Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 4, 2007

J. Manner
TKK
M. Stiemerling
NEC
H. Tschofenig
Siemens Networks GmbH & Co KG

March 3, 2007

Authorization for NSIS Signaling Layer Protocols
draft-manner-nsis-nslp-auth-03.txt

Status of this Memo

Copyright Notice

Abstract

Signaling layer protocols in the NSIS working group may rely on GIST to handle authorization. Still, the signaling layer protocol itself may require separate authorization to be performed when a node receives a request for a certain kind of service or resources. This draft presents a generic model and object formats for session authorization within the NSIS Signaling Layer Protocols. The goal of session authorization is to allow the exchange of information between network elements in order to authorize the use of resources for a service and to coordinate actions between the signaling and transport planes.

Table of Contents

1. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14, RFC 2119 [RFC2119].

The term "NSLP node" (NN) is used to refer to an NSIS node running an NSLP protocol that can make use of the authorization object discussed in this document. Currently, this node would run either the QoS or the NAT/FW NSLP service.

2. Introduction

The NSIS working group is specifying a suite of protocols for the next generation in Internet signaling [RFC4080]. The design is based on a generalized transport protocol for signaling applications, the General Internet Signaling Transport (GIST) [I-D.ietf-nsis-ntlp], and various kinds of signaling applications. Two signaling applications and their NSIS Signaling Layer Protocols (NSLP) have been designed, a Quality of Service application (QoS NSLP) [I-D.ietf-nsis-qos-nslp] and a NAT/firewall application (NAT/FW) [I-D.ietf-nsis-nslp-natfw].

The security architecture is based on a chain-of-trust model, where each GIST hop may chose the appropriate security protocol, taking into account the signaling application requirements. This model is appropriate for a number of different use cases, and allows the signaling applications to leave the handling of security to GIST.

Yet, in order to allow for finer-grain per-session admission control, it is necessary to provide a mechanism for ensuring that the use of resources by a host has been properly authorized before allowing the signaling application to commit the resource request, e.g., a QoS reservation or mappings for NAT traversal. In order to meet this

requirement,there must be information in the NSLP message which may be used to verify the validity of the request. This can be done by providing the host with a session authorization policy element which is inserted into the message and verified by the network.

This document describes a generic NSLP layer session authorization policy object (AUTH_SESSION) used to convey authorization information for the request. The scheme is based on third-party tokens. A trusted third party provides authentication tokens to clients and allows verification of the information by the network elements. The requesting host inserts its authorization information acquired from the trusted third party into the NSLP message to allow verification of the network resource request. Network elements verify the request and then process the resource reservation message based on admission policy. This work is based on RFC 3520 [RFC3520] and RFC 3521 [RFC3521].

The default operation of the authorization is to add one authorization policy object. Yet, in order to support end-to-end signaling and request authorization from different networks, a host initiating an NSLP signaling session may add more than one AUTH_SESSION object in the message. The identifier of the authorizing entity can be used by the network elements to use the third party they trust to verify the request.

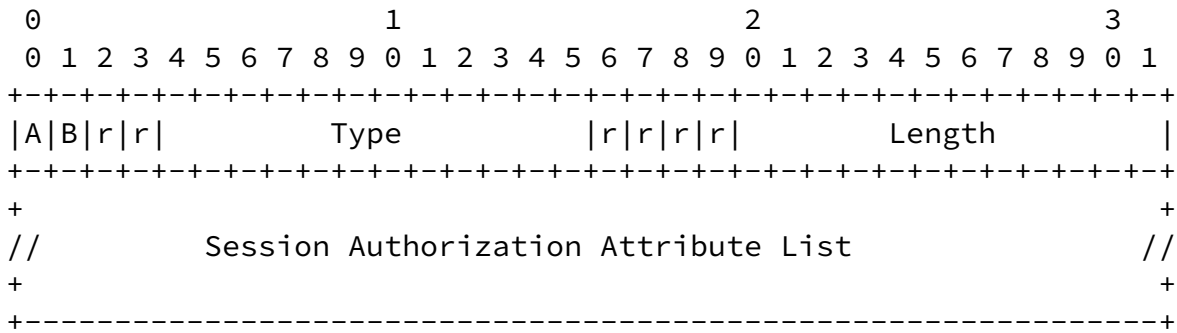
3. Session Authorization Object

This section presents a new NSLP layer object called session authorization (AUTH_SESSION). The AUTH_SESSION object can be used in the currently specified and future NSLP protocols.

The authorization attributes follow the format and specification given in RFC3520 [RFC3520].

3.1. Session Authorization Object format

The AUTH_SESSION object contains a list of fields which describe the session, along with other attributes. The object header follows the generic NSLP object header.



The value for the Type field comes from shared NSLP object type space. The Length field is given in units of 32 bit words and measures the length of the Value component of the TLV object (i.e. it does not include the standard header).

The bits marked 'A' and 'B' are extensibility flags, and used to signal the desired treatment for objects whose treatment has not been defined in the protocol specification (i.e. whose Type field is unknown at the receiver). The following four categories of object have been identified, and are described here.

AB=00 ("Mandatory"): If the object is not understood, the entire message containing it MUST be rejected with a "Object Type Error" message with subcode 1 ("Unrecognised Object"). In the NATFW NSLP case it MUST be rejected with an error response of class 'Protocol error' (0x3) with error code 'Unknown object present' (0x06).

AB=01 ("Ignore"): If the object is not understood, it MUST be deleted and the rest of the message processed as usual.

AB=10 ("Forward"): If the object is not understood, it MUST be

retained unchanged in any message forwarded as a result of message processing, but not stored locally.

AB=11 ("Refresh"): If the object is not understood, it should be incorporated into the locally stored signaling application state for this flow/session, forwarded in any resulting message, and also used in any refresh or repair message which is generated locally. In the NATFW NSLP this combination AB=11 MUST NOT be used and an error

response of class 'Protocol error' (0x3) with error code 'Invalid Flag-Field combination' (0x09) MUST be generated.

The remaining bits marked 'r' are reserved. The extensibility flags follow the definition in the GIST specification. The AUTH_SESSION object defines in this specification MUST have the AB-bits set to "10". An NN may use the authorization information if it is configured to do so, but may also just skip the object.

Type: 0x0a (TBD by IANA)

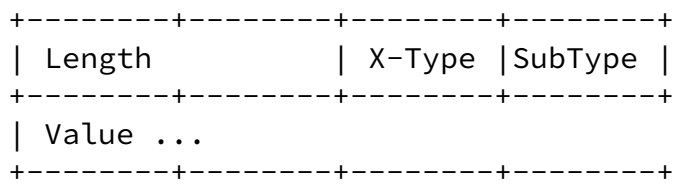
Length: Variable

Session Authorization Attribute List: variable length

The session authorization attribute list is a collection of objects which describes the session and provides other information necessary to verify the resource reservation request. An initial set of valid objects is described in Section 3.2.

3.2. Session Authorization Attributes

A session authorization attribute may contain a variety of information and has both an attribute type and subtype. The attribute itself MUST be a multiple of 4 octets in length, and any attributes that are not a multiple of 4 octets long MUST be padded to a 4-octet boundary. All padding bytes MUST have a value of zero.



Length: 16 bits

The length field is two octets and indicates the actual length of the attribute (including Length, X-Type and SubType fields) in number of octets. The length does NOT include any bytes padding to the value field to make the attribute a multiple of 4 octets

long.

X-Type: 8 bits

Session authorization attribute type (X-Type) field is one octet. IANA acts as a registry for X-Types as described in Section 7, IANA Considerations. Initially, the registry contains the following X-Types:

1. AUTH_ENT_ID The unique identifier of the entity which authorized the session.

2. SOURCE_ADDR Address specification for the session originator.

3. DEST_ADDR Address specification for the session end-point.

4. START_TIME The starting time for the session.

5. END_TIME The end time for the session.

6. AUTHENTICATION_DATA Authentication data of the session authorization policy element.

SubType: 8 bits

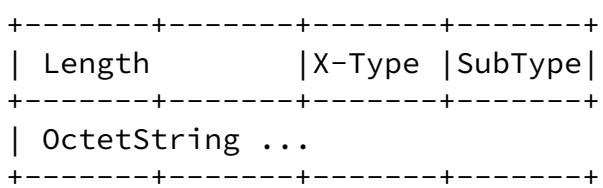
Session authorization attribute sub-type is one octet in length. The value of the SubType depends on the X-Type.

Value: variable length

The attribute specific information.

3.2.1. Authorizing Entity Identifier

AUTH_ENT_ID is used to identify the entity which authorized the initial service request and generated the session authorization policy element. The AUTH_ENT_ID may be represented in various formats, and the SubType is used to define the format for the ID. The format for AUTH_ENT_ID is as follows:



Length: Length of the attribute, which MUST be > 4.

X-Type: AUTH_ENT_ID

SubType:

The following sub-types for AUTH_ENT_ID are defined. IANA acts as a registry for AUTH_ENT_ID sub-types as described in Section 7, IANA Considerations. Initially, the registry contains the following sub-types of AUTH_ENT_ID:

1. IPV4_ADDRESS IPv4 address represented in 32 bits

2. IPV6_ADDRESS IPv6 address represented in 128 bits

3. FQDN Fully Qualified Domain Name as defined in RFC 1034 as an ASCII string.

4. ASCII_DN X.500 Distinguished name as defined in RFC 2253 as an ASCII string.

5. UNICODE_DN X.500 Distinguished name as defined in RFC 2253 as a UTF-8 string.

6. URI Universal Resource Identifier, as defined in RFC 2396.

7. KRB_PRINCIPAL Fully Qualified Kerberos Principal name represented by the ASCII string of a principal followed by the @ realm name as defined in RFC 1510 (e.g., johndoe@nowhere).

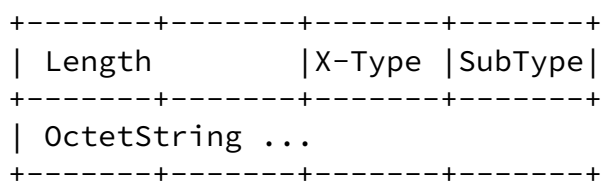
8. X509_V3_CERT The Distinguished Name of the subject of the certificate as defined in RFC 2253 as a UTF-8 string.

9. PGP_CERT The PGP digital certificate of the authorizing entity as defined in RFC 2440.

OctetString: Contains the authorizing entity identifier.

3.2.2. Source Address

SOURCE_ADDR is used to identify the source address specification of the authorized session. This X-Type may be useful in some scenarios to make sure the resource request has been authorized for that particular source address and/or port.



Length: Length of the attribute, which MUST be > 4.

X-Type: SOURCE_ADDR

SubType:

The following sub types for SOURCE_ADDR are defined. IANA acts as a registry for SOURCE_ADDR sub-types as described in Section 7, IANA Considerations. Initially, the registry contains the following sub types for SOURCE_ADDR:

1. IPV4_ADDRESS IPv4 address represented in 32 bits

2. IPV6_ADDRESS IPv6 address represented in 128 bits

3. UDP_PORT_LIST list of UDP port specifications, represented as 16 bits per list entry.

4. TCP_PORT_LIST list of TCP port specifications, represented as 16 bits per list entry.

5. SPI Security Parameter Index represented in 32 bits

OctetString: The OctetString contains the source address information.

In scenarios where a source address is required (see Section 5), at least one of the subtypes 1 or 2 MUST be included in every Session Authorization Data Policy Element. Multiple SOURCE_ADDR attributes MAY be included if multiple addresses have been authorized. The source address of the request (e.g., a QoS NSLP RESERVE) MUST match one of the SOURCE_ADDR attributes contained in this Session Authorization Data Policy Element.

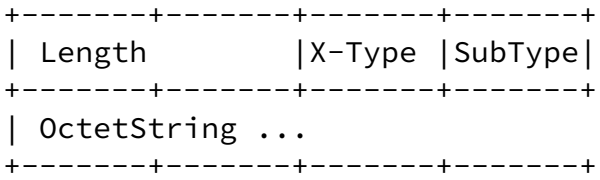
At most, one instance of subtype 3 MAY be included in every Session Authorization Data Policy Element. At most, one instance of subtype 4 MAY be included in every Session Authorization Data Policy Element. Inclusion of a subtype 3 attribute does not prevent inclusion of a subtype 4 attribute (i.e., both UDP and TCP ports may be authorized).

If no PORT attributes are specified, then all ports are considered valid; otherwise, only the specified ports are authorized for use.

Every source address and port list must be included in a separate SOURCE_ADDR attribute.

3.2.3. Destination Address

DEST_ADDR is used to identify the destination address of the authorized session. This X-Type may be useful in some scenarios to make sure the resource request has been authorized for that particular destination address and/or port.



Length: Length of the attribute, which MUST be > 4.

X-Type: DEST_ADDR

SubType:

The following sub types for DEST_ADDR are defined. IANA acts as a registry for DEST_ADDR sub-types as described in Section 7, IANA Considerations. Initially, the registry contains the following sub types for DEST_ADDR:

1. IPV4_ADDRESS IPv4 address represented in 32 bits

2. IPV6_ADDRESS IPv6 address represented in 128 bits

3. UDP_PORT_LIST list of UDP port specifications, represented as 16 bits per list entry.

4. TCP_PORT_LIST list of TCP port specifications, represented as 16 bits per list entry.

5. SPI Security Parameter Index represented in 32 bits

OctetString: The OctetString contains the destination address specification.

In scenarios where a destination address is required (see Section 5), at least one of the subtypes 1 or 2 MUST be included in every Session Authorization Data Policy Element. Multiple DEST_ADDR attributes MAY be included if multiple addresses have been authorized. The

destination address field of the resource reservation datagram (e.g., RSVP PATH) MUST match one of the DEST_ADDR attributes contained in this Session Authorization Data Policy Element.

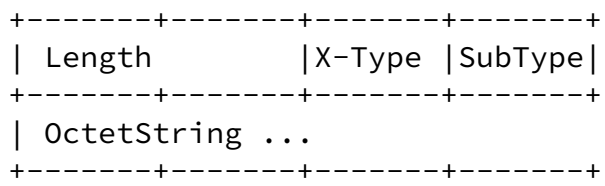
At most, one instance of subtype 3 MAY be included in every Session Authorization Data Policy Element. At most, one instance of subtype 4 MAY be included in every Session Authorization Data Policy Element. Inclusion of a subtype 3 attribute does not prevent inclusion of a subtype 4 attribute (i.e., both UDP and TCP ports may be authorized).

If no PORT attributes are specified, then all ports are considered valid; otherwise, only the specified ports are authorized for use.

Every destination address and port list must be included in a separate DEST_ADDR attribute.

3.2.4. Start time

START_TIME is used to identify the start time of the authorized session and can be used to prevent replay attacks. If the AUTH_SESSION policy element is presented in a resource request, the network SHOULD reject the request if it is not received within a few seconds of the start time specified.



Length: Length of the attribute, which MUST be > 4.

X-Type: START_TIME

SubType:

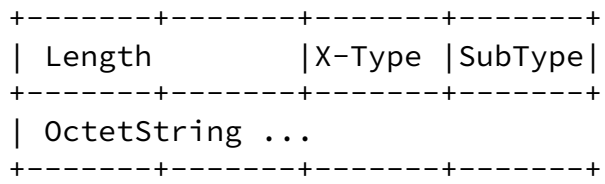
The following sub types for START_TIME are defined. IANA acts as a registry for START_TIME sub-types as described in Section 7, IANA Considerations. Initially, the registry contains the following sub types for START_TIME:

1. 1 NTP_TIMESTAMP NTP Timestamp Format as defined in RFC 1305.

OctetString: The OctetString contains the start time.

3.2.5. End time

END_TIME is used to identify the end time of the authorized session and can be used to limit the amount of time that resources are authorized for use (e.g., in prepaid session scenarios).



Length: Length of the attribute, which MUST be > 4.

X-Type: END_TIME

SubType:

The following sub types for END_TIME are defined. IANA acts as a registry for END_TIME sub-types as described in Section 7, IANA Considerations. Initially, the registry contains the following sub types for END_TIME:

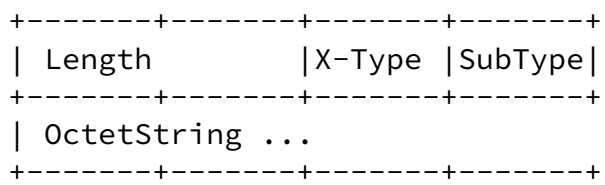
1. NTP_TIMESTAMP NTP Timestamp Format as defined in RFC 1305.

OctetString: The OctetString contains the end time.

3.2.6. Authentication data

The AUTHENTICATION_DATA attribute contains the authentication data of the AUTH_SESSION policy element and signs all the data in the policy element up to the AUTHENTICATION_DATA. If the AUTHENTICATION_DATA attribute has been included in the AUTH_SESSION policy element, it MUST be the last attribute in the list. The algorithm used to compute the authentication data depends on the AUTH_ENT_ID SubType field. See Section 4 entitled Integrity of the AUTH_SESSION policy element.

A summary of AUTHENTICATION_DATA attribute format is described below.



Length: Length of the attribute, which MUST be > 4.

X-Type: AUTHENTICATION_DATA

SubType: No sub types for AUTHENTICATION_DATA are currently defined. This field MUST be set to 0.

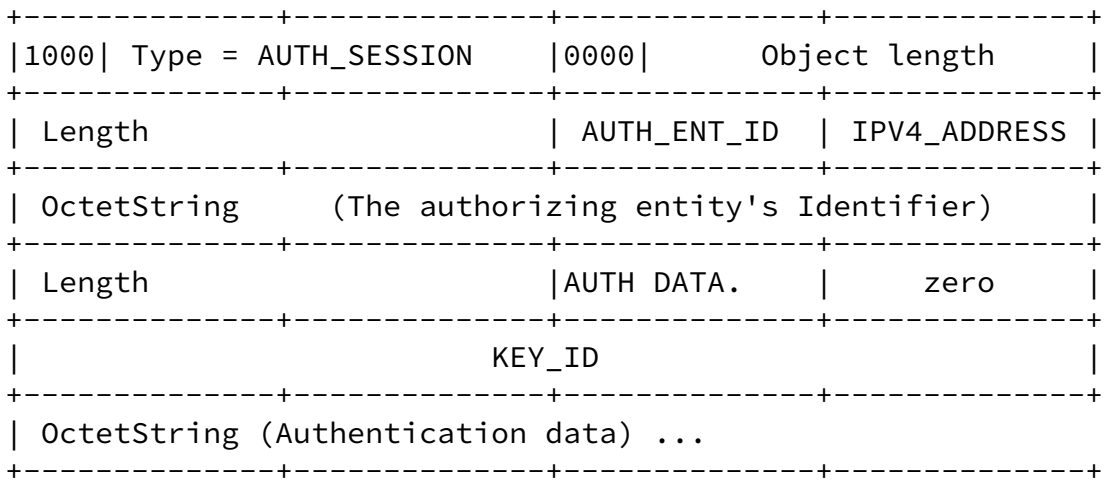
OctetString: The OctetString contains the authentication data of the AUTH_SESSION.

4. Integrity of the AUTH_SESSION policy element

This section describes how to ensure the integrity of the policy element is preserved.

4.1. Shared symmetric keys

In shared symmetric key environments, the AUTH_ENT_ID MUST be of subtypes: IPV4_ADDRESS, IPV6_ADDRESS, FQDN, ASCII_DN, UNICODE_DN or URI. An example AUTH_SESSION object is shown below.



4.1.1. Operational Setting using shared symmetric keys

This assumes both the Authorizing Entity and the Network router/PDP are provisioned with shared symmetric keys and with policies detailing which algorithm to be used for computing the authentication data along with the expected length of the authentication data for that particular algorithm.

Key maintenance is outside the scope of this document, but AUTH_SESSION implementations MUST at least provide the ability to manually configure keys and their parameters. The key used to produce the authentication data is identified by the AUTH_ENT_ID field. Since multiple keys may be configured for a particular AUTH_ENT_ID value, the first 32 bits of the AUTH_DATA field MUST be a key ID to be used to identify the appropriate key. Each key must also be configured with lifetime parameters for the time period within which it is valid as well as an associated cryptographic algorithm parameter specifying the algorithm to be used with the key. At a minimum, all AUTH_SESSION implementations MUST support the HMAC-MD5-128 [RFC1321] [RFC2104] cryptographic algorithm for computing the authentication data.

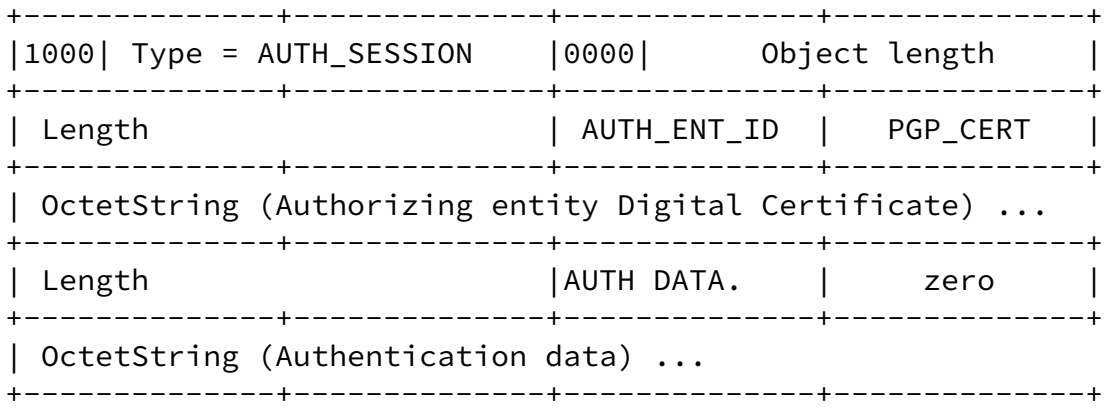
It is good practice to regularly change keys. Keys MUST be configurable such that their lifetimes overlap allowing smooth transitions between keys. At the midpoint of the lifetime overlap between two keys, senders should transition from using the current key to the next/longer-lived key. Meanwhile, receivers simply accept any identified key received within its configured lifetime and reject those that are not.

4.2. Kerberos

RFC 3520 provides a mechanism to secure the authorization token using Kerberos. Kerberos, however, has not seen deployment in this context and is not well applicable for this particular usage scenario. Hence, Kerberos support will not be provided by this specification.

4.3. Public Key

In a public key environment, the AUTH_ENT_ID MUST be of the subtypes: X509_V3_CERT or PGP_CERT. The authentication data is used for authenticating the authorizing entity. An example of the public key AUTH_SESSION policy element is shown below.



4.3.1. Operational Setting for public key based authentication

Public key based authentication assumes the following:

- Authorizing entities have a pair of keys (private key and public key).
- Private key is secured with the authorizing entity.
- Public keys are stored in digital certificates and a trusted party, certificate authority (CA) issues these digital

certificates.

- The verifier (PDP or router) has the ability to verify the digital certificate.

Authorizing entity uses its private key to generate AUTHENTICATION_DATA. Authenticators (router, PDP) use the authorizing entity's public key (stored in the digital certificate) to verify and authenticate the policy element.

4.3.1.1. X.509 V3 digital certificates

When the AUTH_ENT_ID is of type X509_V3_CERT, AUTHENTICATION_DATA MUST be generated following these steps:

- A Signed-data is constructed as defined in RFC3852 [RFC3852] . A digest is computed on the content (as specified in Section 6.1) with a signer-specific message-digest algorithm. The certificates field contains the chain of authorizing entity's X.509 V3 digital certificates. The certificate revocation list is defined in the crls field. The digest output is digitally signed following Section 8 of RFC 3447 [RFC3447], using the signer's private key.

When the AUTH_ENT_ID is of type X509_V3_CERT, verification MUST be done following these steps:

- Parse the X.509 V3 certificate to extract the distinguished name of the issuer of the certificate.
- Certification Path Validation is performed as defined in Section 6 of RFC 3280.
- Parse through the Certificate Revocation list to verify that the received certificate is not listed.
- Once the X.509 V3 certificate is validated, the public key of the authorizing entity can be extracted from the certificate.
- Extract the digest algorithm and the length of the digested data by parsing the CMS signed-data.
- The recipient independently computes the message digest. This

message digest and the signer's public key are used to verify the signature value.

This verification ensures integrity, non-repudiation and data origin.

4.3.1.2. PGP digital certificates

When the AUTH_ENT_ID is of type PGP_CERT, AUTHENTICATION_DATA MUST be generated following these steps:

- AUTHENTICATION_DATA contains a Signature Packet as defined in Section 5.2.3 of RFC 2440. In summary:
- Compute the hash of all data in the AUTH_SESSION policy element up to the AUTHENTICATION_DATA.
- The hash output is digitally signed following Section 8 of RFC 3447, using the signer's private key.

When the AUTH_ENT_ID is of type PGP_CERT, verification MUST be done following these steps:

- Validate the certificate.
- Once the PGP certificate is validated, the public key of the authorizing entity can be extracted from the certificate.
- Extract the hash algorithm and the length of the hashed data by parsing the PGP signature packet.
- The recipient independently computes the message digest. This message digest and the signer's public key are used to verify the signature value.

This verification ensures integrity, non-repudiation and data origin.

5. Framework

RFC3521 [RFC3521] describes a framework in which the AUTH_SESSION policy element may be utilized to transport information required for authorizing resource reservation for media flows. RFC3521 introduces 4 different models:

1. The coupled model

2. The associated model with one policy server

3. The associated model with two policy servers

4. The non-associated model.

The fields that are required in an AUTH SESSION policy element dependent on which of the models is used.

5.1. The Coupled Model

In the coupled model, the only information that MUST be included in the policy element is the SESSION_ID; it is used by the Authorizing Entity to correlate the resource reservation request with the media authorized during session set up. Since the End Host is assumed to be untrusted, the Policy Server SHOULD take measures to ensure that the integrity of the SESSION_ID is preserved in transit; the exact

mechanisms to be used and the format of the SESSION_ID are implementation dependent.

5.2. The associated model with one policy server

In this model, the contents of the AUTH_SESSION policy element MUST include:

- A session identifier - SESSION_ID. This is information that the authorizing entity can use to correlate the resource request with the media authorized during session set up.

- The identity of the authorizing entity - AUTH_ENT_ID. This information is used by an NN to determine which authorizing entity (Policy Server) should be used to solicit resource policy decisions.

In some environments, an NN may have no means for determining if the identity refers to a legitimate Policy Server within its domain. In order to protect against redirection of authorization requests to a bogus authorizing entity, the AUTH_SESSION MUST also include:

AUTHENTICATION_DATA. This authentication data is calculated over all other fields of the AUTH_SESSION policy element.

5.3. The associated model with two policy servers

The content of the AUTH_SESSION Policy Element is identical to the associated model with one policy server.

5.4. The non-associated model

In this model, the AUTH_SESSION MUST contain sufficient information to allow the Policy Server to make resource policy decisions autonomously from the authorizing entity. The policy element is created using information about the session by the authorizing entity. The information in the AUTH_SESSION policy element MUST include:

- Calling party IP address or Identity (e.g., FQDN) - SOURCE_ADDR X-TYPE

- Called party IP address or Identity (e.g., FQDN) - DEST_ADDR X-TYPE
- The characteristics of (each of) the media stream(s) authorized for this session - RESOURCES X-TYPE
- The authorization lifetime - START_TIME X-TYPE
- The identity of the authorizing entity to allow for validation of the token in shared symmetric key and Kerberos schemes - AUTH_ENT_ID X-TYPE
- The credentials of the authorizing entity in a public-key scheme - AUTH_ENT_ID X-TYPE
- Authentication data used to prevent tampering with the AUTH_SESSION policy element - AUTHENTICATION_DATA

Furthermore, the AUTH_SESSION policy element MAY contain:

- The lifetime of (each of) the media stream(s) - END_TIME X-TYPE
- Calling party port number - SOURCE_ADDR X-TYPE
- Called party port number - DEST_ADDR X-TYPE

All AUTH_SESSION fields MUST match with the resource request. If a field does not match, the request SHOULD be denied.

6. Message Processing Rules

This section discusses the message processing related to the AUTH_SESSION object. We describe the details of the QoS NSLP and NAT/FW NSLP. New NSLP protocols should use the same logic in making use of the AUTH_SESSION object.

6.1. Generation of the AUTH_SESSION by the authorizing entity

1. Generate the AUTH_SESSION policy element with the appropriate contents as specified in Section 5.

2. If authentication is needed, the entire AUTH_SESSION policy element is constructed, excluding the length, type and subtype fields of the AUTH_SESSION field. Note that the message MUST include either a START_TIME or a SESSION_ID (See Section 9), to prevent replay attacks. The output of the authentication algorithm, plus appropriate header information, is appended to the AUTH_SESSION policy element.

6.2. Processing within the QoS NSLP

The AUTH_SESSION object may be used with QoS NSLP QUERY and RESERVE messages to authorize the query operation for network resources, and a resource reservation request, respectively.

Moreover, the AUTH_SESSION object may also be used with RESPONSE messages in order to indicate that the authorizing entity changed the original request. For example, the session start or end times may have been modified, or the client may have requested authorization for all ports, but the authorizing entity only allowed the use of certain ports.

If the QoS NSIS Initiator (QNI) receives a RESPONSE message with an AUTH_SESSION object, the QNI MUST inspect the AUTH_SESSION object to see what authentication attribute was changed by an authorizing entity. The QNI SHOULD also silently accept AUTH_SESSION objects in RESPONSE message which do not indicate any change to the original authorization request.

6.2.1. Message Generation

A QoS NSLP message is created as specified in [QoS NSLP].

1. The policy element received from the authorizing entity MUST be copied without modification into the AUTH_SESSION object.

2. The AUTH_SESSION object (containing the policy element) is inserted in the NSLP message in the appropriate place.

6.2.2. Message Reception

The QoS NSLP message is processed as specified in [QOS NSLP] with following modifications.

1. If the QNE is policy aware then it SHOULD use the Diameter QoS application or the RADIUS QoS protocol to communicate with the PDP. To construct the AAA message it is necessary to extract the AUTH_SESSION object and the QoS related objects from the QoS NSLP message and to craft the respective RADIUS or Diameter message. The message processing and object format is described in the respective RADIUS or Diameter QoS protocol, respectively. If the QNE is policy unaware then it ignores the policy data objects and continues processing the NSLP message.

2. If the response from the PDP is negative the request must be rejected. A negative response in RADIUS is an Access-Reject and in Diameter is based on the 'DIAMETER_SUCCESS' value in the Result-Code AVP of the QoS-Authz-Answer (QAA) message. The QNE must contruct and send a RESPONSE message with the status of authorization failure as specified in [QoS NSLP].

3. Continue processing the NSIS message.

6.2.3. Authorization (QNE/PDP)

1. Retrieve the policy element from the AUTH_SESSION object. Check the PE type field and return an error if the identity type is not supported.

2. Verify the message integrity.

   * Shared symmetric key authentication: The QNE/PDP uses the AUTH_ENT_ID field to consult a table keyed by that field. The table should identify the cryptographic authentication algorithm to be used along with the expected length of the authentication data and the shared symmetric key for the authorizing entity. Verify that the indicated length of the authentication data is consistent with the configured table entry and validate the authentication data.

   * Public Key: Validate the certificate chain against the trusted Certificate Authority (CA) and validate the message signature using the public key.

* Kerberos based usage is not provided by this document.

3. Once the identity of the authorizing entity and the validity of the service request has been established, the authorizing router/PDP MUST then consult its authorization policy in order to determine whether or not the specific request is authorized (e.g., based on available credits, information in the subscriber's database). To the extent to which these access control decisions require supplementary information, routers/PDPs MUST ensure that supplementary information is obtained securely.

4. Verify the requested resources do not exceed the authorized QoS.

6.2.4. Error Signaling

When the PDP (e.g., a RADIUS or Diameter server) fails to verify the policy element then the appropriate actions described the respective AAA document need to be taken.

The QNE node MUST return a RESPONSE message with the INFO_SPEC error code Authorization Failure as defined in the QoS NSLP specification. The QNE MAY include an INFO_SPEC Object Value Info to indicate which AUTH_SESSION attribute created the error.

6.3. Processing with the NAT/FW NSLP

This section presents processing tules for the NAT/FW NSLP.

6.3.1. Message Generation

A NAT/FW NSLP message is created as specified in [NATFW NSLP].

1. The policy element received from the authorizing entity MUST be copied without modification into the AUTH_SESSION object.

2. The AUTH_SESSION object (containing the policy element) is inserted in the NATFW NSLP message in the appropriate place.

6.3.2. Message Reception

The NAT/FW NSLP message is processed as specified in [NATFW NSLP] with following modifications.

1. If the router is policy aware then it SHOULD use the Diameter application or the RADIUS protocol to communicate with the PDP. To construct the AAA message it is necessary to extract the AUTH_SESSION element and the NATFW policy rule related objects from the NSLP message and to craft the respective RADIUS or

Diameter message. The message processing and object format is described in the respective RADIUS or Diameter protocols, respectively. If the router is policy unaware then it ignores the policy data objects and continues processing the NSLP message.

2. Reject the message if the response from the PDP is negative. A negative response in RADIUS is an Access-Reject and in Diameter is based on the 'DIAMETER_SUCCESS' value in the Result-Code AVP.

3. Continue processing the NSIS message.

6.3.3. Authorization (Router/PDP)

1. Retrieve the AUTH_SESSION object and the policy element. Check the PE type field and return an error if the identity type is not supported.

2. Verify the message integrity.

   * Shared symmetric key authentication: The Network router/PDP uses the AUTH_ENT_ID field to consult a table keyed by that field. The table should identify the cryptographic authentication algorithm to be used along with the expected length of the authentication data and the shared symmetric key for the authorizing entity. Verify that the indicated length of the authentication data is consistent with the configured table entry and validate the authentication data.

   * Public Key: Validate the certificate chain against the trusted Certificate Authority (CA) and validate the message signature using the public key.

   * - Kerberos based usage is not provided by this document.

3. Once the identity of the authorizing entity and the validity of the service request has been established, the authorizing router/ PDP MUST then consult its authorization policy in order to deter mine whether or not the specific request is authorized. To the extent to which these access control decisions require supplementary information, routers/PDPs MUST ensure that supplementary information is obtained securely.

6.3.4. Error Signaling

When the PDP (e.g., a RADIUS or Diameter server) fails to verify the AUTH_SESSION element then the appropriate actions described the respective AAA document need to be taken. The NATFW NSLP node MUST

return an error message of class 'Permanent failure' (0x5) with error code 'Authorization failed' (0x02).

7. Security Considerations

This document describes a mechanism for session authorization to prevent theft of service. There are three types of security issues to consider: protectiong against replay attacks, integrity of the AUTH_SESSION object, and the choice of the authentication algorithms and keys.

The first issue, replay attacks, MUST be prevented. In the non-associated model, the AUTH_SESSION object MUST include a START_TIME field and the Policy Servers MUST support NTP to ensure proper clock synchronization. Failure to ensure proper clock synchronization will allow replay attacks since the clocks of the different network entities may not be in synch. The start time is used to verify that the request is not being replayed at a later time. In all other models, the SESSION_ID is used by the Policy Server to ensure that the resource request successfully correlates with records of an authorized session. If a AUTH_SESSION object is replayed, it MUST be detected by the policy server (using internal algorithms) and the request MUST be rejected.

The second issue, the integrity of the policy element, is preserved in untrusted environments by including the AUTHENTICATION_DATA attribute. Therefore, this attribute MUST always be included.

In environments where shared symmetric keys are possible, they should be used in order to keep the AUTH_SESSION policy element size to a strict minimum, e.g., when wireless links are used. A secondary

option would be PKI authentication, which provides a high level of security and good scalability. However, it requires the presence of credentials in the AUTH_SESSION policy element which impacts its size.

Further security issues are outlined in RFC 4081 [RFC4081].

8. IANA Considerations

This specification makes the following request to IANA:

1. Assign a new object value for the AUTH_SESSION object from the shared NSLP object value space.

2. All AUTH_SESSION object internal values and numbers should be taken from the allocations already done for RFC 3520 [RFC3520]. Yet, this specification does make use of two X-types introduced by RFC3520: Session ID and Resources.

9. Acknowledgements

This document is based on the RFC 3520 [RFC3520] and credit therefore goes to the authors of RFC 3520, namely Louis-Nicolas Hamer, Brett Kosinski, Bill Gage and Hugh Shieh.

10. References

10.1. Normative References

[I-D.ietf-nsis-nslp-natfw]
Stiemerling, M., "NAT/Firewall NSIS Signaling Layer Protocol (NSLP)", draft-ietf-nsis-nslp-natfw-13 (work in

progress), October 2006.

[I-D.ietf-nsis-ntlp]
Schulzrinne, H. and R. Hancock, "GIST: General Internet Signalling Transport", draft-ietf-nsis-ntlp-12 (work in progress), March 2007.

[I-D.ietf-nsis-qos-nslp]
Manner, J., "NSLP for Quality-of-Service Signaling", draft-ietf-nsis-qos-nslp-12 (work in progress), October 2006.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3447] Jonsson, J. and B. Kaliski, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1", RFC 3447, February 2003.

[RFC4080] Hancock, R., Karagiannis, G., Loughney, J., and S. Van den Bosch, "Next Steps in Signaling (NSIS): Framework", RFC 4080, June 2005.

[RFC4081] Tschofenig, H. and D. Kroeselberg, "Security Threats for Next Steps in Signaling (NSIS)", RFC 4081, June 2005.

10.2. Informative References

[RFC1321] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, April 1992.

[RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, February 1997.

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

[RFC3521] Hamer, L-N., Gage, B., and H. Shieh, "Framework for Session Set-up with Media Authorization", RFC 3521,

April 2003.

[RFC3852] Housley, R., "Cryptographic Message Syntax (CMS)", RFC 3852, July 2004.

Authors' Addresses

Jukka Manner
Helsinki University of Technology (TKK)
P.O. Box 5400
Espoo FIN-02015 TKK
Finland

Phone: +358 9 451 4161
Email: jmanner@tml.hut.fi
URI: http://www.tml.tkk.fi/~jmanner/

Martin Stiemerling
Network Laboratories, NEC Europe Ltd.
Kurfuersten-Anlage 36
Heidelberg 69115
Germany

Phone: +49 (0) 6221 4342 113
Email: stiemerling@netlab.nec.de
URI: http://www.stiemerling.org

Hannes Tschofenig
Siemens Networks GmbH & Co KG
Otto-Hahn-Ring 6
Munich, Bavaria 81739
Germany

Phone: +49 89 636 40390
Email: Hannes.Tschofenig@siemens.com
URI: http://www.tschofenig.com

Full Copyright Statement

Intellectual Property

copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgment

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