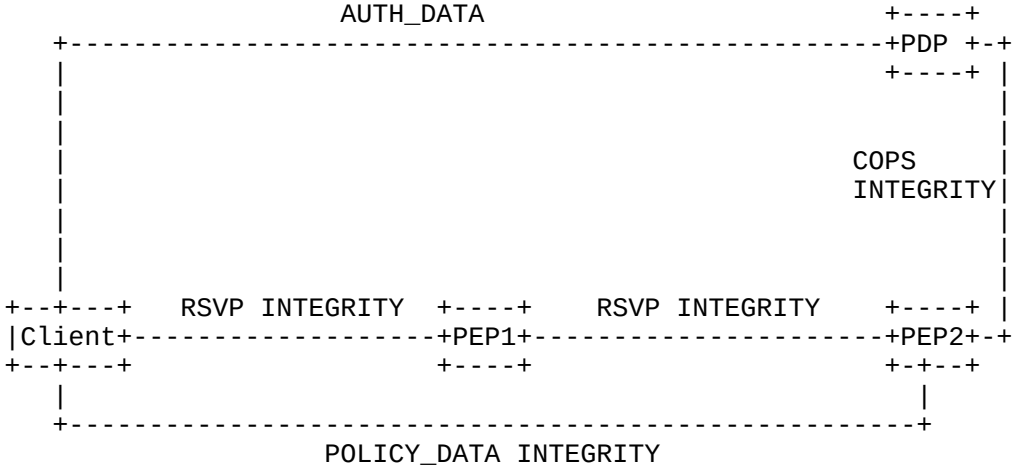

Figure 5: Replay Protection.

Host authentication with the RSVP INTEGRITY object and user authentication with the INTEGRITY object inside the POLICY_DATA element both use the same anti-replay mechanism. The length of

the Sequence Number field, sequence number rollover, and the Integrity Handshake have already been explained in Section 3.1.

Section 9 of [7] states: "RSVP INTEGRITY object is used to protect the policy object containing user identity information from security (replay) attacks." When using public-key-based authentication, RSVP-based replay protection is not supported, because the digital signature does not cover the POLICY_DATA INTEGRITY object with its Sequence Number field. The digital signature covers only the entire AUTH_DATA object.

The use of public key cryptography within the AUTH_DATA object complicates replay protection. Digital signature computation with PGP is described in [31] and in [19]. The data structure preceding the signed message digest includes information about the message digest algorithm used and a 32-bit timestamp of when the signature was created ("Signature creation time"). The timestamp is included in the computation of the message digest. The IETF standardized version of OpenPGP [19] contains more information and describes the different hash algorithms (MD2, MD5, SHA-1, RIPEMD-160) supported. [7] does not make any statements as to whether the "Signature creation time" field is used for replay protection. Using timestamps for replay protection requires different synchronization mechanisms in the case of clock-skew. Traditionally, these cases assume "loosely synchronized" clocks but also require specifying a replay window.

If the "Signature creation time" is not used for replay protection, then a malicious, policy-ignorant node can use this weakness to replace the AUTH_DATA object without destroying the digital signature. If this was not simply an oversight, it is therefore assumed that replay protection of the user credentials was not considered an important security requirement, because the hop-by-hop processing of the RSVP message protects the message against modification by an adversary between two communicating nodes.

The lifetime of the Kerberos ticket is based on the fields starttime and endtime of the EncTicketPart structure in the ticket, as described in Section 5.3.1 of [8]. Because the ticket is created by the KDC located at the network of the verifying entity, it is not difficult to have the clocks roughly synchronized for the purpose of lifetime verification. Additional information about clock-synchronization and Kerberos can be found in [32].

If the lifetime of the Kerberos ticket expires, then a new ticket must be requested and used. Rekeying is implemented with this procedure.

(4) (User Identity) Confidentiality

This section discusses privacy protection of identity information transmitted inside the policy element. User identity confidentiality is of particular interest because there is no built-in RSVP mechanism for encrypting the POLICY_DATA object or the AUTH_DATA elements. Encryption of one of the attributes inside the AUTH_DATA element, the POLICY_LOCATOR attribute, is discussed.

To protect the user's privacy, it is important not to reveal the user's identity to an adversary located between the user's host and the first-hop router (e.g., on a wireless link). Furthermore, user identities should not be transmitted outside the domain of the visited network provider. That is, the user identity information inside the policy data element should be removed or modified by the PDP to prevent revealing its contents to other (unauthorized) entities along the signaling path. It is not possible (with the offered mechanisms) to hide the user's identity in such a way that it is not visible to the first policy-aware RSVP node (or to the attached network in general).

The ASCII or Unicode distinguished name of the user or application inside the POLICY_LOCATOR attribute of the AUTH_DATA element may be encrypted as specified in Section 3.3.1 of [7]. The user (or application) identity is then encrypted with either the Kerberos session key or with the private key in case of public-key-based authentication. When the private key is used, we usually speak of a digital signature that can be verified by everyone possessing the public key. Because the certificate with the public key is included in the message itself, decryption is no obstacle. Furthermore, the included certificate together with the additional (unencrypted) information in the RSVP message provides enough identity information for an eavesdropper. Hence, the possibility of encrypting the policy locator in case of public-key-based authentication is problematic. To encrypt the identities using asymmetric cryptography, the user's host must be able somehow to retrieve the public key of the entity verifying the policy element (i.e., the first policy-aware router or the PDP). Then, this public key could be used to encrypt a symmetric key, which in turn encrypts the user's identity and certificate, as is done, e.g., by PGP. Currently, no such mechanism is defined in [7].

The algorithm used to encrypt the POLICY_LOCATOR with the Kerberos session key is assumed to be the same as the one used for encrypting the service ticket. The information about the algorithm used is available in the etype field of the EncryptedData ASN.1 encoded message part. Section 6.3 of [8] lists the supported algorithms. [33] defines newer encryption algorithms (Rijndael, Serpent, and Twofish).

Evaluating user identity confidentiality also requires looking at protocols executed outside of RSVP (for example, the Kerberos protocol). The ticket included in the CREDENTIAL attribute may provide user identity protection by not including the optional cname attribute inside the unencrypted part of the Ticket. Because the Authenticator is not transmitted with the RSVP message, the cname and the crealm of the unencrypted part of the Authenticator are not revealed. In order for the user to request the Kerberos session ticket for inclusion in the CREDENTIAL attribute, the Kerberos protocol exchange must be executed. Then the Authenticator sent with the TGS_REQ reveals the identity of the user. The AS_REQ must also include the user's identity to allow the Kerberos Authentication Server to respond with an AS_REP message that is encrypted with the user's secret key. Using Kerberos, it is therefore only possible to hide the content of the encrypted policy locator, which is only useful if this value differs from the Kerberos principal name. Hence, using Kerberos it is not "entirely" possible to provide user identity confidentiality.

It is important to note that information stored in the policy element may be changed by a policy-aware router or by the policy decision point. Which parts are changed depends upon whether multicast or unicast is used, how the policy server reacts, where the user is authenticated, whether the user needs to be re-authenticated in other network nodes, etc. Hence, user-specific and application-specific information can leak after the messages leave the first hop within the network where the user's host is attached. As mentioned at the beginning of this section, this information leakage is assumed to be intentional.

(5) Authorization

In addition to the description of the authorization steps of the Host-to-Router interface, user-based authorization is performed with the policy element providing user credentials. The inclusion of user and application specific information enables policy-based admission control with special user policies that are likely to be stored at a dedicated server. Hence, a Policy Decision Point can query, for example, an LDAP server for a

service level agreement that states the amount of resources a certain user is allowed to request. In addition to the user identity information, group membership and other non-security-related information may contribute to the evaluation of the final policy decision. If the user is not registered to the currently attached domain, then there is the question of how much information the home domain of the user is willing to exchange. This also impacts the user's privacy policy.

In general, the user may not want to distribute much of this policy information. Furthermore, the lack of a standardized authorization data format may create interoperability problems when exchanging policy information. Hence, we can assume that the policy decision point may use information from an initial authentication and key agreement protocol (which may have already required cross-realm communication with the user's home domain, if only to show that the home domain knows the user and that the user is entitled to roam), to forward accounting messages to this domain. This represents the traditional subscriber-based accounting scenario. Non-traditional or alternative means of access might be deployed in the near future that do not require any type of inter-domain communication.

Additional discussions are required to determine the expected authorization procedures. [34] and [35] discuss authorization issues for QoS signaling protocols. Furthermore, a number of mobility implications for policy handling in RSVP are described in [36].

(6) Performance

If Kerberos is used for user authentication, then a Kerberos ticket must be included in the CREDENTIAL Section of the AUTH_DATA element. The Kerberos ticket has a size larger than 500 bytes, but it only needs to be sent once because a performance optimization allows the session key to be cached as noted in Section 7.1 of [1]. It is assumed that subsequent RSVP messages only include the POLICY_DATA INTEGRITY object with a keyed message digest that uses the Kerberos session key. However, this assumes that the security association required for the POLICY_DATA INTEGRITY object is created (or modified) to allow the selection of the correct key. Otherwise, it difficult to say which identifier is used to index the security association.

If Kerberos is used as an authentication system then, from a performance perspective, the message exchange to obtain the session key needs to be considered, although the exchange only

needs to be done once in the lifetime of the session ticket. This is particularly true in a mobile environment with a fast roaming user's host.

Public-key-based authentication usually provides the best scalability characteristics for key distribution, but the protocols are performance demanding. A major disadvantage of the public-key-based user authentication in RSVP is the lack of a method to derive a session key. Hence, every RSVP PATH or RESV message includes the certificate and a digital signature, which is a huge performance and bandwidth penalty. For a mobile environment with low power devices, high latency, channel noise, and low-bandwidth links, this seems to be less encouraging. Note that a public key infrastructure is required to allow the PDP (or the first-hop router) to verify the digital signature and the certificate. To check for revoked certificates, certificate revocation lists or protocols like the Online Certificate Status Protocol [27] and the Simple Certificate Validation Protocol [28] are needed. Then the integrity of the AUTH_DATA object can be verified via the digital signature.

4.4. Communication between RSVP-Aware Routers

(1) Authentication

RSVP signaling messages have data origin authentication and are protected against modification and replay with the RSVP INTEGRITY object. The RSVP message flow between routers is protected based on the chain of trust, and hence each router needs only a security association with its neighboring routers. This assumption was made because of performance advantages and because of special security characteristics of the core network to which no user hosts are directly attached. In the core network the network structure does not change frequently and the manual distribution of shared secrets for the RSVP INTEGRITY object may be acceptable. The shared secrets may be either manually configured or distributed by using appropriately secured network management protocols like SNMPv3.

Independent of the key distribution mechanism, host authentication with built-in RSVP mechanisms is accomplished using the keyed message digest in the RSVP INTEGRITY object, computed using the previously exchanged symmetric key.

(2) Integrity Protection

Integrity protection is accomplished with the RSVP INTEGRITY object with the variable length Keyed Message Digest field.

(3) Replay Protection

Replay protection with the RSVP INTEGRITY object is extensively described in previous sections. To enable crashed hosts to learn the latest sequence number used, the Integrity Handshake mechanism is provided in RSVP.

(4) Confidentiality

Confidentiality is not provided by RSVP.

(5) Authorization

Depending on the RSVP network, QoS resource authorization at different routers may need to contact the PDP again. Because the PDP is allowed to modify the policy element, a token may be added to the policy element to increase the efficiency of the re-authorization procedure. This token is used to refer to an already computed policy decision. The communications interface from the PEP to the PDP must be properly secured.

(6) Performance

The performance characteristics for the protection of the RSVP signaling messages is largely determined by the key exchange protocol, because the RSVP INTEGRITY object is only used to compute a keyed message digest of the transmitted signaling messages.

The security associations within the core network, that is, between individual routers (in comparison with the security association between the user's host and the first-hop router or with the attached network in general), can be established more easily because of the normally strong trust assumptions. Furthermore, it is possible to use security associations with an increased lifetime to avoid frequent rekeying. Hence, there is less impact on the performance compared with the user-to-network interface. The security association storage requirements are also less problematic.

5. Miscellaneous Issues

This section describes a number of issues that illustrate some of the shortcomings of RSVP with respect to security.

5.1. First-Hop Issue

In case of end-to-end signaling, an end host starts signaling to its attached network. The first-hop communication is often more difficult to secure because of the different requirements and a missing trust relationship. An end host must therefore obtain some information to start RSVP signaling:

- Does this network support RSVP signaling?
- Which node supports RSVP signaling?
- To which node is authentication required?
- Which security mechanisms are used for authentication?
- Which algorithms are required?
- Where should the keys and security associations come from?
- Should a security association be established?

RSVP, as specified today, is used as a building block. Hence, these questions have to be answered as part of overall architectural considerations. Without answers to these questions, ad hoc RSVP communication by an end host roaming to an unknown network is not possible. A negotiation of security mechanisms and algorithms is not supported for RSVP.

5.2. Next-Hop Problem

Throughout the document it was assumed that the next RSVP node along the path is always known. Knowing the next hop is important to be able to select the correct key for the RSVP Integrity object and to apply the proper protection. In the case in which an RSVP node assumes it knows which node is the next hop, the following protocol exchange can occur:

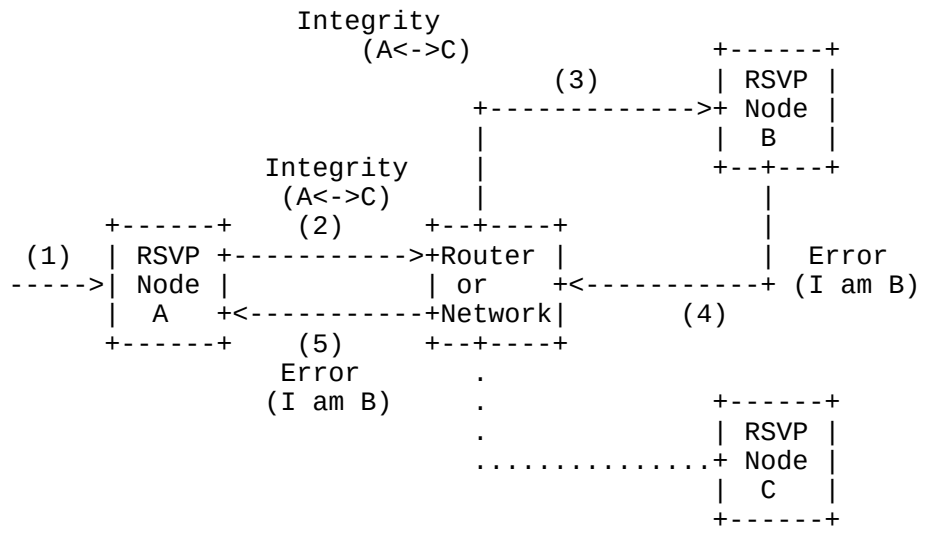

Figure 6: Next-Hop Issue.

When RSVP node A in Figure 6 receives an incoming RSVP Path message, standard RSVP message processing takes place. Node A then has to decide which key to select to protect the signaling message. We assume that some unspecified mechanism is used to make this decision. In this example, node A assumes that the message will travel to RSVP node C. However, for some reasons (e.g., a route change, inability to learn the next RSVP hop along the path, etc.) the message travels to node B via a non-RSVP supporting router that cannot verify the integrity of the message (or cannot decrypt the Kerberos service ticket). The processing failure causes a PathErr message to be returned to the originating sender of the Path message. This error message also contains information about the node that recognized the error. In many cases, a security association might not be available. Node A receiving the PathErr message might use the information returned with the PathErr message to select a different security association (or to establish one).

Figure 6 describes a behavior that might help node A learn that an error occurred. However, the description in Section 4.2 of [1] states in step (5) that a signaling message is silently discarded if the receiving host cannot properly verify the message: "If the calculated digest does not match the received digest, the message is discarded without further processing." For RSVP Path and similar messages, this functionality is not really helpful.

The RSVP Path message therefore provides a number of functions: path discovery, detecting route changes, discovery of QoS capabilities along the path using the Adspec object (with some interpretation), next-hop discovery, and possibly security association establishment (for example, in the case of Kerberos).

From a security point of view, there are conflicts between:

- Idempotent message delivery and efficiency

  The RSVP Path message especially performs a number of functions. Supporting idempotent message delivery somehow contradicts with security association establishment, efficient message delivery, and message size. For example, a "real" idempotent signaling message would contain enough information to perform security processing without depending on a previously executed message exchange. Adding a Kerberos ticket with every signaling message is, however, inefficient. Using public-key-based mechanisms is even more inefficient when included in every signaling message. With public-key-based protection for idempotent messages, there is the additional risk of introducing denial-of-service attacks.

- RSVP Path message functionality and next-hop discovery

  To protect an RSVP signaling message (and an RSVP Path message in particular) it is necessary to know the identity of the next RSVP-aware node (and some other parameters). Without a mechanism for next-hop discovery, an RSVP Path message is also responsible for this task. Without knowing the identity of the next hop, the Kerberos principal name is also unknown. The so-called Kerberos user-to-user authentication mechanism, which would allow the receiver to trigger the process of establishing Kerberos authentication, is not supported. This issue will again be discussed in relationship with the last-hop problem.

  It is fair to assume that an RSVP-supporting node might not have security associations with all immediately neighboring RSVP nodes. Especially for inter-domain signaling, IntServ over DiffServ, or some new applications such as firewall signaling, the next RSVP-aware node might not be known in advance. The number of next RSVP nodes might be considerably large if they are separated by a large number of non-RSVP aware nodes. Hence, a node transmitting an RSVP Path message might experience difficulties in properly protecting the message if it serves as a mechanism to detect both the next RSVP node (i.e., Router Alert Option added to the signaling message and addressed to the destination address) and to detect route changes. It is fair to note that, in the intra-

domain case with a dense distribution of RSVP nodes, protection might be possible with manual configuration.

Nothing prevents an adversary from continuously flooding an RSVP node with bogus PathErr messages, although it might be possible to protect the PathErr message with an existing, available security association. A legitimate RSVP node would believe that a change in the path took place. Hence, this node might try to select a different security association or try to create one with the indicated node. If an adversary is located somewhere along the path, and either authentication or authorization is not performed with the necessary strength and accuracy, then it might also be possible to act as a man-in-the-middle. One method of reducing susceptibility to this attack is as follows: when a PathErr message is received from a node with which no security association exists, attempt to establish a security association and then repeat the action that led to the PathErr message.

5.3. Last-Hop Issue

This section tries to address practical difficulties when authentication and key establishment are accomplished with a two-party protocol that shows some asymmetry in message processing. Kerberos is such a protocol and also the only supported protocol that provides dynamic session key establishment for RSVP. For first-hop communication, authentication is typically done between a user and some router (for example the access router). Especially in a mobile environment, it is not feasible to authenticate end hosts based on their IP or MAC address. To illustrate this problem, the typical processing steps for Kerberos are shown for first-hop communication:

(1) The end host A learns the identity (i.e., Kerberos principal name) of some entity B. This entity B is either the next RSVP node, a PDP, or the next policy-aware RSVP node.

(2) Entity A then requests a ticket granting ticket for the network domain. This assumes that the identity of the network domain is known.

(3) Entity A then requests a service ticket for entity B, whose name was learned in step (1).

(4) Entity A includes the service ticket with the RSVP signaling message (inside the policy object). The Kerberos session key is used to protect the integrity of the entire RSVP signaling message.

For last-hop communication, this processing theoretically has to be reversed: entity A is then a node in the network (for example, the access router) and entity B is the other end host (under the assumption that RSVP signaling is accomplished between two end hosts and not between an end host and an application server). However, the access router in step (1) might not be able to learn the user's principal name because this information might not be available. Entity A could reverse the process by triggering an IAKERB exchange. This would cause entity B to request a service ticket for A as described above. However, IAKERB is not supported in RSVP.

5.4. RSVP- and IPsec-Protected Data Traffic

QoS signaling requires flow information to be established at routers along a path. This flow identifier installed at each device tells the router which data packets should receive QoS treatment. RSVP typically establishes a flow identifier based on the 5-tuple (source IP address, destination IP address, transport protocol type, source port, and destination port). If this 5-tuple information is not available, then other identifiers have to be used. ESP-encrypted data traffic is such an example where the transport protocol and the port numbers are not accessible. Hence, the IPsec SPI is used as a substitute for them. [12] considers these IPsec implications for RSVP and is based on three assumptions:

(1) An end host that initiates the RSVP signaling message exchange has to be able to retrieve the SPI for a given flow. This requires some interaction with the IPsec security association database (SAD) and security policy database (SPD) [3]. An application usually does not know the SPI of the protected flow and cannot provide the desired values. It can provide the signaling protocol daemon with flow identifiers. The signaling daemon would then need to query the SAD by providing the flow identifiers as input parameters and receiving the SPI as an output parameter.

(2) [12] assumes end-to-end IPsec protection of the data traffic. If IPsec is applied in a nested fashion, then parts of the path do not experience QoS treatment. This can be treated as a problem of tunneling that is initiated by the end host. The following figure better illustrates the problem in the case of enforcing secure network access:

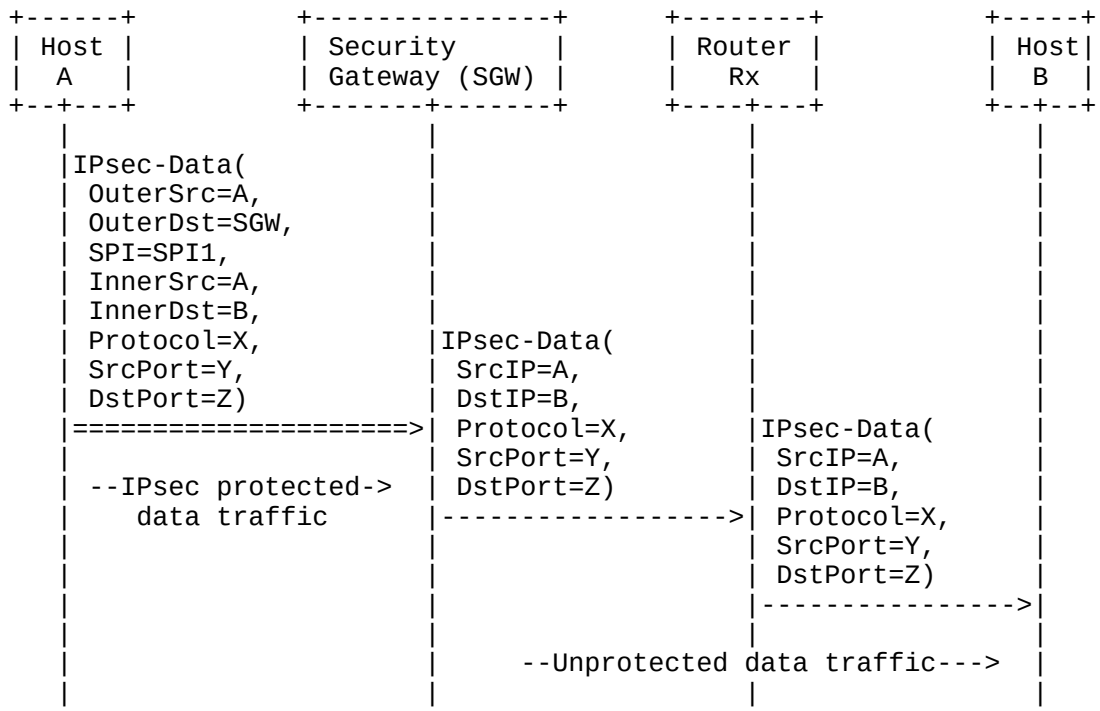

Figure 7: RSVP and IPsec protected data traffic.

Host A, transmitting data traffic, would either indicate a 3-tuple <A, SGW, SPI1> or a 5-tuple <A, B, X, Y, Z>. In any case, it is not possible to make a QoS reservation for the entire path. Two similar examples are remote access using a VPN and protection of data traffic between a home agent (or a security gateway in the home network) and a mobile node. The same problem occurs with a nested application of IPsec (for example, IPsec between A and SGW and between A and B).

One possible solution to this problem is to change the flow identifier along the path to capture the new flow identifier after an IPsec endpoint.

IPsec tunnels that neither start nor terminate at one of the signaling end points (for example between two networks) should be addressed differently by recursively applying an RSVP signaling exchange for the IPsec tunnel. RSVP signaling within tunnels is addressed in [13].

(3) It is assumed that SPIs do not change during the lifetime of the established QoS reservation. If a new IPsec SA is created, then

a new SPI is allocated for the security association. To reflect this change, either a new reservation has to be established or the flow identifier of the existing reservation has to be updated. Because IPsec SAs usually have a longer lifetime, this does not seem to be a major issue. IPsec protection of SCTP data traffic might more often require an IPsec SA (and SPI) change to reflect added and removed IP addresses from an SCTP association.

5.5. End-to-End Security Issues and RSVP

End-to-end security for RSVP has not been discussed throughout the document. In this context, end-to-end security refers to credentials transmitted between the two end hosts using RSVP. It is obvious that care must be taken to ensure that routers along the path are able to process and modify the signaling messages according to prescribed processing procedures. However, some objects or mechanisms could be used for end-to-end protection. The main question, however, is the benefit of such end-to-end security. First, there is the question of how to establish the required security association. Between two arbitrary hosts on the Internet, this might turn out to be quite difficult. Second, the usefulness of end-to-end security depends on the architecture in which RSVP is deployed. If RSVP is used only to signal QoS information into the network, and other protocols have to be executed beforehand to negotiate the parameters and to decide which entity is charged for the QoS reservation, then no end-to-end security is likely to be required. Introducing end-to-end security to RSVP would then cause problems with extensions like RSVP proxy [37], Localized RSVP [38], and others that terminate RSVP signaling somewhere along the path without reaching the destination end host. Such a behavior could then be interpreted as a man-in-the-middle attack.

5.6. IPsec Protection of RSVP Signaling Messages

It is assumed throughout that RSVP signaling messages can also be protected by IPsec [3] in a hop-by-hop fashion between two adjacent RSVP nodes. RSVP, however, uses special processing of signaling messages, which complicates IPsec protection. As explained in this section, IPsec should only be used for protection of RSVP signaling messages in a point-to-point communication environment (i.e., an RSVP message can only reach one RSVP router and not possibly more than one). This restriction is caused by the combination of signaling message delivery and discovery into a single message. Furthermore, end-to-end addressing complicates IPsec handling considerably. This section describes at least some of these complications.

RSVP messages are transmitted as raw IP packets with protocol number 46. It might be possible to encapsulate them in UDP as described in Appendix C of [6]. Some RSVP messages (Path, PathTear, and ResvConf) must have the Router Alert IP Option set in the IP header. These messages are addressed to the (unicast or multicast) destination address and not to the next RSVP node along the path. Hence, an IPsec traffic selector can only use these fields for IPsec SA selection. If there is only a single path (and possibly all traffic along it is protected) then there is no problem for IPsec protection of signaling messages. This type of protection is not common and might only be used to secure network access between an end host and its first-hop router. Because the described RSVP messages are addressed to the destination address instead of the next RSVP node, it is not possible to use IPsec ESP [17] or AH [16] in transport mode--only IPsec in tunnel mode is possible.

If an RSVP message can taket more than one possible path, then the IPsec engine will experience difficulties protecting the message. Even if the RSVP daemon installs a traffic selector with the destination IP address, still, no distinguishing element allows selection of the correct security association for one of the possible RSVP nodes along the path. Even if it possible to apply IPsec protection (in tunnel mode) for RSVP signaling messages by incorporating some additional information, there is still the possibility that the tunneled messages do not recognize a path change in a non-RSVP router. In this case the signaling messages would simply follow a different path than the data.

RSVP messages like RESV can be protected by IPsec, because they contain enough information to create IPsec traffic selectors that allow differentiation between various next RSVP nodes. The traffic selector would then contain the protocol number and the source and destination address pair of the two communicating RSVP nodes.

One benefit of using IPsec is the availability of key management using either IKE [39], KINK [40] or IKEv2 [41].

5.7. Authorization

[34] describes two trust models (NJ Turnpike and NJ Parkway) and two authorization models (per-session and per-channel financial settlement). The NJ Turnpike model gives a justification for hop-by-hop security protection. RSVP focuses on the NJ Turnpike model, although the different trust models are not described in detail. RSVP supports the NJ Parkway model and per-channel financial settlement only to a certain extent. Authentication of the user (or end host) can be provided with the user identity representation

mechanism, but authentication might, in many cases, be insufficient for authorization. The communication procedures defined for policy

objects [42] can be improved to support the more efficient per-channel financial settlement model by avoiding policy handling between inter-domain networks at a signaling message granularity. Additional information about expected behavior of policy handling in RSVP can also be obtained from [43].

[35] and [36] provide additional information on authorization. No good and agreed mechanism for dealing with authorization of QoS reservations in roaming environments is provided. Price distribution mechanisms are only described in papers and never made their way through standardization. RSVP focuses on receiver-initiated reservations with authorization for the QoS reservation by the data receiver, which introduces a fair amount of complexity for mobility handling as described, for example, in [36].

6. Conclusions

RSVP was the first QoS signaling protocol that provided some security protection. Whether RSVP provides appropriate security protection heavily depends on the environment where it is deployed. RSVP as specified today should be viewed as a building block that has to be adapted to a given architecture.

This document aims to provide more insight into the security of RSVP. It cannot be interpreted as a pass or fail evaluation of the security provided by RSVP.

Certainly this document is not a complete description of all security issues related to RSVP. Some issues that require further consideration are RSVP extensions (for example [12]), multicast issues, and other security properties like traffic analysis. Additionally, the interaction with mobility protocols (micro- and macro-mobility) demands further investigation from a security point of view.

What can be learned from practical protocol experience and from the increased awareness regarding security is that some of the available credential types have received more acceptance than others. Kerberos is a system that is integrated into many IETF protocols today. Public-key-based authentication techniques are, however, still considered to be too heavy-weight (computationally and from a bandwidth perspective) to be used for per-flow signaling. The increased focus on denial of service attacks puts additional demands on the design of public-key-based authentication.

The following list briefly summarizes a few security or architectural issues that deserve improvement:

- Discovery and signaling message delivery should be separated.
- For some applications and scenarios, it cannot be assumed that neighboring RSVP-aware nodes know each other. Hence, some in-path discovery mechanism should be provided.
- Addressing for signaling messages should be done in a hop-by-hop fashion.
- Standard security protocols (IPsec, TLS, or CMS) should be used whenever possible. Authentication and key exchange should be separated from signaling message protection. In general, it is necessary to provide key management to establish security associations dynamically for signaling message protection. Relying on manually configured keys between neighboring RSVP nodes is insufficient. A separate, less frequently executed key management and security association establishment protocol is a good place to perform entity authentication, security service negotiation and selection, and agreement on mechanisms, transforms, and options.
- The use of public key cryptography in authorization tokens, identity representations, selective object protection, etc. is likely to cause fragmentation, the need to protect against denial of service attacks, and other problems.
- Public key authentication and user identity confidentiality provided with RSVP require some improvement.
- Public-key-based user authentication only provides entity authentication. An additional security association is required to protect signaling messages.
- Data origin authentication should not be provided by non-RSVP nodes (such as the PDP). Such a procedure could be accomplished by entity authentication during the authentication and key exchange phase.
- Authorization and charging should be better integrated into the base protocol.
- Selective message protection should be provided. A protected message should be recognizable from a flag in the header.

- o Confidentiality protection is missing and should therefore be added to the protocol. The general principle is that protocol designers can seldom foresee all of the environments in which protocols will be run, so they should allow users to select from a full range of security services, as the needs of different user communities vary.

- o Parameter and mechanism negotiation should be provided.

7. Security Considerations

This document discusses security properties of RSVP and, as such, it is concerned entirely with security.

8. Acknowledgements

We would like to thank Jorge Cuellar, Robert Hancock, Xiaoming Fu, Guenther Schaefer, Marc De Vuyst, Bob Grillo, and Jukka Manner for their comments. Additionally, Hannes would like to thank Robert and Jorge for their time discussing various issues.

Finally, we would like to thank Allison Mankin and John Loughney for their guidance and input.

9. References

9.1. Normative References

[1] Baker, F., Lindell, B., and M. Talwar, "RSVP Cryptographic Authentication", RFC 2747, January 2000.

[2] Herzog, S., "RSVP Extensions for Policy Control", RFC 2750, January 2000.

[3] Kent, S. and R. Atkinson, "Security Architecture for the Internet Protocol", RFC 2401, November 1998.

[4] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, February 1997.

[5] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, April 1992.

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

[7] Yadav, S., Yavatkar, R., Pabbati, R., Ford, P., Moore, T., Herzog, S., and R. Hess, "Identity Representation for RSVP", RFC 3182, October 2001.

[8] Kohl, J. and C. Neuman, "The Kerberos Network Authentication Service (V5)", RFC 1510, September 1993. Obsoleted by RFC 4120.

[9] Calhoun, P., Loughney, J., Guttman, E., Zorn, G., and J. Arkko, "Diameter Base Protocol", RFC 3588, September 2003.

[10] Durham, D., Boyle, J., Cohen, R., Herzog, S., Rajan, R., and A. Sastry, "The COPS (Common Open Policy Service) Protocol", RFC 2748, January 2000.

[11] Herzog, S., Boyle, J., Cohen, R., Durham, D., Rajan, R., and A. Sastry, "COPS usage for RSVP", RFC 2749, January 2000.

[12] Berger, L. and T. O'Malley, "RSVP Extensions for IPSEC Data Flows", RFC 2207, September 1997.

[13] Terzis, A., Krawczyk, J., Wroclawski, J., and L. Zhang, "RSVP Operation Over IP Tunnels", RFC 2746, January 2000.

9.2. Informative References

[14] Hess, R. and S. Herzog, "RSVP Extensions for Policy Control", Work in Progress, June 2001.

[15] "Secure Hash Standard, NIST, FIPS PUB 180-1", Federal Information Processing Society, April 1995.

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

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

[18] Fowler, D., "Definitions of Managed Objects for the DS1, E1, DS2 and E2 Interface Types", RFC 2495, January 1999.

[19] Callas, J., Donnerhacke, L., Finney, H., and R. Thayer, "OpenPGP Message Format", RFC 2440, November 1998.

[20] Hornstein, K. and J. Altman, "Distributing Kerberos KDC and Realm Information with DNS", Work in Progress, July 2002.

[21] Dobbertin, H., Bosselaers, A., and B. Preneel, "RIPEMD-160: A strengthened version of RIPEMD in Fast Software Encryption", LNCS vol. 1039, pp. 71-82, 1996.

[22] Dobbertin, H., "The Status of MD5 After a Recent Attack", RSA Laboratories CryptoBytes, vol. 2, no. 2, 1996.

[23] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, "Extensible Authentication Protocol (EAP)", RFC 3748, June 2004.

[24] Rigney, C., Willens, S., Rubens, A., and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, June 2000.

[25] "Microsoft Authorization Data Specification v. 1.0 for Microsoft Windows 2000 Operating Systems", April 2000.

[26] Cable Television Laboratories, Inc., "PacketCable Security Specification, PKT-SP-SEC-I01-991201", website: http://www.PacketCable.com/, June 2003.

[27] Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", RFC 2560, June 1999.

[28] Malpani, A., Housley, R., and T. Freeman, "Simple Certificate Validation Protocol (SCVP)", Work in Progress, October 2005.

[29] Housley, R., "Cryptographic Message Syntax (CMS)", RFC 3369, August 2002.

[30] Kaliski, B., "PKCS #7: Cryptographic Message Syntax Version 1.5", RFC 2315, March 1998.

[31] "Specifications and standard documents", website: http://www.PacketCable.com/, March 2002.

[32] Davis, D. and D. Geer, "Kerberos With Clocks Adrift: History, Protocols and Implementation", USENIX Computing Systems, vol 9 no. 1, Winter 1996.

[33] Raeburn, K., "Encryption and Checksum Specifications for Kerberos 5", RFC 3961, February 2005.

[34] Tschofenig, H., Buechli, M., Van den Bosch, S., and H. Schulzrinne, "NSIS Authentication, Authorization and Accounting Issues", Work in Progress, March 2003.

[35] Tschofenig, H., Buechli, M., Van den Bosch, S., Schulzrinne, H., and T. Chen, "QoS NSLP Authorization Issues", Work in Progress, June 2003.

[36] Thomas, M., "Analysis of Mobile IP and RSVP Interactions", Work in Progress, October 2002.

[37] Gai, S., Gaitonde, S., Elfassy, N., and Y. Bernet, "RSVP Proxy", Work in Progress, March 2002.

[38] Manner, J., Suihko, T., Kojo, M., Liljeberg, M., and K. Raatikainen, "Localized RSVP", Work in Progress, September 2004.

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

[40] Thomas, M., "Kerberized Internet Negotiation of Keys (KINK)", Work in Progress, October 2005.

[41] Kaufman, C., "Internet Key Exchange (IKEv2) Protocol", RFC 4306, November 2005.

[42] Herzog, S., "Accounting and Access Control in RSVP", PhD Dissertation, USC, Work in Progress, November 1995.

[43] Herzog, S., "Accounting and Access Control for Multicast Distributions: Models and Mechanisms", June 1996.

[44] Pato, J., "Using Pre-Authentication to Avoid Password Guessing Attacks", Open Software Foundation DCE Request for Comments, December 1992.

[45] Tung, B. and L. Zhu, "Public Key Cryptography for Initial Authentication in Kerberos", Work in Progress, November 2005.

[46] Wu, T., "A Real-World Analysis of Kerberos Password Security", in Proceedings of the 1999 Internet Society Network and Distributed System Security Symposium, San Diego, February 1999.

[47] Wu, T., Wu, F., and F. Gong, "Securing QoS: Threats to RSVP Messages and Their Countermeasures", IEEE IWQoS, pp. 62-64, 1999.

[48] Talwar, V., Nahrstedt, K., and F. Gong, "Securing RSVP For Multimedia Applications", Proc ACM Multimedia 2000 (Multimedia Security Workshop), November 2000.

[49] Talwar, V., Nahrstedt, K., and S. Nath, "RSVP-SQoS: A Secure RSVP Protocol", International Conf on Multimedia and Exposition, Tokyo, Japan, August 2001.

[50] Jablon, D., "Strong Password-only Authenticated Key Exchange", ACM Computer Communication Review, 26(5), pp. 5-26, October 1996.

Appendix A. Dictionary Attacks and Kerberos

Kerberos might be used with RSVP as described in this document. Because dictionary attacks are often mentioned in relationship with Kerberos, a few issues are addressed here.

The initial Kerberos AS_REQ request (without pre-authentication, without various extensions, and without PKINIT) is unprotected. The response message AS_REP is encrypted with the client's long-term key. An adversary can take advantage of this fact by requesting AS_REP messages to mount an off-line dictionary attack. Pre-authentication ([44]) can be used to reduce this problem. However, pre-authentication does not entirely prevent dictionary attacks by an adversary who can still eavesdrop on Kerberos messages along the path between a mobile node and a KDC. With mandatory pre-authentication for the initial request, an adversary cannot request a Ticket Granting Ticket for an arbitrary user. On-line password guessing attacks are still possible by choosing a password (e.g., from a dictionary) and then transmitting an initial request that includes a pre-authentication data field. An unsuccessful authentication by the KDC results in an error message and thus gives the adversary a hint to restart the protocol and try a new password.

There are, however, some proposals that prevent dictionary attacks. The use of Public Key Cryptography for initial authentication [45] (PKINIT) is one such solution. Other proposals use strong-password-based authenticated key agreement protocols to protect the user's password during the initial Kerberos exchange. [46] discusses the security of Kerberos and also discusses mechanisms to prevent dictionary attacks.

Appendix B. Example of User-to-PDP Authentication

The following Section describes an example of user-to-PDP authentication. Note that the description below is not fully covered by the RSVP specification and hence it should only be viewed as an example.

Windows 2000, which integrates Kerberos into RSVP, uses a configuration with the user authentication to the PDP as described in [25]. The steps for authenticating the user to the PDP in an intra-realm scenario are the following:

- Windows 2000 requires the user to contact the KDC and to request a Kerberos service ticket for the PDP account AcsService in the local realm.

- This ticket is then embedded into the AUTH_DATA element and included in either the PATH or the RESV message. In the case of Microsoft's implementation, the user identity encoded as a distinguished name is encrypted with the session key provided with the Kerberos ticket. The Kerberos ticket is sent without the Kerberos authdata element that contains authorization information, as explained in [25].

- The RSVP message is then intercepted by the PEP, which forwards it to the PDP. [25] does not state which protocol is used to forward the RSVP message to the PDP.

- The PDP that finally receives the message and decrypts the received service ticket. The ticket contains the session key used by the user's host to

  * Encrypt the principal name inside the policy locator field of the AUTH_DATA object and to

  * Create the integrity-protected Keyed Message Digest field in the INTEGRITY object of the POLICY_DATA element. The protection described here is between the user's host and the PDP. The RSVP INTEGRITY object on the other hand is used to protect the path between the user's host and the first-hop router, because the two message parts terminate at different nodes, and different security associations must be used. The interface between the message-intercepting, first-hop router and the PDP must be protected as well.

  * The PDP does not maintain a user database, and [25] describes how the PDP may query the Active Directory (a LDAP based directory service) for user policy information.

Appendix C. Literature on RSVP Security

Few documents address the security of RSVP signaling. This section briefly describes some important documents.

Improvements to RSVP are proposed in [47] to deal with insider attacks. Insider attacks are caused by malicious RSVP routers that modify RSVP signaling messages in such a way that they cause harm to the nodes participating in the signaling message exchange.

As a solution, non-mutable RSVP objects are digitally signed by the sender. This digital signature is added to the RSVP PATH message. Additionally, the receiver attaches an object to the RSVP RESV message containing a "signed" history. This value allows

intermediate RSVP routers (by examining the previously signed value) to detect a malicious RSVP node.

A few issues are, however, left open in this document. Replay attacks are not covered, and it is therefore assumed that timestamp-based replay protection is used. To identify a malicious node, it is necessary that all routers along the path are able to verify the digital signature. This may require a global public key infrastructure and also client-side certificates. Furthermore, the bandwidth and computational requirements to compute, transmit, and verify digital signatures for each signaling message might place a burden on a real-world deployment.

Authorization is not considered in the document, which might have an influence on the implications of signaling message modification. Hence, the chain-of-trust relationship (or this step in a different direction) should be considered in relationship with authorization.

In [48], the above-described idea of detecting malicious RSVP nodes is improved by addressing performance aspects. The proposed solution is somewhere between hop-by-hop security and the approach in [47], insofar as it separates the end-to-end path into individual networks. Furthermore, some additional RSVP messages (e.g., feedback messages) are introduced to implement a mechanism called "delayed integrity checking." In [49], the approach presented in [48] is enhanced.

Authors' Addresses

Hannes Tschofenig
Siemens
Otto-Hahn-Ring 6
Munich, Bavaria 81739
Germany

EMail: Hannes.Tschofenig@siemens.com

Richard Graveman
RFG Security
15 Park Avenue
Morristown, NJ 07960
USA

EMail: rfg@acm.org

Full Copyright Statement

Intellectual Property

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.