Network Working Group
INTERNET-DRAFT
Updates: 5176
Category: Standards Track
<draft-dekok-radext-coa-proxy-00.txt>
3 July 2014

DeKok, Alan
FreeRADIUS
J. Korhonen
Nokia Siemens Networks

Dynamic Authorization Proxying in Remote Authorization Dial-In User Service Protocol (RADIUS) draft-dekok-radext-coa-proxy-00.txt

Abstract

RFC 5176 defines Change of Authorization (CoA) and Disconnect Message (DM) behavior for RADIUS. Section 3.1 of that document suggests that proxying these messages is possible, but gives no guidance as to how that is done. This specification corrects that ommission.

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

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

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

This Internet-Draft will expire on January 4, 2015.

Copyright Notice

Table of Contents

1. Introduction

RFC 5176 [RFC5176] defines Change of Authorization (CoA) and Disconnect Message (DM) behavior for RADIUS. Section 3.1 of that document suggests that proxying these messages is possible, but gives no guidance as to how that is done. This ommission means that proxying of CoA packets is, in practice, impossible.

We correct that ommission here.

1.1. Terminology

This document frequently uses the following terms:

Network Access Identifier

The Network Access Identifier (NAI) is the user identity submitted by the client during network access authentication. The purpose of the NAI is to identify the user as well as to assist in the routing of the authentication request. Please note that the NAI may not necessarily be the same as the user's email address or the user identity submitted in an application layer authentication.

Network Access Server

The Network Access Server (NAS) is the device that clients connect to in order to get access to the network. In PPTP terminology, this is referred to as the PPTP Access Concentrator (PAC), and in L2TP terminology, it is referred to as the L2TP Access Concentrator (LAC). In IEEE 802.11, it is referred to as an Access Point.

Home Network

The home network of a user.

Visited Network

The network which is accessed by a user, when that network is not their home network.

1.2. Requirements Language

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 [RFC2119].

2. Problem Statement

This section describes how RADIUS proxying works, how CoA packets work, and why CoA proxying does not work in the current system.

2.1. Typical RADIUS Proxying

When a RADIUS server proxies an Access-Request packet, it typically does so based on the contents of the User-Name attribute, which contains Network Access Identifier [NAI]. Other methods are possible, but we restrict ourselves to the most common usage.

The proxy server looks up the "Realm" portion of the NAI in a logical AAA routing table, as described in Section 3 of [NAI]. The entry in that table is the "next hop" to which the packet is sent. This "next hop" may be another proxy, or it may be the home server for that realm.

The "next hop" may perform the same Realm lookup, and the proxy the packet also. Alternatively, if the "next hop" is the Home Server for that realm, it will typically authenticate the user, and respond with an Access-Accept, Access-Reject, or Access-Challenge.

The response can be returned from the home server to the visited network, because each proxy server tracks the requests it has forwarded. When a response packet is by the proxy, it is matched to an incoming request, which lets the proxy forward the response to the source of the original request.

2.2. CoA Processing

[RFC5176] describes how CoA clients (often RADIUS servers) will send packets to CoA servers (often RADIUS clients). In typical use, CoA packets are sent within one network. That is, within the same "Realm". When used within one "Realm", there is only one "hop" for packets to take, so no proxying is necessary.

2.3. Failure of CoA Proxying

In the case of CoA proxying, the above scenarios fail. CoA packets may be sent minutes to hours after reception of the original Access-Request. In addition, the packet codes are different, so there is no way to match a CoA-Request packet to a particular Access-Request packet. There is therefore no "reverse path" for the CoA packet to follow.

As with Access-Request proxying, CoA proxying can be done done between Realms. There exists potentially multiple "hops" for packets

to follow. Packets cannot be forwarded to the Visited Network, based on the contents of the User-Name attribute, as that contains the Realm of the Home Network.

The conclusion is therefore that CoA proxying is impossible when using behavior defined in [RFC5176]. There is, however a solution.

3. How to Perform CoA Proxying

The solution is seen in the Operator-Name attribute defined in [RFC5580], Section 4.1. We repeat portions of that definition here for clarity:

> This attribute carries the operator namespace identifier and the operator name. The operator name is combined with the namespace identifier to uniquely identify the owner of an access network.

Followed by a description of the REALM namespace:

> REALM ('1' (0x31)):
>
> The REALM operator namespace can be used to indicate operator names based on any registered domain name. Such names are required to be unique, and the rights to use a given realm name are obtained coincident with acquiring the rights to use a particular Fully Qualified Domain Name (FQDN). ...

In short, the Operator-Name attribute contains the an ASCII "1", followed by the Realm of the Visited Network. e.g. for the "example.com" realm, the Operator-Name attribute contains the text "1example.com". This information is precisely what we need to perform CoA proxying.

The only missing information is which NAS is managing the user. We may expect that the Visited Network will track this information, but there is no requirement for it to do so. We therefore need an additional attribute to contain this information.

3.1. Operator-NAS-Identifier

The Operator-NAS-Identifier attribute contains opaque information identifying a NAS. It MAY appear in the following packets: Access-Request, Accounting-Request, CoA-Request, DM-Request. Operator-NAS-Identifier MUST NOT appear in any other packet.

Operator-NAS-Identifier MAY occur in a packet if the packet also contains an Operator-Name attribute. Operator-NAS-Identifier MUST NOT appear in a packet if there is no Operator-Name in the packet.

Operator-NAS-Identifier MUST NOT occur more than once in a packet.

When an Operator-NAS-Identifer attribute is added by a proxy in a Visited Network, the following attributes MUST be deleted: NAS-IP-Address, NAS-IPv6-Address, NAS-Identifier. The proxy MUST then add a NAS-Identifier attribute, in order satisfy the requirments of Secton 4.1 of [RFC2865], and of [RFC2866]. We suggest that the contents of the NAS-Identifier be the Realm name of the Visited Network.

Description

An opaque token describing the NAS a user has logged into.

Type

TBD. To be assigned by IANA

Length

TBD. Depends on IANA allocation.

Implementations supporting this attribute MUST be able to handle between one (1) and twenty (20) octets of data. Implementations creating an Operator-NAS-Identifier SHOULD NOT create attributes with more than twenty octets of data. A twenty octet string is more than sufficient to individually address all of the NASes on the planet.

Data Type

string. See [DATA] Section 2.6 for a definition.

Value

The contents of this attribute are an opaque token interpretable only by the Visited Network. The attribute MUST NOT contain any secret or private information.

4. Functionality

This section describes how the two attributes work together to permit CoA proxying.

4.1. User Login

The user logs in. When a Visited Network sees that the packet is proxied, it adds an Operator-Name with "1" followed by it’s own realm

name. It MAY also add an Operator-NAS-Identifier.

The proxies then forward the packet. They MUST NOT delete or modify Operator-Name and/or Operator-NAS-Identifier.

The Home Server records both Operator-Name and Operator-NAS-Identfier along with other information about the users session.

4.2. CoA Proxing

When the Home Server decides to disconnect a user, it looks up the Operator-Name and Operator-NAS-Identifer, along with other user session identifiers as described in [RFC5176]. It then looks up the Operator-Name in the logical AAA routing table to find the CoA server for that realm (which may be a proxy). The CoA-Request is then sent to that server.

The CoA server receives the request, and if it is a proxy, performs a similar lookup as done by the Home Server. The packet is then proxied repeatedly until it reaches the Visited Network.

If the proxy cannot find a destination for the request, or if no Operator-Name attribute exists in the request, the proxy returns a CoA-NAK with Error-Cause 502 (Request Not Routable).

The Visited Network recieves the CoA-Request packet, and uses the Operator-NAS-Identifier attribute to determine which local CoA server (i.e. NAS) the packet should be sent to.

If no CoA server can be found, the Visited Network return a CoA-NAK with Error-Cause 403 (NAS Identification Mismatch).

Any response from the CoA server (NAS) is returned to the Home Network.

5. Security Considerations

This specification incorporates by reference the [RFC6929] Section 11. In short, RADIUS has known issues which are discussed there.

This specification adds one new attribute, and defines new behavior for RADIUS proxying. As this behavior mirrors existing RADIUS proxying, we do not believe that it introduces any new security issues.

Operator-NAS-Identifier should remain secure. We don’t say how.

6. IANA Considerations

IANA is instructed to allocated one new RADIUS attribute, as per Section 3.1, above.

7. References

7.1. Normative References

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

[RFC2865]
Rigney, C., Willens, S., Rubens, A. and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, June 2000.

[RFC5580]
Tschofenig H., Ed. "Carrying Location Objects in RADIUS and Diameter", RFC 5580, August 2009.

[RFC6929]
DeKok A. and Lior, A., "Remote Authentication Dial-In User Service (RADIUS) Protocol Extensions", RFC 6929, April 2013.

[NAI]
DeKok A., "The Network Access Identifier", draft-ietf-radext-nai-06.txt, June 2013.

[DATA]
DeKok A., "Data Types in the Remote Authentication Dial-In User Service Protocol (RADIUS)", draft-dekok-radext-datatypes-04.txt, Juen 2014

7.2. Informative References

[RFC2866]
Rigney, C., "RADIUS Accounting", RFC 2866, June 2000.

[RFC5176]
Chiba, M. et al, "Dynamic Authorization Extensions to Remote Authentication Dial In User Service (RADIUS)", RFC 5176, January 2008.

Acknowledgments

Stuff

Authors' Addresses

Alan DeKok
The FreeRADIUS Server Project

Email: aland@freeradius.org

Jouni Korhonen
Nokia Siemens Networks
Linnoitustie 6
Espoo FI-02600
Finland

EMail: jouni.nospam@gmail.com

Network Working Group
INTERNET-DRAFT
Updates: 2865,3162,6158,6572
Category: Standards Track
<draft-dekok-radext-datatypes-06.txt>
1 April 2015

DeKok, Alan
FreeRADIUS

Data Types in the Remote Authentication Dial-In User Service Protocol (RADIUS) draft-dekok-radext-datatypes-06.txt

Abstract

RADIUS specifications have used data types for two decades without defining them as managed entities. During this time, RADIUS implementations have named the data types, and have used them in attribute definitions. This document updates the specifications to better follow established practice. We do this by naming the data types defined in RFC 6158, which have been used since at least RFC 2865. We provide an IANA registry for the data types, and update the RADIUS Attribute Type registry to include a "Data Type" field for each attribute. Finally, we recommend that authors of RADIUS specifications use these types in preference to existing practice.

Status of this Memo

Table of Contents

1. Introduction

RADIUS specifications have historically defined attributes in terms of name, type value, and data type. Of these three pieces of information, only the type value is managed by IANA. There is no management of, or restriction on, the attribute name, as discussed in [RFC6929] Section 2.7.1. There is no management of data type name or definition. This document defines an IANA registry for data types, and updates the RADIUS Attribute Type registry to use those newly defined data types.

In this section, we review the use of data types in specifications and implementations. Whe highlight ambiguities and inconsistencies. The rest of this document is devoted to resolving those problems.

1.1. Specification use of Data Types

A number of data type names and definitions are given in [RFC2865] Section 5, at the bottom of page 25. These data types are named and clearly defined. However, this practice was not continued in later specifications.

Specifically, [RFC2865] defines attributes of data type "address" to carry IPv4 addresses. Despite this definition, [RFC3162] defines attributes of data type "Address" to carry IPv6 addresses. We suggest that the use of the word "address" to refer to disparate data types is problematic.

Other failures are that [RFC3162] does not give a data type name and definition for the data types IPv6 address, Interface-Id, or IPv6 prefix. [RFC2869] defines Event-Timestamp to carry a time, but does not re-use the "time" data type defined in [RFC2865]. Instead, it just repeats the "time" definition. [RFC6572] defines multiple attributes which carry IPv4 prefixes. However, an "IPv4 prefix" data type is not named, defined as a data type, or called out as an addition to RADIUS. Further, [RFC6572] does not follow the recommendations of [RFC6158], and does not explain why it fails to follow those recommendations.

These ambiguities and inconsistencies need to be resolved.

1.2. Implementation use of Data Types

RADIUS implementations often use "dictionaries" to map attribute names to type values, and to define data types for each attribute. The data types in the dictionaries are defined by each implementation, but correspond to the "ad hoc" data types used in the specifications.

In effect, implementations have seen the need for well-defined data types, and have created them. It is time for RADIUS specifications to follow this practice.

This document requires no changes to any RADIUS implementation, past, present, or future. It instead documents existing practice, in order to simplify the process of writing RADIUS specifications, to clarify the interpretation of RADIUS standards, and to improve the communication between specification authors and IANA.

1.3. Requirements Language

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 [RFC2119].

2. Data Type Definitions

This section defines the new data types. For each data type, it gives a definition, a name, a number, a length, and an encoding format. Where relevant, it describes subfields contained within the data type. These definitions have no impact on existing RADIUS implementations. There is no requirement that implementations use these names.

Where possible, the name of each data type has been taken from previous specifications. In some cases, a different name has been chosen. The change of name is sometimes required to avoid ambiguity (i.e. "address" versus "Address"). Otherwise, the new name has been chosen to be compatible with [RFC2865], or with use in common implementations. In some cases, new names are chosen to clarify the interpretation of the data type.

The numbers assigned herein for the data types have no meaning other than to permit them to be tracked by IANA. As RADIUS does not encode information about data types in a packet, the numbers assigned to a data type will never occur in a packet.

The encoding of each data type is taken from previous specifications. The fields are transmitted from left to right.

Where the data types have inter-dependencies, the simplest data type is given first, and dependent ones are given later.

We do not create specific data types for the "tagged" attributes, as discussed in [RFC2868]. That specification defines the "tagged" attributes as being backwards compatible with pre-existing data types. In addition, [RFC6158] Section 2.1 says that "tagged" attributes should not be used. There is therefore no benefit to defining additional data types for these attributes.

Similarly, we do not create data types for some attributes having complex structure, such as CHAP-Password, ARAP-Features, or Location-Capable. We need to strike a balance between correcting earlier mistakes, and making this document more complex. In some cases, it is better to treat complex attributes as being of type "string", even though they need to be interpreted by RADIUS implementations.

Implementations not supporting a particular data type MUST treat attributes of that data type as being of data type "string". See Section 2.6, below for a definition of the "string" data type.

The definitions below use specialized names for various fields of attributes and data types. These names serve to address ambiguity of

the field names in previous specifications. For example, the term "Value" is used in [RFC2865] Section 5 to define a field which carries the contents of attribute. It is then used in later sections as the sub-field of attribute contents. The result is that the field is defined as recursively containing itself. Similarly, "String" is used both as a data type, and as a sub-field of other data types.

This document uses slightly different terminology than previous specifications, in order to be avoid ambiguity. The first addition is the following term:

Attr-Data

The "Value" field of an Attribute as defined in [RFC2865] Section 5. The contents of this field MUST be a valid data type as defined in the RADIUS Data Type registry.

In this document, we use the term "Value" only to refer to the contents of a data type, where that data type cannot carry other data types. In other cases, we refer to the contents of a data type as "Type-Data", to distinguish it from data of other types. For example, a data type "vsa" will contain a data field called "VSA-Data".

These terms are used in preference to the term "String", which was used in multiple incompatible ways. It is RECOMMENDED that future specifications use the new terms in order to maintain consistent definitions, and to avoid ambiguities.

2.1. integer

The "integer" data type encodes a 32-bit unsigned integer in network byte order. Where the range of values for a particular attribute is limited to a sub-set of the values, specifications MUST define the valid range. Values outside of the allowed ranges SHOULD be treated as invalid.

Name

integer

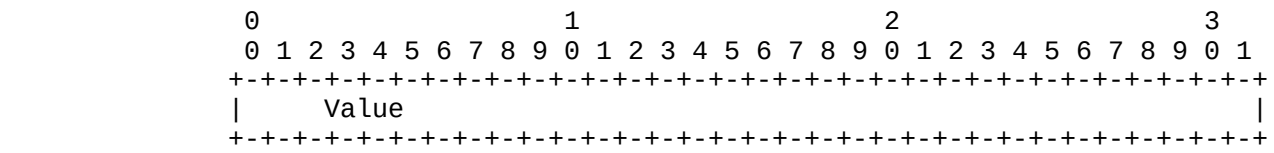
Number

1

Length

Four octets

Format



2.2. enum

The "enum" data type encodes a 32-bit unsigned integer in network byte order. It differs from the "integer" data type only in that it is used to define enumerated types, such as Service-Type. Specifications MUST define a valid set of enumerated values, along with a unique name for each value.

Name

enum

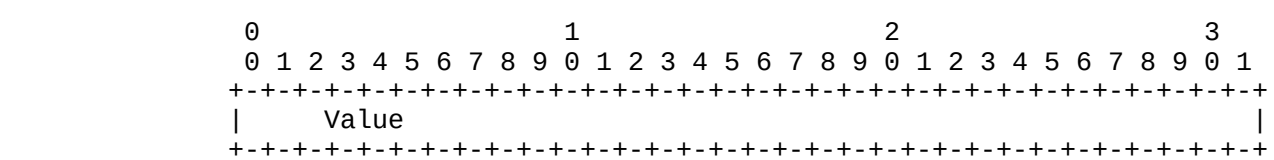
Number

2

Length

Four octets

Format



2.3. ipv4addr

The "ipv4addr" data type encodes an IPv4 address in network byte order. Where the range of address for a particular attribute is limited to a sub-set of possible addresses, specifications MUST define the valid range(s). Values outside of the allowed range SHOULD be treated as invalid.

Name

ipv4addr

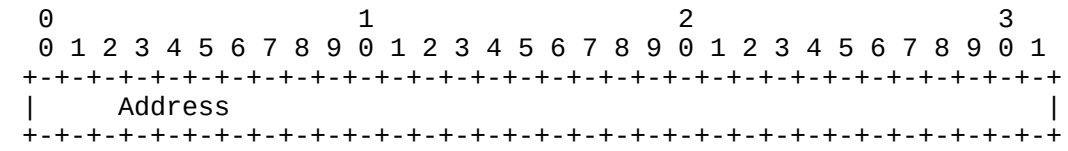
Number

3

Length

Four octets

Format



2.4. time

The "time" data type encodes time as a 32-bit unsigned value in network byte order and in seconds since 00:00:00 UTC, January 1, 1970. We note that dates before the year 2013 are likely to be erroneous.

Note that the "time" attribute is defined to be unsigned, which means it is not subject to a signed integer overflow in the year 2038.

Name

time

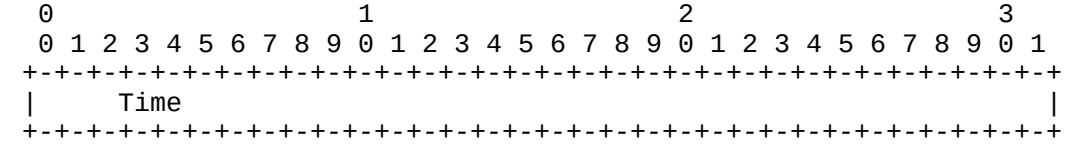
Number

4

Length

Four octets

Format



2.5. text

The "text" data type encodes UTF-8 text [RFC3629]. The maximum length of the text is given by the encapsulating attribute. Where the range of lengths for a particular attribute is limited to a sub-set of possible lengths, specifications MUST define the valid range(s).

Note that the "text" type does not terminate with a NUL octet (hex 00). The Attribute has a Length field and does not use a terminator. Texts of length zero (0) MUST NOT be sent; omit the entire attribute instead.

Name

text

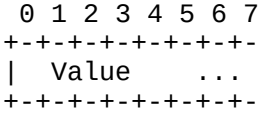
Number

5

Length

One or more octets.

Format



2.6. string

The "string" data type encodes binary data, as a sequence of undistinguished octets. Where the range of lengths for a particular attribute is limited to a sub-set of possible lengths, specifications MUST define the valid range(s).

Note that the "string" data type does not terminate with a NUL octet (hex 00). The Attribute has a Length field and does not use a terminator. Strings of length zero (0) MUST NOT be sent; omit the entire attribute instead.

Where there is a need to encapsulate complex data structures, and

TLVs cannot be used, the "string" data type MUST be used. This requirement include encapsulation of data structures defined outside of RADIUS, which are opaque to the RADIUS infrastucture. It also includes encapsulation of some data structures which are not opaque to RADIUS, such as the contents of CHAP-Password.

There is little reason to define a new RADIUS data type for only one attribute. However, where the complex data type cannot be represented as TLVs, and is expected to be used in many attributes, a new data type SHOULD be defined.

These requirements are stronger than [RFC6158], which makes the above encapsulation a "SHOULD". This document defines data types for use in RADIUS, so there are few reasons to avoid using them.

Name

string

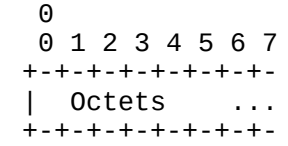
Number

6

Length

One or more octets.

Format



2.7. concat

The "concat" data type permits the transport of more than 253 octets of data in a "standard space" [RFC6929] attribute. It is otherwise identical to the "string" data type.

If multiple attributes of this data type are contained in a packet, all attributes of the same type code MUST be in order and they MUST be consecutive attributes in the packet.

The amount of data transported in a "concat" data type can be no more than the RADIUS packet size. In practice, the requirement to

transport multiple attributes means that the limit may be substantially smaller than one RADIUS packet. As a rough guide, is RECOMMENDED that this data type transport no more than 2048 octets of data.

The "concat" data type MAY be used for "standard space" attributes. It MUST NOT be used for attributes in the "short extended space" or the "long extended space". It MUST NOT be used in any field or subfields of the following data types: "tlv", "vsa", "extended", "long-extended", or "evs".

Name

concat

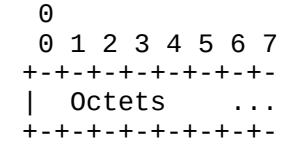
Number

7

Length

One or more octets.

Format



2.8. ifid

The "ifid" data type encodes an Interface-Id as an 8-octet string in network byte order.

Name

ifid

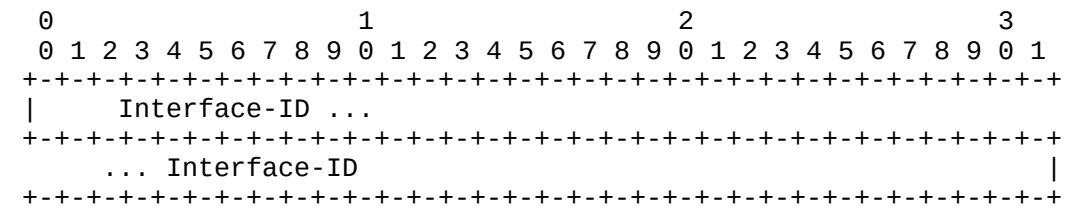
Number

8

Length

Eight octets

Format



2.9. ipv6addr

The "ipv6addr" data type encodes an IPv6 address in network byte order. Where the range of address for a particular attribute is limited to a sub-set of possible addresses, specifications MUST define the valid range(s).

Name

ipv6addr

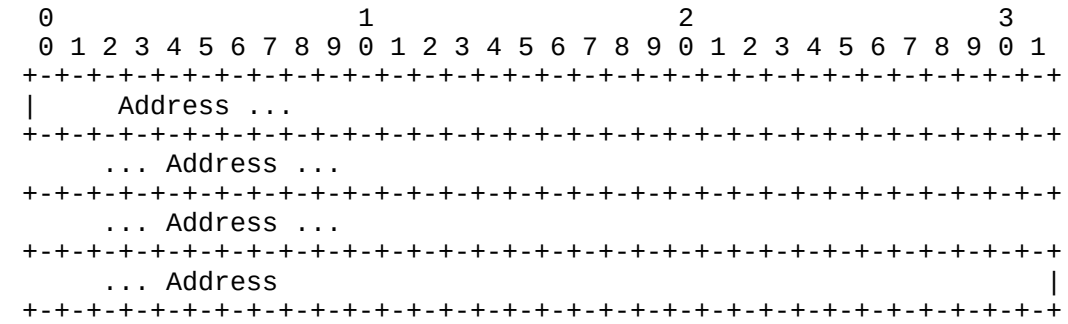
Number

9

Length

Sixteen octets

Format



2.10. ipv6prefix

The "ipv6prefix" data type encodes an IPv6 prefix, using both a prefix length and an IPv6 address in network byte order.

Name

ipv6prefix

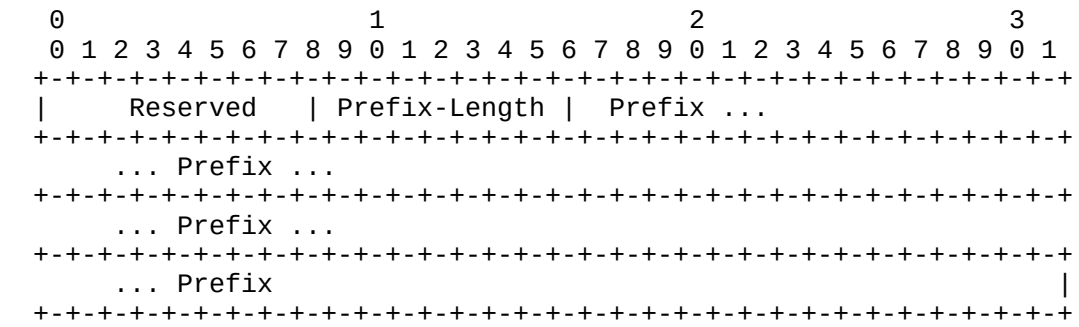
Number

10

Length

At least two, and no more than eighteen octets.

Format



Subfields

Reserved

This field, which is reserved and MUST be present, is always set to zero.

Prefix-Length

The length of the prefix, in bits. At least 0 and no larger than 128.

Prefix

The Prefix field is up to 16 octets in length. Bits outside of the Prefix-Length, if included, must be zero.

2.11. ipv4prefix

The "ipv4prefix" data type encodes an IPv4 prefix, using both a prefix length and an IPv4 address in network byte order.

Name

ipv4prefix

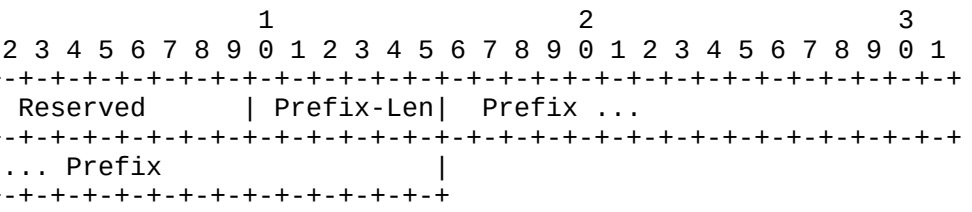
Number

11

Length

At least two, and no more than eighteen octets.

Format



Subfields

Reserved

This field, which is reserved and MUST be present, is always set to zero.

Prefix-Length

A 6-bit unsigned integer containing the length of the prefix, in bits. The values MUST be no larger than 32.

Prefix

The Prefix field is 4 octets in length. Bits outside of the Prefix-Length must be zero. Unlike the "ipv6prefix" data type, this field is fixed length. If the address is all zeros (i.e. "0.0.0.0", then the Prefix-Length MUST be set to 32.

2.12. integer64

The "integer64" data type encodes a 64-bit unsigned integer in network byte order. Where the range of values for a particular attribute is limited to a sub-set of the values, specifications MUST define the valid range(s).

Name

integer64

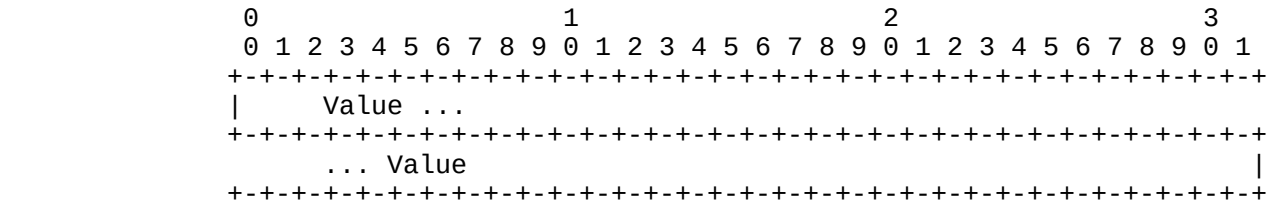
Number

12

Length

Eight octets

Format



2.13. tlv

The "tlv" data type encodes a type-length-value, as defined in [RFC6929] Section 2.3.

Name

tlv

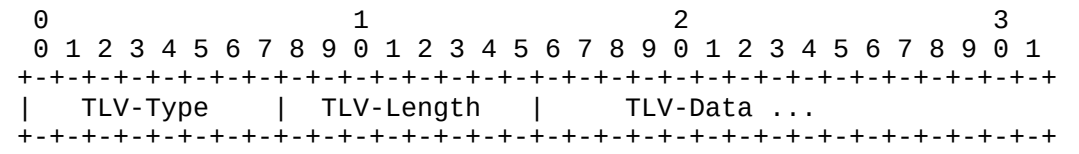
Number

13

Length

Three or more octets

Format



Subfields

TLV-Type

This field is one octet. Up-to-date values of this field are specified according to the policies and rules described in [RFC6929] Section 10. Values 254-255 are "Reserved" for use by future extensions to RADIUS. The value 26 has no special meaning, and MUST NOT be treated as a Vendor Specific attribute.

The TLV-Type is meaningful only within the context defined by "Type" fields of the encapsulating Attributes, using the dotted-number notation introduced in [RFC6929].

A RADIUS server MAY ignore Attributes with an unknown "TLV-Type".

A RADIUS client MAY ignore Attributes with an unknown "TLV-Type".

A RADIUS proxy SHOULD forward Attributes with an unknown "TLV-Type" verbatim.

TLV-Length

The TLV-Length field is one octet, and indicates the length of this TLV including the TLV-Type, TLV-Length and TLV-Value fields. It MUST have a value between 3 and 255. If a client or server receives a TLV with an invalid TLV-Length, then the attribute which encapsulates that TLV MUST be considered to be an "invalid attribute", and handled as per [RFC6929] Section 2.8.

TLVs having TLV-Length of zero (0) MUST NOT be sent; omit the entire TLV instead.

TLV-Data

The TLV-Data field is one or more octets and contains information specific to the Attribute. The format and length of the TLV-Data field is determined by the TLV-Type and TLV-

Length fields.

The TLV-Data field MUST contain only known RADIUS data types. The TLV-Data field MUST NOT contain any of the following data types: "concat", "vsa", "extended", "long-extended", or "evs".

2.14. vsa

The "vsa" data type encodes Vendor-Specific data, as given in [RFC2865] Section 5.26. It is used only in the Attr-Data field of a Vendor-Specific Attribute. It MUST NOT appear in the contents of any other data type.

Name

vsa

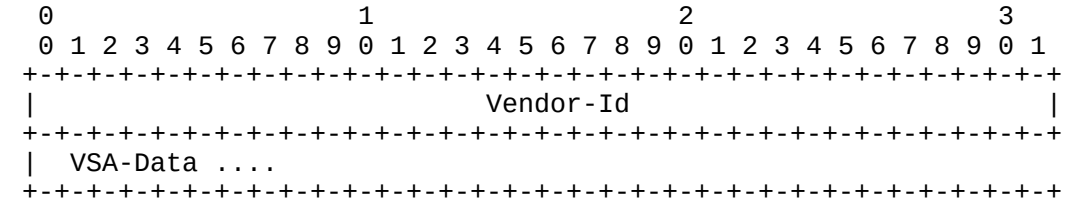
Number

14

Length

Five or more octets

Format



Subfields

Vendor-Id

The 4 octets are the Network Management Private Enterprise Code [PEN] of the Vendor in network byte order.

VSA-Data

The VSA-Data field is one or more octets. The actual format of the information is site or application specific, and a robust implementation SHOULD support the field as undistinguished

octets.

The codification of the range of allowed usage of this field is outside the scope of this specification.

It SHOULD be encoded as a sequence of "tlv" fields. The interpretation of the TLV-Type and TLV-Data fields are dependent on the vendor's definition of that attribute.

The "vsa" data type MUST be used as contents of the Attr-Data field of the Vendor-Specific attribute. The "vsa" data type MUST NOT appear in the contents of any other data type.

2.15. extended

The "extended" data type encodes the "Extended Type" format, as given in [RFC6929] Section 2.1. It is used only in the Attr-Data field of an Attribute allocated from the "standard space". It MUST NOT appear in the contents of any other data type.

Name

extended

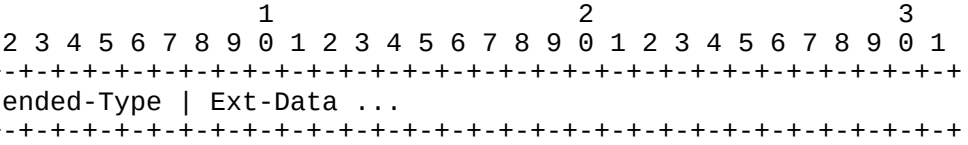
Number

15

Length

Two or more octets

Format



Subfields

Extended-Type

The Extended-Type field is one octet. Up-to-date values of this field are specified according to the policies and rules described in [RFC6929] Section 10. Unlike the Type field

defined in [RFC2865] Section 5, no values are allocated for experimental or implementation-specific use. Values 241-255 are reserved and MUST NOT be used.

The Extended-Type is meaningful only within a context defined by the Type field. That is, this field may be thought of as defining a new type space of the form "Type.Extended-Type". See [RFC6929] Section 2.5 for additional discussion.

A RADIUS server MAY ignore Attributes with an unknown "Type.Extended-Type".

A RADIUS client MAY ignore Attributes with an unknown "Type.Extended-Type".

Ext-Data

The contents of this field MUST be a valid data type as defined in the RADIUS Data Type registry. The Ext-Data field MUST NOT contain any of the following data types: "concat", "vsa", "extended", "long-extended", or "evs".

The Ext-Data field is one or more octets.

Implementations supporting this specification MUST use the Identifier of "Type.Extended-Type" to determine the interpretation of the Ext-Data field.

2.16. long-extended

The "long-extended" data type encodes the "Long Extended Type" format, as given in [RFC6929] Section 2.2. It is used only in the Attr-Data field of an Attribute. It MUST NOT appear in the contents of any other data type.

Name

long-extended

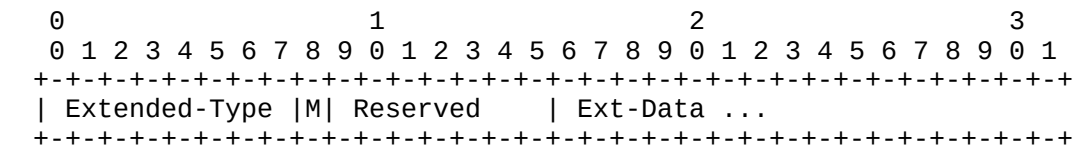
Number

16

Length

Three or more octets

Format



Subfields

Extended-Type

This field is identical to the Extended-Type field defined above in Section 2.13.

M (More)

The More field is one (1) bit in length, and indicates whether or not the current attribute contains "more" than 251 octets of data. The More field MUST be clear (0) if the Length field has value less than 255. The More field MAY be set (1) if the Length field has value of 255.

If the More field is set (1), it indicates that the Ext-Data field has been fragmented across multiple RADIUS attributes. When the More field is set (1), the attribute MUST have a Length field of value 255; there MUST be an attribute following this one; and the next attribute MUST have both the same Type and Extended Type. That is, multiple fragments of the same value MUST be in order and MUST be consecutive attributes in the packet, and the last attribute in a packet MUST NOT have the More field set (1).

That is, a packet containing a fragmented attribute needs to contain all fragments of the attribute, and those fragments need to be contiguous in the packet. RADIUS does not support inter-packet fragmentation, which means that fragmenting an attribute across multiple packets is impossible.

If a client or server receives an attribute fragment with the "More" field set (1), but for which no subsequent fragment can be found, then the fragmented attribute is considered to be an "invalid attribute", and handled as per [RFC6929] Section 2.8.

Reserved

This field is 7 bits long, and is reserved for future use. Implementations MUST set it to zero (0) when encoding an

attribute for sending in a packet. The contents SHOULD be ignored on reception.

Future specifications may define additional meaning for this field. Implementations therefore MUST NOT treat this field as invalid if it is non-zero.

Ext-Data

The contents of this field MUST be a valid data type as defined in the RADIUS Data Type registry. The Ext-Data field MUST NOT contain any of the following data types: "concat", "vsa", "extended", "long-extended", or "evs".

The Ext-Data field is one or more octets.

Implementations supporting this specification MUST use the Identifier of "Type.Extended-Type" to determine the interpretation of the Ext-Data field.

The length of the data MUST be taken as the sum of the lengths of the fragments (i.e. Ext-Data fields) from which it is constructed. Any interpretation of the resulting data MUST occur after the fragments have been reassembled. If the reassembled data does not match the expected format, each fragment MUST be treated as an "invalid attribute", and the reassembled data MUST be discarded.

We note that the maximum size of a fragmented attribute is limited only by the RADIUS packet length limitation. Implementations MUST be able to handle the case where one fragmented attribute completely fills the packet.

2.17. evs

The "evs" data type encodes an "Extended Vendor-Specific" attribute, as given in [RFC6929] Section 2.4. The "evs" data type is used solely to extend the Vendor Specific space. It MAY appear inside of an "extended" or a "long-extended" data type. It MUST NOT appear in the contents of any other data type.

Name

evs

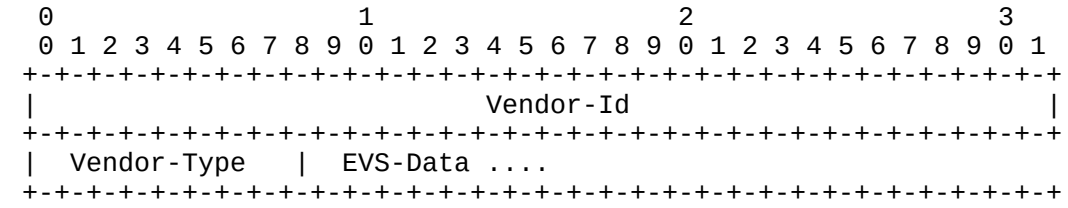
Number

17

Length

Six or more octets

Format



Subfields

Vendor-Id

The 4 octets are the Network Management Private Enterprise Code [PEN] of the Vendor in network byte order.

Vendor-Type

The Vendor-Type field is one octet. Values are assigned at the sole discretion of the Vendor.

EVS-Data

The EVS-Data field is one or more octets. It SHOULD encapsulate a previously defined RADIUS data type. Non-standard data types SHOULD NOT be used. We note that the EVS-Data field may be of data type "tlv".

The actual format of the information is site or application specific, and a robust implementation SHOULD support the field as undistinguished octets. We recognise that Vendors have complete control over the contents and format of the Ext-Data field, while at the same time recommending that good practices be followed.

Further codification of the range of allowed usage of this field is outside the scope of this specification.

3. Updated Registries

This section defines a new IANA registry for RADIUS data types, and updates the existing RADIUS Attribute Type registry.

3.1. Create a Data Type Registry

This section defines a new RADIUS registry, called "Data Type". Allocation in this registry requires IETF Review. The "Registration Procedures" for this registry are "Standards Action".

The registry contains three columns of data, as follows.

Value

The number of the data type. The value field is an artifact of the registry, and has no on-the-wire meaning.

Description

The name of the data type. The name field is used only for the registry, and has no on-the-wire meaning.

Reference

The specification where the data type was defined.

The initial contents of the registry are as follows.

Value	Description	Reference
1	integer	[RFC2865], TBD
2	enum	[RFC2865], TBD
3	ipv4addr	[RFC2865], TBD
4	time	[RFC2865], TBD
5	text	[RFC2865], TBD
6	string	[RFC2865], TBD
7	concat	TBD
8	ifid	[RFC3162], TBD
9	ipv6addr	[RFC3162], TBD
10	ipv6prefix	[RFC3162], TBD
11	ipv4prefix	[RFC6572], TBD
12	integer64	[RFC6929], TBD
13	tlv	[RFC6929], TBD
14	evs	[RFC6929], TBD
15	extended	[RFC6929], TBD
16	long-extended	[RFC6929], TBD

3.2. Updates to the Attribute Type Registry

This section updates the RADIUS Attribute Type Registry to have a new column, which is inserted in between the existing "Description" and "Reference" columns. The new column is named "Data Type". The contents of that column are the name of a data type, corresponding to the attribute in that row, or blank if the attribute type is unassigned. The name of the data type is taken from the RADIUS Data Type registry, defined above.

The updated registry follows in CSV format.

```
Value,Description,Data Type,Reference
1,User-Name,string,[RFC2865]
2,User-Password,string,[RFC2865]
3,CHAP-Password,string,[RFC2865]
4,NAS-IP-Address,ipv4addr,[RFC2865]
5,NAS-Port,integer,[RFC2865]
6,Service-Type,enum,[RFC2865]
7,Framed-Protocol,enum,[RFC2865]
8,Framed-IP-Address,ipv4addr,[RFC2865]
9,Framed-IP-Netmask,ipv4addr,[RFC2865]
10,Framed-Routing,enum,[RFC2865]
11,Filter-Id,text,[RFC2865]
12,Framed-MTU,integer,[RFC2865]
13,Framed-Compression,enum,[RFC2865]
14,Login-IP-Host,ipv4addr,[RFC2865]
15,Login-Service,enum,[RFC2865]
16,Login-TCP-Port,integer,[RFC2865]
17,Unassigned,,
18,Reply-Message,text,[RFC2865]
19,Callback-Number,text,[RFC2865]
20,Callback-Id,text,[RFC2865]
21,Unassigned,,
22,Framed-Route,text,[RFC2865]
23,Framed-IPX-Network,ipv4addr,[RFC2865]
24,State,string,[RFC2865]
25,Class,string,[RFC2865]
26,Vendor-Specific,vsa,[RFC2865]
27,Session-Timeout,integer,[RFC2865]
28,Idle-Timeout,integer,[RFC2865]
29,Termination-Action,enum,[RFC2865]
30,Called-Station-Id,text,[RFC2865]
31,Calling-Station-Id,text,[RFC2865]
32,NAS-Identifier,text,[RFC2865]
33,Proxy-State,string,[RFC2865]
34,Login-LAT-Service,text,[RFC2865]
35,Login-LAT-Node,text,[RFC2865]
```

```
36,Login-LAT-Group,string,[RFC2865]
37,Framed-AppleTalk-Link,integer,[RFC2865]
38,Framed-AppleTalk-Network,integer,[RFC2865]
39,Framed-AppleTalk-Zone,text,[RFC2865]
40,Acct-Status-Type,enum,[RFC2866]
41,Acct-Delay-Time,integer,[RFC2866]
42,Acct-Input-Octets,integer,[RFC2866]
43,Acct-Output-Octets,integer,[RFC2866]
44,Acct-Session-Id,text,[RFC2866]
45,Acct-Authentic,enum,[RFC2866]
46,Acct-Session-Time,integer,[RFC2866]
47,Acct-Input-Packets,integer,[RFC2866]
48,Acct-Output-Packets,integer,[RFC2866]
49,Acct-Terminate-Cause,enum,[RFC2866]
50,Acct-Multi-Session-Id,text,[RFC2866]
51,Acct-Link-Count,integer,[RFC2866]
52,Acct-Input-Gigawords,integer,[RFC2869]
53,Acct-Output-Gigawords,integer,[RFC2869]
54,Unassigned,,
55,Event-Timestamp,time,[RFC2869]
56,Egress-VLANID,integer,[RFC4675]
57,Ingress-Filters,enum,[RFC4675]
58,Egress-VLAN-Name,text,[RFC4675]
59,User-Priority-Table,string,[RFC4675]
60,CHAP-Challenge,string,[RFC2865]
61,NAS-Port-Type,enum,[RFC2865]
62,Port-Limit,integer,[RFC2865]
63,Login-LAT-Port,text,[RFC2865]
64,Tunnel-Type,enum,[RFC2868]
65,Tunnel-Medium-Type,enum,[RFC2868]
66,Tunnel-Client-Endpoint,text,[RFC2868]
67,Tunnel-Server-Endpoint,text,[RFC2868]
68,Acct-Tunnel-Connection,text,[RFC2867]
69,Tunnel-Password,text,[RFC2868]
70,ARAP-Password,string,[RFC2869]
71,ARAP-Features,string,[RFC2869]
72,ARAP-Zone-Access,enum,[RFC2869]
73,ARAP-Security,integer,[RFC2869]
74,ARAP-Security-Data,text,[RFC2869]
75,Password-Retry,integer,[RFC2869]
76,Prompt,enum,[RFC2869]
77,Connect-Info,text,[RFC2869]
78,Configuration-Token,text,[RFC2869]
79,EAP-Message,concat,[RFC2869]
80,Message-Authenticator,string,[RFC2869]
81,Tunnel-Private-Group-ID,text,[RFC2868]
82,Tunnel-Assignment-ID,text,[RFC2868]
83,Tunnel-Preference,integer,[RFC2868]
```

```
84,ARAP-Challenge-Response,string,[RFC2869]
85,Acct-Interim-Interval,integer,[RFC2869]
86,Acct-Tunnel-Packets-Lost,integer,[RFC2867]
87,NAS-Port-Id,text,[RFC2869]
88,Framed-Pool,text,[RFC2869]
89,CUI,string,[RFC4372]
90,Tunnel-Client-Auth-ID,text,[RFC2868]
91,Tunnel-Server-Auth-ID,text,[RFC2868]
92,NAS-Filter-Rule,text,[RFC4849]
93,Unassigned,,
94,Originating-Line-Info,string,[RFC7155]
95,NAS-IPv6-Address,ipv6addr,[RFC3162]
96,Framed-Interface-Id,ifid,[RFC3162]
97,Framed-IPv6-Prefix,ipv6prefix,[RFC3162]
98,Login-IPv6-Host,ipv6addr,[RFC3162]
99,Framed-IPv6-Route,text,[RFC3162]
100,Framed-IPv6-Pool,text,[RFC3162]
101,Error-Cause Attribute,enum,[RFC3576]
102,EAP-Key-Name,string,[RFC4072][RFC7268]
103,Digest-Response,text,[RFC5090]
104,Digest-Realm,text,[RFC5090]
105,Digest-Nonce,text,[RFC5090]
106,Digest-Response-Auth,text,[RFC5090]
107,Digest-Nextnonce,text,[RFC5090]
108,Digest-Method,text,[RFC5090]
109,Digest-URI,text,[RFC5090]
110,Digest-Qop,text,[RFC5090]
111,Digest-Algorithm,text,[RFC5090]
112,Digest-Entity-Body-Hash,text,[RFC5090]
113,Digest-CNonce,text,[RFC5090]
114,Digest-Nonce-Count,text,[RFC5090]
115,Digest-Username,text,[RFC5090]
116,Digest-Opaque,text,[RFC5090]
117,Digest-Auth-Param,text,[RFC5090]
118,Digest-AKA-Auts,text,[RFC5090]
119,Digest-Domain,text,[RFC5090]
120,Digest-Stale,text,[RFC5090]
121,Digest-HA1,text,[RFC5090]
122,SIP-AOR,text,[RFC5090]
123,Delegated-IPv6-Prefix,ipv6prefix,[RFC4818]
124,MIP6-Feature-Vector,string,[RFC5447]
125,MIP6-Home-Link-Prefix,ipv6prefix,[RFC5447]
126,Operator-Name,text,[RFC5580]
127,Location-Information,string,[RFC5580]
128,Location-Data,string,[RFC5580]
129,Basic-Location-Policy-Rules,string,[RFC5580]
130,Extended-Location-Policy-Rules,string,[RFC5580]
131,Location-Capable,enum,[RFC5580]
```

```
132,Requested-Location-Info,enum,[RFC5580]
133,Framed-Management-Protocol,enum,[RFC5607]
134,Management-Transport-Protection,enum,[RFC5607]
135,Management-Policy-Id,text,[RFC5607]
136,Management-Privilege-Level,integer,[RFC5607]
137,PKM-SS-Cert,concat,[RFC5904]
138,PKM-CA-Cert,concat,[RFC5904]
139,PKM-Config-Settings,string,[RFC5904]
140,PKM-Cryptosuite-List,string,[RFC5904]
141,PKM-SAID,text,[RFC5904]
142,PKM-SA-Descriptor,string,[RFC5904]
143,PKM-Auth-Key,string,[RFC5904]
144,DS-Lite-Tunnel-Name,text,[RFC6519]
145,Mobile-Node-Identifier,string,[RFC6572]
146,Service-Selection,text,[RFC6572]
147,PMIP6-Home-LMA-IPv6-Address,ipv6addr,[RFC6572]
148,PMIP6-Visited-LMA-IPv6-Address,ipv6addr,[RFC6572]
149,PMIP6-Home-LMA-IPv4-Address,ipv4addr,[RFC6572]
150,PMIP6-Visited-LMA-IPv4-Address,ipv4addr,[RFC6572]
151,PMIP6-Home-HN-Prefix,ipv6prefix,[RFC6572]
152,PMIP6-Visited-HN-Prefix,ipv6prefix,[RFC6572]
153,PMIP6-Home-Interface-ID,ifid,[RFC6572]
154,PMIP6-Visited-Interface-ID,ifid,[RFC6572]
155,PMIP6-Home-IPv4-HoA,ipv4prefix,[RFC6572]
156,PMIP6-Visited-IPv4-HoA,ipv4prefix,[RFC6572]
157,PMIP6-Home-DHCP4-Server-Address,ipv4addr,[RFC6572]
158,PMIP6-Visited-DHCP4-Server-Address,ipv4addr,[RFC6572]
159,PMIP6-Home-DHCP6-Server-Address,ipv6addr,[RFC6572]
160,PMIP6-Visited-DHCP6-Server-Address,ipv6addr,[RFC6572]
161,PMIP6-Home-IPv4-Gateway,ipv4addr,[RFC6572]
162,PMIP6-Visited-IPv4-Gateway,ipv4addr,[RFC6572]
163,EAP-Lower-Layer,enum,[RFC6677]
164,GSS-Acceptor-Service-Name,text,[RFC7055]
165,GSS-Acceptor-Host-Name,text,[RFC7055]
166,GSS-Acceptor-Service-Specifics,text,[RFC7055]
167,GSS-Acceptor-Realm-Name,text,[RFC7055]
168,Framed-IPv6-Address,ipv6addr,[RFC6911]
169,DNS-Server-IPv6-Address,ipv6addr,[RFC6911]
170,Route-IPv6-Information,ipv6prefix,[RFC6911]
171,Delegated-IPv6-Prefix-Pool,text,[RFC6911]
172,Stateful-IPv6-Address-Pool,text,[RFC6911]
173,IPv6-6rd-Configuration,tlv,[RFC6930]
174,Allowed-Called-Station-Id,text,[RFC7268]
175,EAP-Peer-Id,string,[RFC7268]
176,EAP-Server-Id,string,[RFC7268]
177,Mobility-Domain-Id,integer,[RFC7268]
178,Preauth-Timeout,integer,[RFC7268]
179,Network-Id-Name,string,[RFC7268]
```

```
180,EAPoL-Announcement,concat,[RFC7268]
181,WLAN-HESSID,text,[RFC7268]
182,WLAN-Venue-Info,integer,[RFC7268]
183,WLAN-Venue-Language,string,[RFC7268]
184,WLAN-Venue-Name,text,[RFC7268]
185,WLAN-Reason-Code,integer,[RFC7268]
186,WLAN-Pairwise-Cipher,integer,[RFC7268]
187,WLAN-Group-Cipher,integer,[RFC7268]
188,WLAN-AKM-Suite,integer,[RFC7268]
189,WLAN-Group-Mgmt-Cipher,integer,[RFC7268]
190,WLAN-RF-Band,integer,[RFC7268]
191,Unassigned,,
192-223,Experimental Use,,[RFC3575]
224-240,Implementation Specific,,[RFC3575]
241,Extended-Attribute-1,extended,[RFC6929]
241.{1-25},Unassigned,,
241.26,Extended-Vendor-Specific-1,evs,[RFC6929]
241.{27-240},Unassigned,,
241.{241-255},Reserved,,[RFC6929]
242,Extended-Attribute-2,extended,[RFC6929]
242.{1-25},Unassigned,,
242.26,Extended-Vendor-Specific-2,evs,[RFC6929]
242.{27-240},Unassigned,,
242.{241-255},Reserved,,[RFC6929]
243,Extended-Attribute-3,extended,[RFC6929]
243.{1-25},Unassigned,,
243.26,Extended-Vendor-Specific-3,evs,[RFC6929]
243.{27-240},Unassigned,,
243.{241-255},Reserved,,[RFC6929]
244,Extended-Attribute-4,extended,[RFC6929]
244.{1-25},Unassigned,,
244.26,Extended-Vendor-Specific-4,evs,[RFC6929]
244.{27-240},Unassigned,,
244.{241-255},Reserved,,[RFC6929]
245,Extended-Attribute-5,long-extended,[RFC6929]
245.{1-25},Unassigned,,
245.26,Extended-Vendor-Specific-5,evs,[RFC6929]
245.{27-240},Unassigned,,
245.{241-255},Reserved,,[RFC6929]
246,Extended-Attribute-6,long-extended,[RFC6929]
246.{1-25},Unassigned,,
246.26,Extended-Vendor-Specific-6,evs,[RFC6929]
246.{27-240},Unassigned,,
246.{241-255},Reserved,,[RFC6929]
247-255,Reserved,,[RFC3575]
```

4. Suggestions for Specifications

We suggest that these data types be used in new RADIUS specifications. Attributes can usually be completely described through their Attribute Type code, name, and data type. The use of "ASCII art" is then limited only to the definition of new data types, and complex data types.

Use of the new extended attributes [RFC6929] makes ASCII art even more problematic. An attribute can be allocated from the standard space, or from one of the extended spaces. This allocation decision is made after the specification has been accepted for publication. That allocation strongly affects the format of the attribute header, making it nearly impossible to create the correct ASCII art prior to final publication. Allocation from the different spaces also changes the value of the Length field, also making it difficult to define it correctly prior to final publication of the document.

The following fields SHOULD be given when defining new attributes:

Description

A description of the meaning and interpretation of the attribute.

Type

The Attribute Type code, given in the "dotted number" notation from [RFC6929]. Specifications can often leave this as "TBD", and request that IANA fill in the allocated values.

Length

A description of the length of the attribute. For attributes of variable length, a maximum length SHOULD be given.

Data Type

One of the named data types from the RADIUS Data Type registry.

Value

A description of any attribute-specific limitations on the values carried by the specified data type. If there are no attribute-specific limitations, then the description of this field can be omitted.

Where the values are limited to a subset of the possible range, valid range(s) MUST be defined.

For attributes of data type "enum", a list of enumerated values and names MUST be given, as with [RFC2865] Section 5.6.

5. Security Considerations

This specification is concerned solely with updates to IANA registries. As such, there are no security considerations with the document itself.

However, the use of inconsistent names and poorly-defined entities in a protocol is problematic. Inconsistencies in specifications can lead to security and interoperability problems in implementations. Further, having one canonical source for the definition of data types means an implementor has fewer specifications to read. The implementation work is therefore simpler, and is more likely to be correct.

The goal of this specification is to reduce ambiguities in the RADIUS protocol, which we believe will lead to more robust and more secure implementations.

6. IANA Considerations

IANA is instructed to create one new registry as described above in Section 3.1. The "TBD" text in that section should be replaced with the RFC number of this document when it is published.

IANA is instructed to update the RADIUS Attribute Type registry, as described above in Section 3.2.

IANA is instructed to require that all allocation requests in the RADIUS Attribute Type Registry contain a "data type" field. That field is required to contain one of the "data type" names contained in the RADIUS Data Type registry.

7. References

7.1. Normative References

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

[RFC2865]
Rigney, C., Willens, S., Rubens, A. and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, June 2000.

[RFC3629]
Yergeau, F., "UTF-8, a transformation format of ISO 10646", RFC 3629, November 2003.

[RFC6158]
DeKok, A., and Weber, G., "RADIUS Design Guidelines", RFC 6158, March 2011.

[RFC6572]
Xia, F., et al, "RADIUS Support for Proxy Mobile IPv6", RFC 6572, June 2012.

7.2. Informative References

[RFC2868]
Zorn, G., Leifer, D., Rubens, A., Shriver, J., Holdrege, M., and I. Goyret, "RADIUS Attributes for Tunnel Protocol Support", RFC 2868, June 2000.

[RFC2869]
Rigney, C., et al, "RADIUS Extensions", RFC 2869, June 2000.

[RFC3162]
Aboba, B., Zorn, G., and D. Mitton, "RADIUS and IPv6", RFC 3162, August 2001.

[RFC6929]
DeKok, A., and Lior, A., "Remote Authentication Dial In User Service (RADIUS) Protocol Extensions", RFC 6929, April 2013.

[PEN]
http://www.iana.org/assignments/enterprise-numbers

Acknowledgments

Stuff

Authors' Addresses

Alan DeKok
The FreeRADIUS Server Project

Email: aland@freeradius.org

Network Working Group
Internet-Draft
Updates: 6613, 6614 (if approved)
Intended status: Experimental
Expires: September 7, 2015

S. Hartman
Painless Security

March 6, 2015

Larger Packets for RADIUS over TCP
draft-ietf-radext-bigger-packets-03.txt

Abstract

The RADIUS over TLS experiment described in RFC 6614 has opened RADIUS to new use cases where the 4096-octet maximum RADIUS packet proves problematic. This specification extends the RADIUS over TCP experiment (RFC 6113) to permit larger RADIUS packets. This specification compliments other ongoing work to permit fragmentation of RADIUS authorization information. This document registers a new RADIUS code, an action which requires IESG approval.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 7, 2015.

Copyright Notice

Table of Contents

1. Introduction

The Remote Access Dial-In User Server (RADIUS) over TLS [RFC6614] experiment provides strong confidentiality and integrity for RADIUS [RFC2865]. This enhanced security has opened new opportunities for using RADIUS to convey additional authorization information. As an example, [I-D.ietf-abfab-aaa-saml] describes a mechanism for using RADIUS to carry Security Assertion Markup Language (SAML) messages in RADIUS. Many attributes carried in these SAML messages will require confidentiality or integrity such as that provided by TLS.

These new use cases involve carrying additional information in RADIUS packets. The maximum packet length of 4096 octets is proving insufficient for some SAML messages and for other structures that may be carried in RADIUS.

One approach is to fragment a RADIUS message across multiple packets at the RADIUS layer. RADIUS Fragmentation [I-D.ietf-radext-radius-fragmentation] provides a mechanism to split authorization information across multiple RADIUS messages. That mechanism is necessary in order to split authorization information across existing unmodified proxies.

However, there are some significant disadvantages to RADIUS fragmentation. First, RADIUS is a lock-step protocol, and only one

fragment can be in transit at a time as part of a given request. Also, there is no current mechanism to discover the path Maximum Transmission Unit (MTU) across the entire path that the fragment will travel. As a result, fragmentation is likely both at the RADIUS layer and at the transport layer. When TCP is used, much better transport characteristics can be achieved by fragmentation only at the TCP layer. This specification provides a mechanism to achieve these better transport characteristics when TCP is used. As part of this specification, a new RADIUS code is registered.

This specification is published as an experimental specification because the TCP extensions to RADIUS are currently experimental. The need for this specification arises from operational experience with the TCP extensions. However, this specification introduces no new experimental evaluation criteria beyond those in the base TCP specification; this specification can be evaluated along with that one for advancement on the standards track.

1.1. Requirements notation

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 [RFC2119].

2. Changes to Packet Processing

The maximum length of a RADIUS message is increased from 4096 to 65535. A RADIUS Server implementing this specification MUST be able to receive a packet of maximum length. Servers MAY have a maximum size over which they choose to return an error as discussed in Section 5 rather than processing a received packet; this size MUST be at least 4096 octets.

Clients implementing this specification MUST be able to receive a packet of maximum length; that is clients MUST NOT close a TCP connection simply because a large packet is sent over it. Clients MAY include the Response-Length attribute defined in Section 6 to indicate the maximum size of a packet that they can successfully process. Clients MAY silently discard a packet greater than some configured size; this size MUST be at least 4096 octets. Clients MUST NOT retransmit an unmodified request whose response is larger than the client can process as subsequent responses will likely continue to be too large.

Proxies MUST be able to receive a packet of maximum length without cloing the TCP connection. Proxies SHOULD be able to process and forward packets of maximum length. When a proxy receives a request over a transport with a 4096-octet maximum length and the proxy

forwards that request over a transport with a larger maximum length, the proxy MUST include the Response-Length attribute with a value of 4096.

2.1. Status-Server Considerations

This section extends processing of Status-Server messages as described in section 4.1 and 4.2 of [RFC5997].

Clients implementing this specification SHOULD include the Response-Length attribute in Status-Server requests. Servers are already required to ignore unknown attributes received in this message. by including the attribute the client indicates how large of a response it can process to its Status-Server request. It is very unlikely that a response to Status-Server is greater than 4096 octets. However the client also indicates support for this specification which triggers server behavior below.

If a server implementing this specification receives a Response-Length attribute in a Status-Server request, it MUST include a Response-Length attribute indicating the maximum size request it can process in its response to the Status-Server request.

3. Forward and backward Compatibility

An implementation of [RFC6613] will silently discard any packet larger than 4096 octets and will close the TCP connection. This section provides guidelines for interoperability with these implementations. These guidelines are stated at the SHOULD level. In some environments support for large packets will be important enough that roaming or other agreements will mandate their support. In these environments, all implementations might be required to support this specification removing the need for interoperability with RFC 6613. It is likely that these guidelines will be relaxed to the MAY level and support for this specification made a requirement if RADIUS over TLS and TCP are moved to the standards track in the future.

Clients SHOULD provide configuration for the maximum size of a request sent to each server. Servers SHOULD provide configuration for the maximum size of a response sent to each client. If dynamic discovery mechanisms are supported, configuration SHOULD be provided for the maximum size of clients and servers in each dynamic discovery category.

If a client sends a request larger than 4096 octets and the TCP connection is closed without a response, the client SHOULD treat the request as if a request too big error (Section 5) specifying a

maximum size of 4096 is received. Clients or proxies sending multiple requests over a single TCP connection without waiting for responses SHOULD implement capability discovery as discussed in Section 3.2.

By default, a server SHOULD not generate a response larger than 4096 octets. The Response-Length attribute MAY be included in a request to indicate that larger responses are acceptable. Other attributes or configuration MAY be used as an indicator that large responses are likely to be acceptable.

A proxy that implements both this specification and RADIUS Fragmentation [I-D.ietf-radext-radius-fragmentation] SHOULD use RADIUS fragmentation when the following conditions are met:

1. A packet is being forwarded towards an endpoint whose configuration does not support a packet that large.

2. RADIUS Fragmentation can be used for the packet in question.

3.1. Rationale

The interoperability challenge appears at first significant. This specification proposes to introduce behavior where new implementations will fail to function with existing implementations.

However, these capabilities are introduced to support new use cases. If an implementation has 10000 octets of attributes to send, it cannot in general trim down the response to something that can be sent. Under this specification a large packet would be generated that will be silently discarded by an existing implementation. Without this specification, no packet is generated because the required attributes cannot be sent.

The biggest risk to interoperability would be if requests and responses are expanded to include additional information that is not strictly necessary. So, avoiding creating situations where large packets are sent to existing implementations is mostly an operational matter. Interoperability is most impacted when the size of packets in existing use cases is significantly increased and least impacted when large packets are used for new use cases where the deployment is likely to require updated RADIUS implementations.

There is a special challenge for proxies or clients with high request volume. When an RFC 6113 implementation receives a packet that is too large, it closes the connection and does not respond to any requests in process. Such a client would lose requests and might find distinguishing request-too-big situations from other failures

difficult. In these cases, the discovery mechanism described in Section 3.2 can be used.

Also, RFC 6613 is an experiment. Part of running that experiment is to evaluate whether additional changes are required to RADIUS. A lower bar for interoperability should apply to changes to experimental protocols than standard protocols.

This specification provides good facilities to enable implementations to understand packet size when proxying to/from standards-track UDP RADIUS.

3.2. Discovery

As discussed in Section 2.1, a client MAY send a Status-Server message to discover whether an authentication or accounting server supports this specification. The client includes a Response-Length attribute; this signals the server to include a Response-Length attribute indicating the maximum packet size the server can process. In this one instance, Response-Length indicate the size of a request that can be processed rather than a response.

4. Protocol-Error Code

This document defines a new RADIUS code, TBDCODE (IANA), called Protocol-Error. This packet code may be used in response to any request packet, such as Access-Request, Accounting-Request, CoA-Request, or Disconnect-Request. It is a response packet sent by a server to a client. The packet indicates to the client that the server is unable to process the request for some reason.

A Protocol-Error packet MUST contain a Original-Packet-Code attribute, along with an Error-Cause attribute. Other attributes MAY be included if desired. The Original-Packet-Code contains the code from the request that generated the protocol error so that clients can disambiguate requests with different codes and the same ID. Regardless of the original packet code, the RADIUS server calculates the Message-Authenticator attribute as if the original packet were an Access-Request packet. The identifier is copied from the original request.

Clients processing Protocol-Error MUST ignore unknown or unexpected attributes.

This RADIUS code is hop-by-hop. Proxies MUST not forward a Protocol-Error packet they receive.

5. Too Big Response

When a RADIUS server receives a request that is larger than can be processed, it generates a Protocol-Error response as follows:

The code is Protocol-Error.

The Response-Length attribute MUST be included and its value is the maximum size of request that will be processed.

The Error-Cause attribute is included with a value of TOOBIGTBD.

The Original-Packet-Code attribute is copied from the request.

Clients will not typically be able to adjust and resend requests when this error is received. In some cases the client can fall back to RADIUS Fragmentation. In other cases this code will provide for better client error reporting and will avoid retransmitting requests guaranteed to fail.

6. IANA Considerations

A new RADIUS packet type code is registered in the RADIUS packet type codes registry discussed in section 2.1 of RFC 3575 [RFC3575]. The name is "Protocol-Error" and the code is TBDCODE. The IESG is requested to approve this registration along with approving publication of this document.

The following RADIUS attribute type values [RFC3575] are assigned. The assignment rules in section 10.3 of [RFC6929] are used.

Name	Attribute	Description
Response-Length	TBD	2-octet unsigned integer maximum response length
Original-Packet-Code	TBD2	An integer attribute containing the code from a packet resulting in a Protocol-Error response.

The Response-Length attribute MAY be included in any RADIUS request. In this context it indicates the maximum length of a response the client is prepared to receive. Values are between 4096 and 65535. The attribute MAY also be included in a response to a Status-Server

message. In this case the attribute indicate the maximum size RADIUS request that is permitted.

A new Error-Cause value is registered in the registry at http://www.iana.org/assignments/radius-types/radius-types.xhtml#radius-types-18 for "Response Too Big" with value TOOBIGTBD.

7. Security Considerations

This specification updates RFC 6613 and will be used with [RFC6614]. When used over plain TCP, this specification creates new opportunities for an on-path attacker to impact availability. These attacks can be entirely mitigated by using TLS. If these attacks ar acceptable, then this specification can be used over TCP.

8. Acknowledgements

Sam Hartman’s time on this draft was funded by JANET as part of Project Moonshot.

Alan DeKok provided valuable review and text for the Protocol-Error packet code.

Alejandro Perez Mendez provided valuable review comments.

9. References

9.1. Normative References

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

[RFC2865] Rigney, C., Willens, S., Rubens, A., and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, June 2000.

[RFC3575] Aboba, B., "IANA Considerations for RADIUS (Remote Authentication Dial In User Service)", RFC 3575, July 2003.

[RFC5997] DeKok, A., "Use of Status-Server Packets in the Remote Authentication Dial In User Service (RADIUS) Protocol", RFC 5997, August 2010.

[RFC6613] DeKok, A., "RADIUS over TCP", RFC 6613, May 2012.

[RFC6614] Winter, S., McCauley, M., Venaas, S., and K. Wierenga, "Transport Layer Security (TLS) Encryption for RADIUS", RFC 6614, May 2012.

[RFC6929] DeKok, A. and A. Lior, "Remote Authentication Dial In User Service (RADIUS) Protocol Extensions", RFC 6929, April 2013.

9.2. References

[I-D.ietf-abfab-aaa-saml]
Howlett, J. and S. Hartman, "A RADIUS Attribute, Binding, Profiles, Name Identifier Format, and Confirmation Methods for SAML", draft-ietf-abfab-aaa-saml-09 (work in progress), February 2014.

[I-D.ietf-radext-radius-fragmentation]
Perez-Mendez, A., Lopez, R., Pereniguez-Garcia, F., Lopez-Millan, G., Lopez, D., and A. DeKok, "Support of fragmentation of RADIUS packets", draft-ietf-radext-radius-fragmentation-06 (work in progress), April 2014.

Author’s Address

Sam Hartman
Painless Security

Email: hartmans-ietf@mit.edu

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 10, 2015

D. Cheng
Huawei
J. Korhonen
Broadcom
M. Boucadair
France Telecom
S. Sivakumar
Cisco Systems

February 6, 2015

RADIUS Extensions for IP Port Configuration and Reporting
draft-ietf-radext-ip-port-radius-ext-03

Abstract

This document defines three new RADIUS attributes. For devices that implementing IP port ranges, these attributes are used to communicate with a RADIUS server in order to configure and report TCP/UDP ports and ICMP identifiers, as well as mapping behavior for specific hosts. This mechanism can be used in various deployment scenarios such as CGN (Carrier Grade NAT), NAT64, Provider WLAN Gateway, etc.

Requirements Language

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 RFC 2119 [RFC2119].

Status of This Memo

Table of Contents

1. Introduction

In a broadband network, customer information is usually stored on a RADIUS server [RFC2865] and at the time when a user initiates an IP connection request, the RADIUS server will populate the user's configuration information to the Network Access Server (NAS), which is usually co-located with the Border Network Gateway (BNG), after the connection request is granted. The Carrier Grade NAT (CGN) function may also be implemented on the BNG, and therefore the CGN TCP/UDP port (or ICMP identifier) mapping(s) behavior(s) can be configured on the RADIUS server as part of the user profile, and populated to the NAS in the same manner. In addition, during the operation, the CGN can also convey port/identifier mapping behavior specific to a user to the RADIUS server, as part of the normal RADIUS accounting process.

The CGN device that communicates with a RADIUS server using RADIUS extensions defined in this document may perform NAT44 [RFC3022], NAT64 [RFC6146], or Dual-Stack Lite AFTR [RFC6333] function.

For the CGN case, when IP packets traverse a CGN device, it would perform TCP/UDP source port mapping or ICMP identifier mapping as required. A TCP/ UDP source port or ICMP identifier, along with source IP address, destination IP address, destination port and protocol identifier if applicable, uniquely identify a session. Since the number space of TCP/UDP ports and ICMP identifiers in CGN's external realm is shared among multiple users assigned with the same IPv4 address, the total number of a user's simultaneous IP sessions is likely to be subject to port quota (see Section 5 of [RFC6269]).

The attributes defined in this document may also be used to report the assigned port range in some deployments such as Provider WLAN [I-D.gundavelli-v6ops-community-wifi-svcs]. For example, a visiting host can be managed by a CPE (Customer Premises Equipment) which will need to report the assigned port range to the service platform. This is required for identification purposes (see TR-146 [TR-146] for example).

This document proposes three new attributes as RADIUS protocol's extensions, and they are used for separate purposes as follows:

1. IP-Port-Limit: This attribute may be carried in RADIUS Acces-Accept, Access-Request, Accounting-Request or CoA-Request packet. The purpose of this attribute is to limit the total number of TCP/UDP ports and/or ICMP identifiers that an IP subscriber can use, associated with one or more IPv4 addresses.

2. IP-Port-Range: This attribute may be carried in RADIUS Accounting-Request packet. The purpose of this attribute is to report by an address sharing device (e.g., a CGN) to the RADIUS server the range of TCP/UDP ports and/or ICMP identifiers that have been allocated or deallocated associated with a given IPv4 address for a subscriber.

3. IP-Port-Forwarding-Map: This attribute may be carried in RADIUS Access-Accept, Access-Request, Accounting-Request or CoA-Request packet. The purpose of this attribute is to specify how a TCP/UDP port (or an ICMP identifier) mapping to another TCP/UDP port (or an ICMP identifier), and each is associated with its respective IPv4 address.

This document leverages the protocol defined in [RFC7012] by proposing a mapping between type field of RADIUS TLV and Element ID of IPFIX. It also proposes a few new IPFIX Elements as required by this document (see Section 3).

This document was constructed using the [RFC2629].

2. Terminology

This document makes use if the following terms:

- IP Port: refers to the port numbers of IP transport protocols, including TCP port, UDP port and ICMP identifier.

- IP Port Type: refers to one of the following: (1) TCP/UDP port and ICMP identifier, (2) TCP port and UDP port, (3) TCP port, (4) UDP port, or (5) ICMP identifier.

- IP Port Limit: denotes the maximum number of IP ports for a specific IP port type, that a device supporting port ranges can use when performing port number mapping for a specific user. Note, this limit is usually associated with one or more IPv4 addresses.

- IP Port Range: specifies a set of contiguous IP ports, indicated by the smallest numerical number and the largest numerical number, inclusively.

- Internal IP Address: refers to the IP address that is used as a source IP address in an outbound IP packet sent towards a device supporting port ranges in the internal realm. In the IPv4 case, it is typically a private address [RFC1918].

- External IP Address: refers to the IP address that is used as a source IP address in an outbound IP packet after traversing a device supporting port ranges in the external realm. In the IPv4 case, it is typically a global routable IP address.

- Internal Port: is a UDP or TCP port, or an ICMP identifier, which is allocated by a host or application behind a device supporting port ranges for an outbound IP packet in the internal realm.

- External Port: is a UDP or TCP port, or an ICMP identifier, which is allocated by a device supporting port ranges upon receiving an outbound IP packet in the internal realm, and is used to replace the internal port that is allocated by a user or application.

- External realm: refers to the networking segment where IPv4 public addresses are used in respective of the device supporting port ranges.

- Internal realm: refers to the networking segment that is behind a device supporting port ranges and where IPv4 private addresses are used.

- Mapping: associates with a device supporting port ranges for a relationship between an internal IP address, internal port and the protocol, and an external IP address, external port, and the protocol.

- Port-based device: a device that is capable of providing IP address and IP port mapping services and in particular, with the granularity of one or more subsets within the 16-bit IP port number range. A typical example of this device is a CGN, CPE, Provider WLAN Gateway, etc.

Note the terms "internal IP address", "internal port", "internal realm", "external IP address", "external port", "external realm", and "mapping" and their semantics are the same as in [RFC6887], and [RFC6888].

3. Extensions of RADIUS Attributes and TLVs

These three new attributes are defined in the following sub-sections:

1. IP-Port-Limit Attribute

2. IP-Port-Range Attribute

3. IP-Port-Forwarding-Map Attribute

All these attributes are allocated from the RADIUS "Extended Type" code space per [RFC6929].

3.1. Extended Attributes for IP Ports

3.1.1. Extended-Type

This section defines a new Extended-Type (see Figure 1).

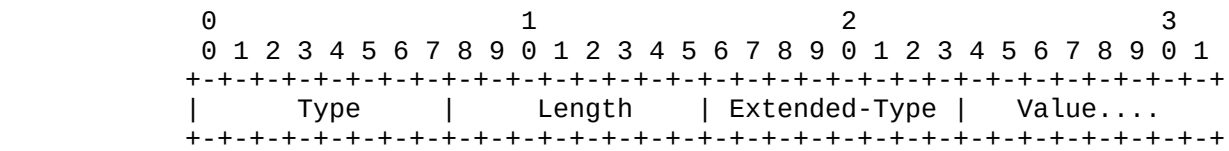


Figure 1

Type:

TBA1 - Extended-Type-1 (241), Extended-Type-2 (242), Extended-Type-3 (243), or Extended-Type-4 (244) per [RFC6929].

Length:

Indicates the total length in bytes of all fields of this attribute, including Type, Length, Extended-Type, and the embedded TLVs.

Extended-Type:

TBA2.

The interpretation of this field is determined by the identifier of "TBA1.TBA2..." along with the embedded TLVs.

3.1.2. IP-Port-Limit Attribute

This attribute contains the Extended-Type defined in Section 3.1.1, along with a set of embedded TLVs defined in Section 3.2.1 (IP-Port-Type TLV), Section 3.2.2 (IP-Port-Limit TLV), and Section 3.2.3 (IP-Port-Ext-IPv4-Addr TLV). It specifies the maximum number of IP ports as indicated in IP-Port-Limit TLV, of a specific port type as

indicated in IP-Port-Type TLV, and associated with a given IPv4 address as indicated in IP-Port-Ext-IPv4-Addr TLV for an end user.

Note that when IP-Port-Ext-IPv4-Addr TLV is not included as part of the IP-Port-Limit Attribute, the port limit is applied to all the IPv4 addresses managed by the port device, e.g., a CGN or NAT64 device.

The IP-Port-Limit Attribute MAY appear in an Access-Accept packet. It MAY also appear in an Access-Request packet as a hint by the device supporting port ranges, which is co-allocated with the NAS, to the RADIUS server as a preference, although the server is not required to honor such a hint.

The IP-Port-Limit Attribute MAY appear in a CoA-Request packet.

The IP-Port-Limit Attribute MAY appear in an Accounting-Request packet.

The IP-Port-Limit Attribute MUST NOT appear in any other RADIUS packets.

The format of the IP-Port-Limit Attribute is shown in Figure 2. The fields are transmitted from left to right.

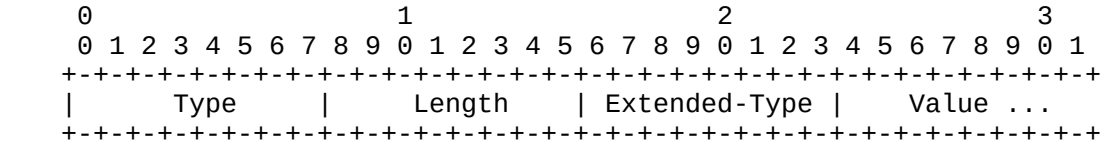


Figure 2

Type:

TBA1 - Extended-Type-1 (241), Extended-Type-2 (242), Extended-Type-3 (243), or Extended-Type-4 (244) per [RFC6929].

Length:

This field indicates the total length in bytes of all fields of this attribute, including the Type, Length, Extended-Type, and the entire length of the embedded TLVs.

Extended-Type:

TBA2.

Value:

This field contains a set of TLVs as follows:

IP-Port-Type TLV:

This TLV contains a value that indicates the IP port type. Refer to Section 3.2.1.

IP-Port-Limit TLV:

This TLV contains the maximum number of IP ports of a specific IP port type and associated with a given IPv4 address for an end user. This TLV must be included in the IP-Port-Limit Attribute. Refer to Section 3.2.2.

IP-Port-Ext-IPv4-Addr TLV:

This TLV contains the IPv4 address that is associated with the IP port limit contained in the IP-Port-Limit TLV. This TLV is optionally included as part of the IP-Port-Limit Attribute. Refer to Section 3.2.3.

IP-Port-Limit attribute is associated with the following identifier: Type(TBA1).Extended-Type(TBA2).IP-Port-Type TLV(TBA3).[IP-Port-Limit TLV(TBA4), {IP-Port-Ext-IPv4-Addr TLV(TBA5)}].

3.1.3. IP-Port-Range Attribute

This attribute contains the Extended-Type defined in Section 3.1.1, along with a set of embedded TLVs defined in Section 3.2.1(IP-Port-Type TLV), Section 3.2.9(IP-Port-Range-Start TLV), Section 3.2.10 (IP-Port-Range-End TLV), Section 3.2.8 (IP-Port-Alloc TLV), Section 3.2.3 (IP-Port-Ext-IPv4-Addr TLV), and Section 3.2.11 (IP-Port-Local-Id TLV).

This attribute contains a range of contiguous IP ports of a specific port type and associated with an IPv4 address that are either allocated or deallocated by a device for a given subscriber, and the information is intended to send to RADIUS server.

This attribute can be used to convey a single IP port number; in such case IP-Port-Range-Start and IP-Port-Range-End conveys the same value.

Within an IP-Port-Range Attribute, the IP-Port-Alloc TLV is always included. For port allocation, both IP-Port-Range-Start TLV and IP-Port-Range-End TLV must be included; for port deallocation, the

inclusion of these two TLVs is optional and if not included, it implies that all ports that are previously allocated are now deallocated. Both IP-Port-Ext-IPv4-Addr TLV and IP-Port-Local-Id TLV are optional and if included, they are used by a port device (e.g., a CGN device) to identify the end user.

The IP-Port-Range Attribute MAY appear in an Accounting-Request packet.

The IP-Port-Range Attribute MUST NOT appear in any other RADIUS packets.

The format of the IP-Port-Range Attribute format is shown in Figure 3. The fields are transmitted from left to right.

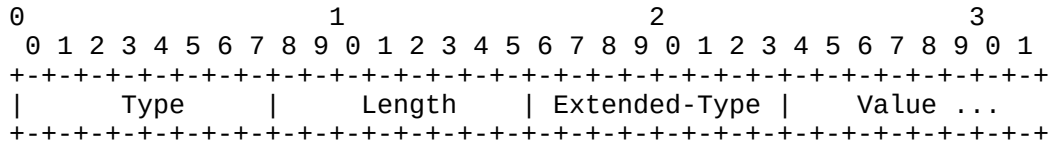


Figure 3

Type:

TBA1 - Extended-Type-1 (241), Extended-Type-2 (242), Extended-Type-3 (243), or Extended-Type-4 (244) per [RFC6929]

Length:

This field indicates the total length in bytes of all fields of this attribute, including the Type, Length, Extended-Type, and the entire length of the embedded TLVs.

Extended-Type:

TBA2.

Value:

This field contains a set of TLVs as follows:

IP-Port-Type TLV:

This TLV contains a value that indicates the IP port type. Refer to Section 3.2.1.

IP-Port-Alloc TLV:

This TLV contains a flag to indicate that the range of the specified IP ports for either allocation or deallocation. This TLV must be included as part of the IP-Port-Range Attribute. Refer to Section 3.2.8.

IP-Port-Range-Start TLV:

This TLV contains the smallest port number of a range of contiguous IP ports. To report the port allocation, this TLV must be included together with IP-Port-Range-End TLV as part of the IP-Port-Range Attribute. Refer to Section 3.2.9.

IP-Port-Range-End TLV:

This TLV contains the largest port number of a range of contiguous IP ports. To report the port allocation, this TLV must be included together with IP-Port-Range-Start TLV as part of the IP-Port-Range Attribute. Refer to Section 3.2.10.

IP-Port-Ext-IPv4-Addr TLV:

This TLV contains the IPv4 address that is associated with the IP port range, as collectively indicated in the IP-Port-Range-Start TLV and the IP-Port-Range-End TLV. This TLV is optionally included as part of the IP-Port-Range Attribute. Refer to Section 3.2.3.

IP-Port-Local-Id TLV:

This TLV contains a local session identifier at the customer premise, such as MAC address, interface ID, VLAN ID, PPP sessions ID, VRF ID, IPv6 address/prefix, etc. This TLV is optionally included as part of the IP-Port-Range Attribute. Refer to Section 3.2.11.

The IP-Port-Range attribute is associated with the following identifier: Type(TBA1).Extended-Type(TBA2).IP-Port-Type TLV(TBA3).[IP-Port-Alloc TLV(TBA10), {IP-Port-Range-Start TLV(TBA11), IP-Port-Range-End TLV(TBA12)}, {IP-Port-Ext-IPv4-Addr TLV (TBA5)}, {IP-Port-Local-Id TLV (TBA13)}].

3.1.4. IP-Port-Forwarding-Map Attribute

This attribute contains the Extended-Type defined in Section 3.1.1, along with a set of embedded TLVs defined in Section 3.2.1(IP-Port-Type TLV), Section 3.2.6(IP-Port-Int-Port TLV), Section 3.2.7(IP-

Port-Ext-Port TLV), Section 3.2.4(IP-Port-Int-IPv4-Addr TLV) or Section 3.2.5(IP-Port-Int-IPv6-Addr TLV), Section 3.2.11(IP-Port-Local-Id TLV) and Section 3.2.3 (IP-Port-Ext-IP-Addr TLV).

The attribute contains a 2-byte IP internal port number that is associated with an internal IPv4 or IPv6 address, or a locally significant identifier at the customer site, and a 2-byte IP external port number that is associated with an external IPv4 address. The internal IPv4 or IPv6 address, or the local identifier must be included; the external IPv4 address may also be included.

The IP-Port-Forwarding-Map Attribute MAY appear in an Access-Accept packet. It MAY also appear in an Access-Request packet as a hint by the device supporting port mapping, which is co-allocated with the NAS, to the RADIUS server as a preference, although the server is not required to honor such a hint.

The IP-Port-Forwarding-Map Attribute MAY appear in a CoA-Request packet.

The IP-Port-Forwarding-Map Attribute MAY also appear in an Accounting-Request packet.

The attribute MUST NOT appear in any other RADIUS packet.

The format of the IP-Port-Forwarding-Map Attribute is shown in Figure 4. The fields are transmitted from left to right.

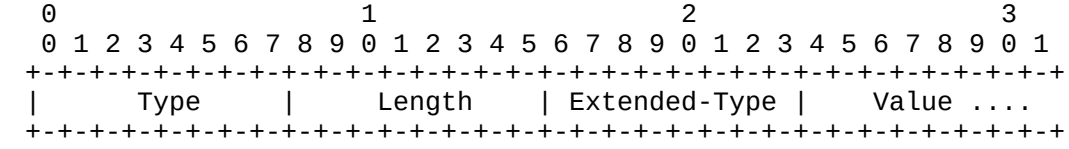


Figure 4

Type:

TBA1 - Extended-Type-1 (241), Extended-Type-2 (242), Extended-Type-3 (243), or Extended-Type-4 (244) per [RFC6929]

Length:

This field indicates the total length in bytes of all fields of this attribute, including the Type, Length, Extended-Type, and the entire length of the embedded TLVs.

Extended-Type:

TBA2.

Value:

This field contains a set of TLVs as follows:

IP-Port-Type TLV:

This TLV contains a value that indicates the IP port type. Refer to Section 3.2.1.

IP-Port-Int-Port TLV:

This TLV contains an internal IP port number associated with an internal IPv4 or IPv6 address. This TLV must be included together with IP-Port-Ext-Port TLV as part of the IP-Port-Forwarding-Map attribute. Refer to Section 3.2.6.

IP-Port-Ext-Port TLV:

This TLV contains an external IP port number associated with an external IPv4 address. This TLV must be included together with IP-Port-Int-Port TLV as part of the IP-Port-Forwarding-Map attribute. Refer to Section 3.2.7.

IP-Port-Int-IPv4-Addr TLV:

This TLV contains an IPv4 address that is associated with the internal IP port number contained in the IP-Port-Int-Port TLV. For IPv4 network, either this TLV or IP-Port-Local-Id TLV must be included as part of the IP-Port-Forwarding-Map Attribute. Refer to Section 3.2.4.

IP-Port-Int-IPv6-Addr TLV:

This TLV contains an IPv4 address that is associated with the internal IP port number contained in the IP-Port-Int-Port TLV. For IPv6 network, either this TLV or IP-Port-Local-Id TLV must be included as part of the IP-Port-Forwarding-Map Attribute. Refer to Section 3.2.5.

IP-Port-Local-Id TLV:

This TLV contains a local session identifier at the customer premise, such as MAC address, interface ID, VLAN ID, PPP sessions ID, VRF ID, IPv6 address/prefix, etc. Either this TLV

or IP-Port-Int-IP-Addr TLV must be included as part of the IP-Port-Forwarding-Map Attribute. Refer to Section 3.2.11.

IP-Port-Ext-IPv4-Addr TLV:

This TLV contains an IPv4 address that is associated with the external IP port number contained in the IP-Port-Ext-Port TLV. This TLV may be included as part of the IP-Port-Forwarding-Map Attribute. Refer to Section 3.2.3.

The IP-Port-Forwarding-Map attribute is associated with the following identifier: Type(TBA1).Extended-Type(TBA2).IP-Port-Type TLV(TBA3). [IP-Port-Int-Port TLV(TBA8), IP-Port-Ext-Port TLV(TBA9), {IP-Port-Int-IPv4-Addr TLV(TBA6) | IP-Port-Int-IPv6-Addr TLV(TBA7) }, {IP-Port-Ext-IPv4-Addr TLV(TBA5)}].

3.2. RADIUS TLVs for IP Ports

3.2.1. IP-Port-Type TLV

This TLV (Figure 5) uses the format defined in [RFC6929]. Its Type field contains a value that uniquely refers to IPFIX Element transportType (TBAx1), and its Value field contains IPFIX Element transportType, which indicates the type of IP transport type as follows:

1:

Refer to TCP port, UDP port, and ICMP identifier as a whole.

2:

Refer to TCP port and UDP port as a whole.

3:

Refer to TCP port only.

4:

Refer to UDP port only.

5:

Refer to ICMP identifier only.

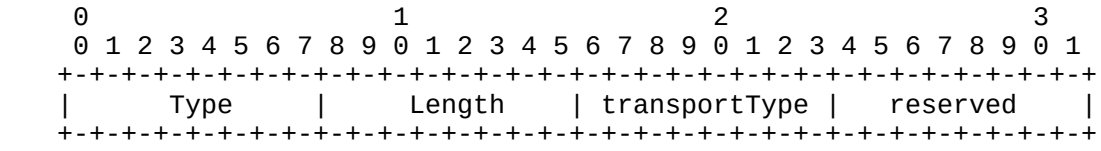


Figure 5

Type:

TBA3: This uniquely refers to IPFIX Element ID TBA0.

Length:

3.

transportType:

unsigned8.

3.2.2. IP-Port-Limit TLV

This TLV (Figure 6) uses the format defined in [RFC6929]. Its Type field contains a value that uniquely refers to IPFIX Element natTransportLimit (TBAx2), and its Value field contains IPFIX Element natTransportLimit, which indicates the maximum number of ports of a specified IP-Port-Type and associated with a given IPv4 address assigned to a subscriber (refer to [IPFIX])

Note that IP-Port-Limit TLV is embedded within IP-Port-Type TLV (refer to Section 3.2.1) for detail.

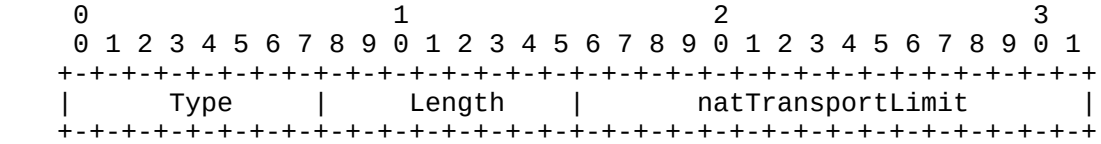


Figure 6

Type:

TBA4: This uniquely refers to IPFIX Element ID Limit TBD.

Length:

4.

natTransportLimit:

unsigned16.

3.2.3. IP-Port-Ext-IPv4-Addr TLV

This TLV (Figure 7) uses the format defined in[RFC6929]. Its Type field contains a value that uniquely refers to IPFIX Element postNATSourceIPv4Address(225), and its Value field contains IPFIX Element postNATSourceIPv4Address, which is the IPv4 source address after NAT operation (refer to [IPFIX]).

IP-Port-Ext-IPv4-Addr TLV can be included as part of the IP-Port-Limit Attribute (refer to Section 3.1.2), IP-Port-Range Attribute (refer to Section 3.1.3), and IP-Port-Forwarding-Map Attribute (refer to Section 3.1.4).

Note that IP-Port-Ext-IPv4-Addr TLV is embedded within IP-Port-Type TLV (refer to Section 3.2.1) for detail.

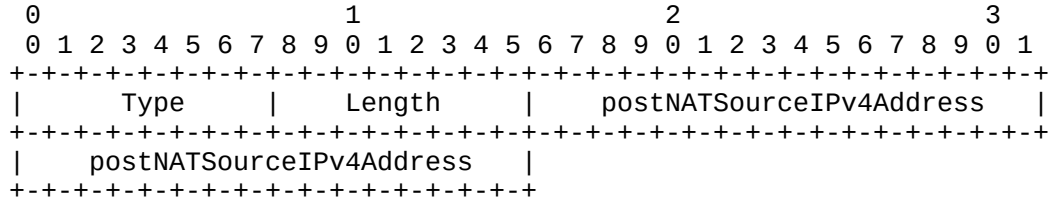


Figure 7

Type:

TBA5: The type field uniquely refers to the IPFIX Element ID 225.

Length:

6

postNATSourceIPv4Address:

ipv4Address.

3.2.4. IP-Port-Int-IPv4-Addr TLV

This TLV (Figure 8) uses format defined in [RFC6929]. Its Type field contains a value that uniquely refers to IPFIX Element sourceIPv4Address (8), and its Value field contains IPFIX Element sourceIPv4Address, which is the IPv4 source address before NAT operation (refer to [IPFIX]).

IP-Port-Int-IPv4-Addr TLV can be included as part of the IP-Port-Forwarding-Map Attribute (refer to Section 3.1.4).

Note that IP-Port-Int-IPv4-Addr TLV is embedded within IP-Port-Type TLV (refer to Section 3.2.1) for detail.

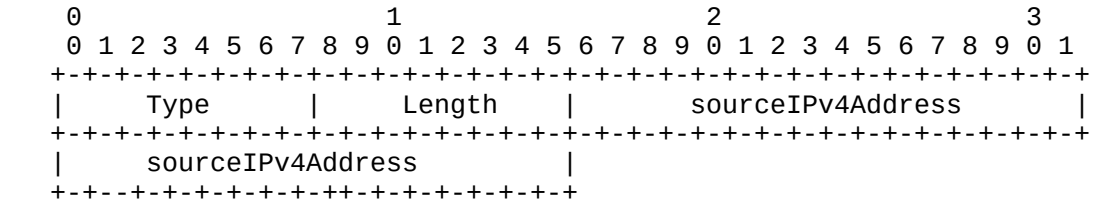


Figure 8

Type:

TBA6: The type field uniquely refers to the IPFIX Element ID 8.

Length:

6.

sourceIPv4Address:

unsigned16.

3.2.5. IP-Port-Int-IPv6-Addr TLV

This TLV (Figure 9) uses format defined in [RFC6929]. Its Type field contains a value that uniquely refers to IPFIX Element sourceIPv6Address(27), and its Value field contains IPFIX Element sourceIPv6Address, which is the IPv6 source address before NAT operation (refer to [IPFIX]).

IP-Port-Int-IPv6-Addr TLV can be included as part of the IP-Port-Forwarding-Map Attribute (refer to Section 3.1.4).

Note that IP-Port-Int-IPv6-Addr TLV is embedded within IP-Port-Type TLV (refer to Section 3.2.1) for detail.

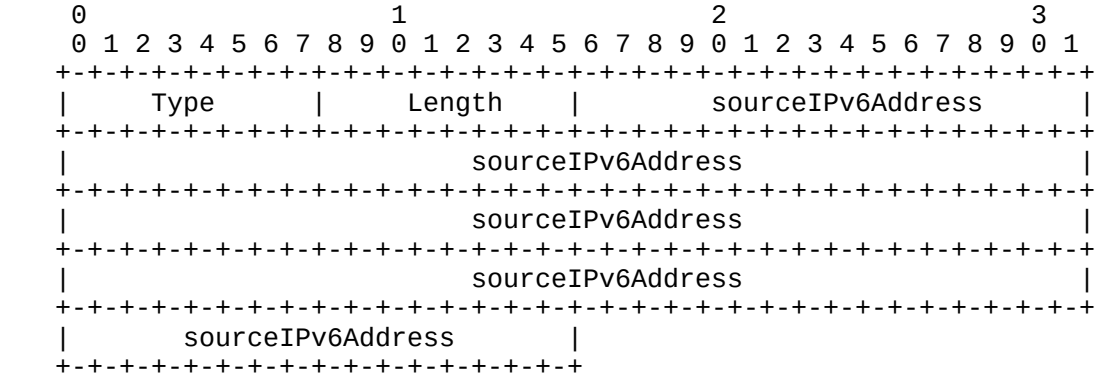


Figure 9

Type:

TBA7: The type field uniquely refers to the IPFIX Element ID 27.

Length:

18.

sourceIPv6Address:

unsigned64.

3.2.6. IP-Port-Int-Port TLV

This TLV (Figure 10) uses format defined in [RFC6929]. Its Type field contains a value that uniquely refers to IPFIX Element sourceTransportPort (7), and its Value field contains IPFIX Element sourceTransportPort, which is the source transport number associated with an internal IPv4 or IPv6 address (refer to [IPFIX]).

IP-Port-Int-Port TLV is included as part of the IP-Port-Forwarding-Map Attribute (refer to Section 3.1.4).

IP-Port-Int-Port TLV is embedded within embedded within IP-Port-Type TLV (refer to Section 3.2.1) for detail.

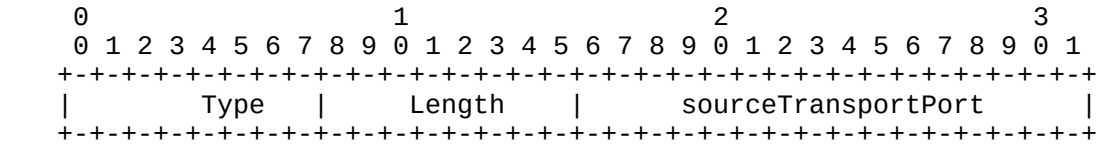


Figure 10

Type:

TBA8: This uniquely refers to the IPFIX Element ID 7.

Length:

4.

sourceTransportPort:

unsigned16.

3.2.7. IP-Port-Ext-Port TLV

This TLV (Figure 11) uses format defined in [RFC6929]. Its Type field contains a value that uniquely refers to IPFIX Element postNAPTSourceTransportPort (227), and its Value field contains IPFIX Element postNAPTSourceTransportPort, which is the transport number associated with an external IPv4 address(refer to [IPFIX]).

IP-Port-Ext-Port TLV is included as part of the IP-Port-Forwarding-Map Attribute (refer to Section 3.1.4).

IP-Port-Ext-Port TLV is embedded within IP-Port-Type TLV (refer to Section 3.2.1) for detail.

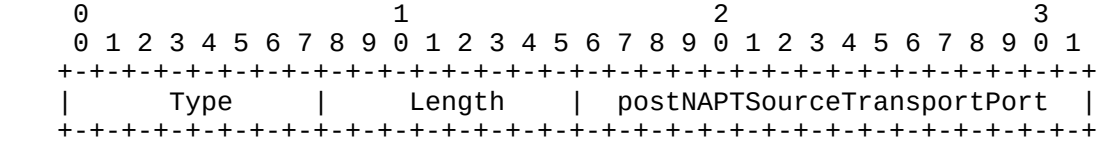


Figure 11

Type:

TBA9: This uniquely refers to the IPFIX Element ID 227 .

Length:

6.

postNAPTSourceTransportPort:

unsigned16.

3.2.8. IP-Port-Alloc TLV

This TLV (Figure 12) uses format defined in [RFC6929]. Its Type field contains a value that uniquely refers to IPFIX Element natEvent (230), and its Value field contains IPFIX Element "natEvent", which is a flag to indicate an action of NAT operation (refer to [IPFIX]).

When the value of natEvent is "1" (Create event), it means to allocate a range of transport ports; when the value is "2", it means to de-allocate a range of transports ports. For the purpose of this TLV, no other value is used.

IP-Port-Alloc TLV is included as part of the IP-Port-Range Attribute (refer to Section 3.1.3).

Note that IP-Port-Alloc TLV is embedded within IP-Port-Type TLV (refer to Section 3.2.1) for detail.

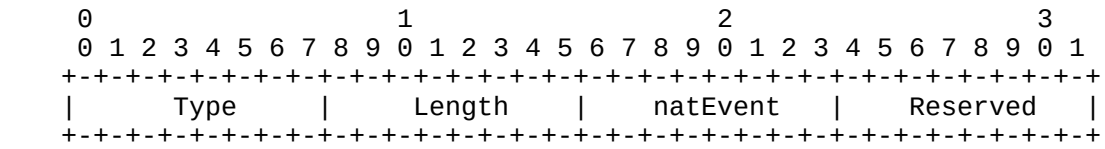


Figure 12

Type:

TBA10: This uniquely refers to the IPFIX Element ID 230 .

Length:

3.

natEvent:

unsigned8. This field indicates the allocation or deallocation of a range of IP ports as follows:

1:

Allocation

2:

Deallocation

Reserved:

0.

3.2.9. IP-Port-Range-Start TLV

This TLV (Figure 13) uses format defined in [RFC6929]. Its Type field contains a value that uniquely refers to IPFIX Element portRangeStart (361), and its Value field contains IPFIX Element portRangeStart, which is the smallest port number of a range of contiguous transport ports (refer to [IPFIX]).

IP-Port-Range-Start TLV is included as part of the IP-Port-Range Attribute (refer to Section 3.1.3).

Note that IP-Port-Range-Start TLV is embedded within IP-Port-Type TLV (refer to Section 3.1.1) for detail.

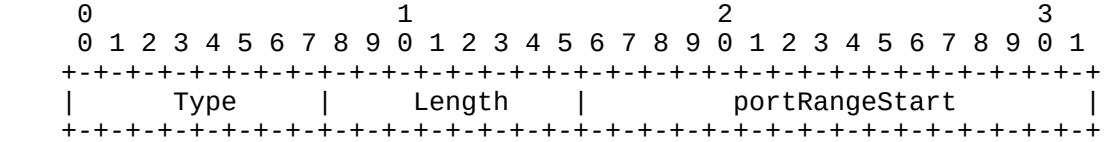


Figure 13

Type:

TBA11: This uniquely refers to the IPFIX Element ID 361.

TLV8-Length:

4.

portRangeStart:

unsigned16. This field contains the smallest port number of a range of contiguous IP transport ports.

3.2.10. IP-Port-Range-End TLV

This TLV (Figure 14) uses format defined in [RFC6929]. Its Type field contains a value that uniquely refers to IPFIX Element portRangeEnd (362), and its Value field contains IPFIX Element portRangeEnd, which is the largest port number of a range of contiguous transport ports (refer to [IPFIX]).

IP-Port-Range-End TLV is included as part of the IP-Port-Range Attribute (refer to Section 3.1.3).

Note that IP-Port-Range-End TLV is embedded within IP-Port-Type TLV (refer to Section 3.1.1) for detail.

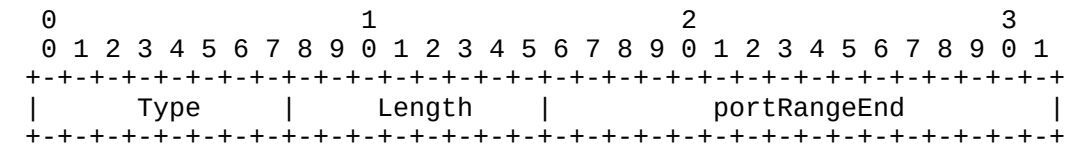


Figure 14

Type:

TBA12: This uniquely refers to IPFIC Element ID 362.

Length:

4. The Length field for IP-Port-Range-End TLV.

portRangeEnd:

unsigned16. This field contains the largest port number of a range of contiguous IP ports.

3.2.11. IP-Port-Local-Id TLV

This TLV (Figure 15) uses format defined in [RFC6929]. Its Type field contains a value that uniquely refers to IPFIX Element localID (TBAx3), and its Value field contains IPFIX Element localID, which is a local significant identifier as explained below.

In some CGN deployment scenarios such as L2NAT [I-D.miles-behave-l2nat], DS-Extra-Lite [RFC6619] and Lightweight 4over6 [I-D.ietf-softwire-lw4over6], parameters at a customer premise such as MAC address, interface ID, VLAN ID, PPP session ID, IPv6

prefix, VRF ID, etc., may also be required to pass to the RADIUS server as part of the accounting record.

IP-Port-Local-Id TLV can be included as part of the IP-Port-Range Attribute (refer to Section 3.1.3) and IP-Port-Forwarding-Map Attribute (refer to Section 3.1.4).

Note that IP-Port-Local-Id TLV is embedded within IP-Port-Type TLV (refer to Section 3.1.1) for detail.

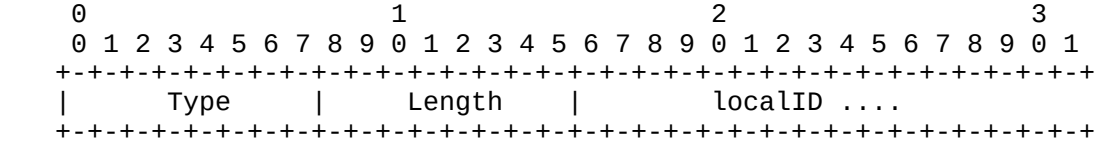


Figure 15

Type:

TBA13: This uniquely refers to IPFIX Element ID TBD.

Length:

Variable number of bytes.

localID:

string. This is a local session identifier at the customer premise, such as MAC address, interface ID, VLAN ID, PPP sessions ID, VRF ID, IPv6 address/prefix, etc.

4. Applications, Use Cases and Examples

This section describes some applications and use cases to illustrate the use of the attributes proposed in this document.

4.1. Managing CGN Port Behavior using RADIUS

In a broadband network, customer information is usually stored on a RADIUS server, and the BNG hosts the NAS. The communication between the NAS and the RADIUS server is triggered by a subscriber when the user signs in to the Internet service, where either PPP or DHCP/ DHCPv6 is used. When a user signs in, the NAS sends a RADIUS Access-Request message to the RADIUS server. The RADIUS server validates the request, and if the validation succeeds, it in turn sends back a RADIUS Access-Accept message. The Access-Accept message carries configuration information specific to that user, back to the NAS,

where some of the information would pass on to the requesting user via PPP or DHCP/DHCPv6.

A CGN function in a broadband network would most likely reside on a BNG. In that case, parameters for CGN port/identifier mapping behavior for users can be configured on the RADIUS server. When a user signs in to the Internet service, the associated parameters can be conveyed to the NAS, and proper configuration is accomplished on the CGN device for that user.

Also, CGN operation status such as CGN port/identifier allocation and de-allocation for a specific user on the BNG can also be transmitted back to the RADIUS server for accounting purpose using the RADIUS protocol.

RADIUS protocol has already been widely deployed in broadband networks to manage BNG, thus the functionality described in this specification introduces little overhead to the existing network operation.

In the following sub-sections, we describe how to manage CGN behavior using RADIUS protocol, with required RADIUS extensions proposed in Section 3.

4.1.1. Configure IP Port Limit for a User

In the face of IPv4 address shortage, there are currently proposals to multiplex multiple subscribers' connections over a smaller number of shared IPv4 addresses, such as Carrier Grade NAT [RFC6888], Dual-Stack Lite [RFC6333], NAT64 [RFC6146], etc. As a result, a single IPv4 public address may be shared by hundreds or even thousands of subscribers. As indicated in [RFC6269], it is therefore necessary to impose limits on the total number of ports available to an individual subscriber to ensure that the shared resource, i.e., the IPv4 address remains available in some capacity to all the subscribers using it, and port limiting is also documented in [RFC6888] as a requirement.

The IP port limit imposed to a specific subscriber may be on the total number of TCP and UDP ports plus the number of ICMP identifiers, or with other granularities as defined in Section 3.1.2.

The per-subscriber based IP port limit is configured on a RADIUS server, along with other user information such as credentials. The value of these IP port limit is based on service agreement and its specification is out of the scope of this document.

When a subscriber signs in to the Internet service successfully, the IP port limit for the subscriber is passed to the BNG based NAS,

where CGN also locates, using a new RADIUS attribute called IP-Port-Limit (defined in Section 3.1.2), along with other configuration parameters. While some parameters are passed to the subscriber, the IP port limit is recorded on the CGN device for imposing the usage of TCP/UDP ports and ICMP identifiers for that subscriber.

Figure 16 illustrates how RADIUS protocol is used to configure the maximum number of TCP/UDP ports for a given subscriber on a NAT44 device.

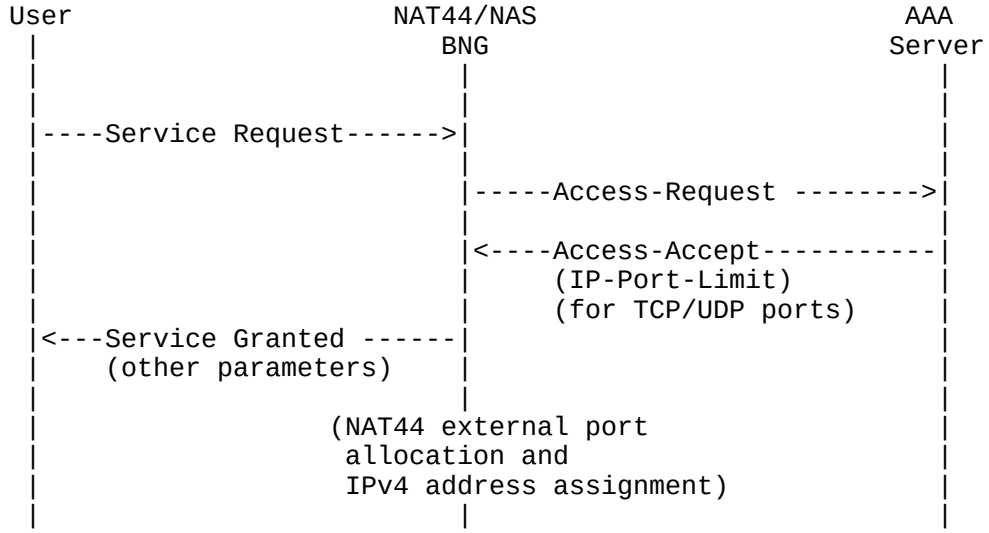


Figure 16: RADIUS Message Flow for Configuring NAT44 Port Limit

The IP port limit created on a CGN device for a specific user using RADIUS extension may be changed using RADIUS CoA message [RFC5176] that carries the same RADIUS attribute. The CoA message may be sent from the RADIUS server directly to the NAS, which once accepts and sends back a RADIUS CoA ACK message, the new IP port limit replaces the previous one.

Figure 17 illustrates how RADIUS protocol is used to increase the TCP/UDP port limit from 1024 to 2048 on a NAT44 device for a specific user.

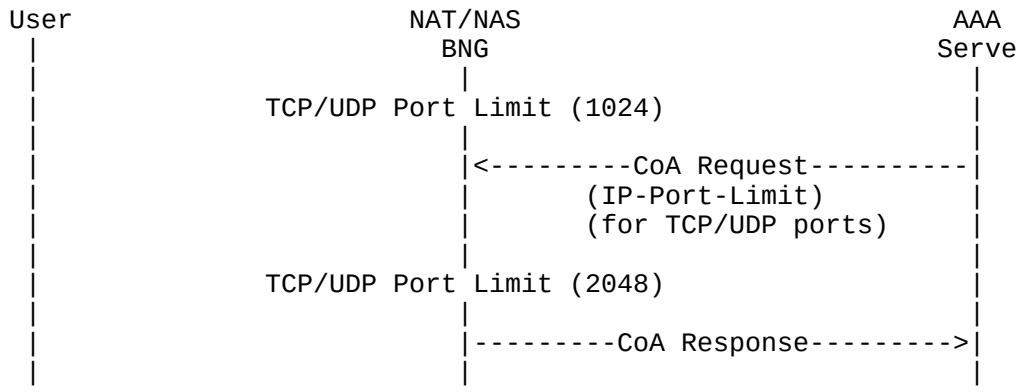


Figure 17: RADIUS Message Flow for changing a user's NAT44 port limit

4.1.2. Report IP Port Allocation/De-allocation

Upon obtaining the IP port limit for a subscriber, the CGN device needs to allocate a TCP/UDP port or an ICMP identifiers for the subscriber when receiving a new IP flow sent from that subscriber.

As one practice, a CGN may allocate a bulk of TCP/UDP ports or ICMP identifiers once at a time for a specific user, instead of one port/identifier at a time, and within each port bulk, the ports/identifiers may be randomly distributed or in consecutive fashion. When a CGN device allocates bulk of TCP/UDP ports and ICMP identifiers, the information can be easily conveyed to the RADIUS server by a new RADIUS attribute called the IP-Port-Range (defined in Section 3.1.3). The CGN device may allocate one or more TCP/UDP port ranges or ICMP identifier ranges, or generally called IP port ranges, where each range contains a set of numbers representing TCP/UDP ports or ICMP identifiers, and the total number of ports/identifiers must be less or equal to the associated IP port limit imposed for that subscriber. A CGN device may choose to allocate a small port range, and allocate more at a later time as needed; such practice is good because its randomization in nature.

At the same time, the CGN device also needs to decide the shared IPv4 address for that subscriber. The shared IPv4 address and the pre-allocated IP port range are both passed to the RADIUS server.

When a subscriber initiates an IP flow, the CGN device randomly selects a TCP/UDP port or ICMP identifier from the associated and pre-allocated IP port range for that subscriber to replace the original source TCP/UDP port or ICMP identifier, along with the replacement of the source IP address by the shared IPv4 address.

A CGN device may decide to "free" a previously assigned set of TCP/ UDP ports or ICMP identifiers that have been allocated for a specific subscriber but not currently in use, and with that, the CGN device must send the information of the de-allocated IP port range along with the shared IPv4 address to the RADIUS server.

Figure 18 illustrates how RADIUS protocol is used to report a set of ports allocated and de-allocated, respectively, by a NAT44 device for a specific user to the RADIUS server.

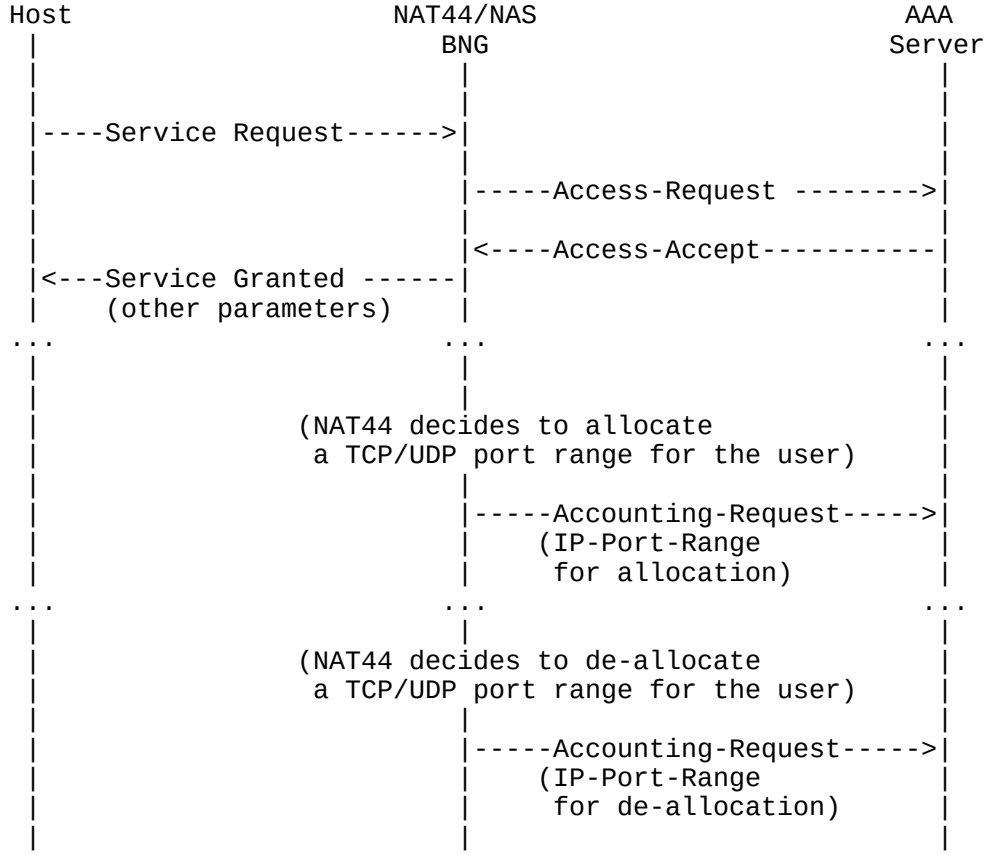


Figure 18: RADIUS Message Flow for reporting NAT44 allocation/de-allocation of a port set

4.1.3. Configure Forwarding Port Mapping

In most scenarios, the port mapping on a NAT device is dynamically created when the IP packets of an IP connection initiated by a user arrives. For some applications, the port mapping needs to be pre-

defined allowing IP packets of applications from outside a CGN device to pass through and "port forwarded" to the correct user located behind the CGN device.

Port Control Protocol [RFC6887], provides a mechanism to create a mapping from an external IP address and port to an internal IP address and port on a CGN device just to achieve the "port forwarding" purpose. PCP is a server-client protocol capable of creating or deleting a mapping along with a rich set of features on a CGN device in dynamic fashion. In some deployment, all users need is a few, typically just one pre-configured port mapping for applications such as web cam at home, and the lifetime of such a port mapping remains valid throughout the duration of the customer's Internet service connection time. In such an environment, it is possible to statically configure a port mapping on the RADIUS server for a user and let the RADIUS protocol to propagate the information to the associated CGN device.

Figure 19 illustrates how RADIUS protocol is used to configure a forwarding port mapping on a NAT44 device by using RADIUS protocol.

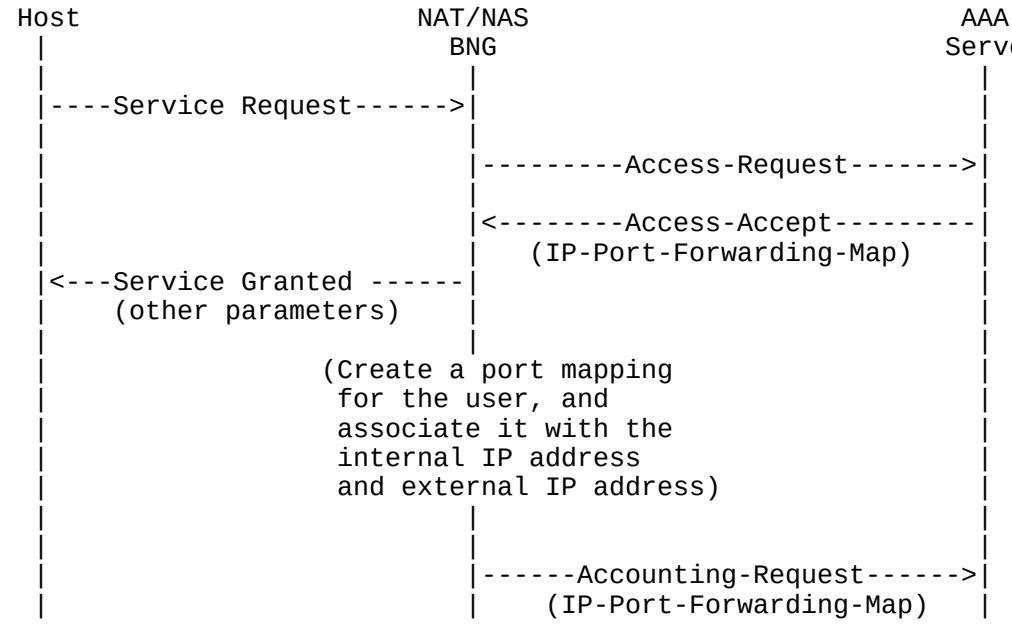


Figure 19: RADIUS Message Flow for configuring a forwarding port mapping

A port forwarding mapping that is created on a CGN device using RADIUS extension as described above may also be changed using RADIUS

CoA message [RFC5176] that carries the same RADIUS associate. The CoA message may be sent from the RADIUS server directly to the NAS, which once accepts and sends back a RADIUS CoA ACK message, the new port forwarding mapping then replaces the previous one.

Figure 20 illustrates how RADIUS protocol is used to change an existing port mapping from (a:X) to (a:Y), where "a" is an internal port, and "X" and "Y" are external ports, respectively, for a specific user with a specific IP address

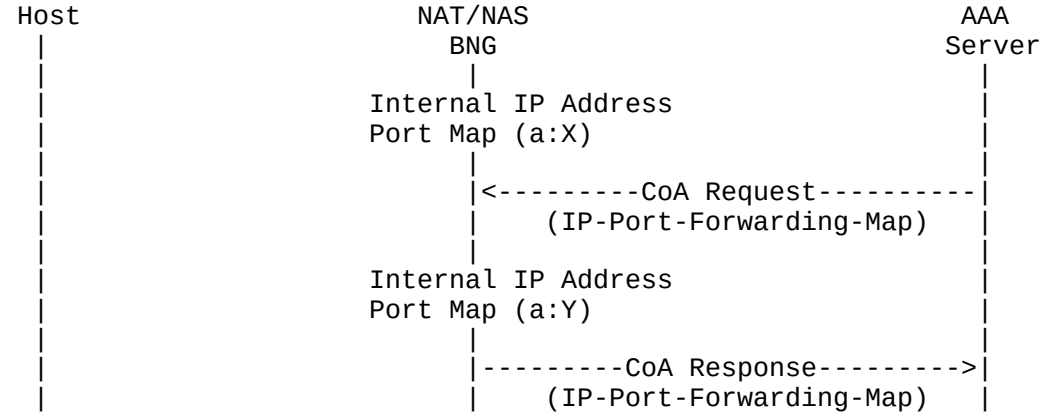


Figure 20: RADIUS Message Flow for changing a user’s forwarding port mapping

4.1.4. An Example

An Internet Service Provider (ISP) assigns TCP/UDP 500 ports for the subscriber Joe. This number is the limit that can be used for TCP/UDP ports on a NAT44 device for Joe, and is configured on a RADIUS server. Also, Joe asks for a pre-defined port forwarding mapping on the NAT44 device for his web cam applications (external port 5000 maps to internal port 80).

When Joe successfully connects to the Internet service, the RADIUS server conveys the TCP/UDP port limit (1000) and the forwarding port mapping (external port 5000 to internal port 80) to the NAT44 device, using IP-Port-Limit attribute and IP-Port-Forwarding-Map attribute, respectively, carried by an Access-Accept message to the BNG where NAS and CGN co-located.

Upon receiving the first outbound IP packet sent from Joe’s laptop, the NAT44 device decides to allocate a small port pool that contains 40 consecutive ports, from 3500 to 3540, inclusively, and also assign a shared IPv4 address 192.0.2.15, for Joe. The NAT44 device also

randomly selects one port from the allocated range (say 3519) and use that port to replace the original source port in outbound IP packets.

For accounting purpose, the NAT44 device passes this port range (3500-3540) and the shared IPv4 address 192.0.2.15 together to the RADIUS server using IP-Port-Range attribute carried by an Accounting-Request message.

When Joe works on more applications with more outbound IP sessions and the port pool (3500-3540) is close to exhaust, the NAT44 device allocates a second port pool (8500-8800) in a similar fashion, and also passes the new port range (8500-8800) and IPv4 address 192.0.2.15 together to the RADIUS server using IP-Port-Range attribute carried by an Accounting-Request message. Note when the CGN allocates more ports, it needs to assure that the total number of ports allocated for Joe is within the limit.

Joe decides to upgrade his service agreement with more TCP/UDP ports allowed (up to 1000 ports). The ISP updates the information in Joe's profile on the RADIUS server, which then sends a CoA-Request message that carries the IP-Port-Limit attribute with 1000 ports to the NAT44 device; the NAT44 device in turn sends back a CoA-ACK message. With that, Joe enjoys more available TCP/UDP ports for his applications.

When Joe travels, most of the IP sessions are closed with their associated TCP/UDP ports released on the NAT44 device, which then sends the relevant information back to the RADIUS server using IP-Port-Range attribute carried by Accounting-Request message.

Throughout Joe's connection with his ISP Internet service, applications can communicate with his web cam at home from external realm directly traversing the pre-configured mapping on the CGN device.

When Joe disconnects from his Internet service, the CGN device will de-allocate all TCP/UDP ports as well as the port-forwarding mapping, and send the relevant information to the RADIUS server.

4.2. Report Assigned Port Set for a Visiting UE

Figure 21 illustrates an example of the flow exchange which occurs when a visiting UE connects to a CPE offering WLAN service.

For identification purposes (see [RFC6967]), once the CPE assigns a port set, it issues a RADIUS message to report the assigned port set.

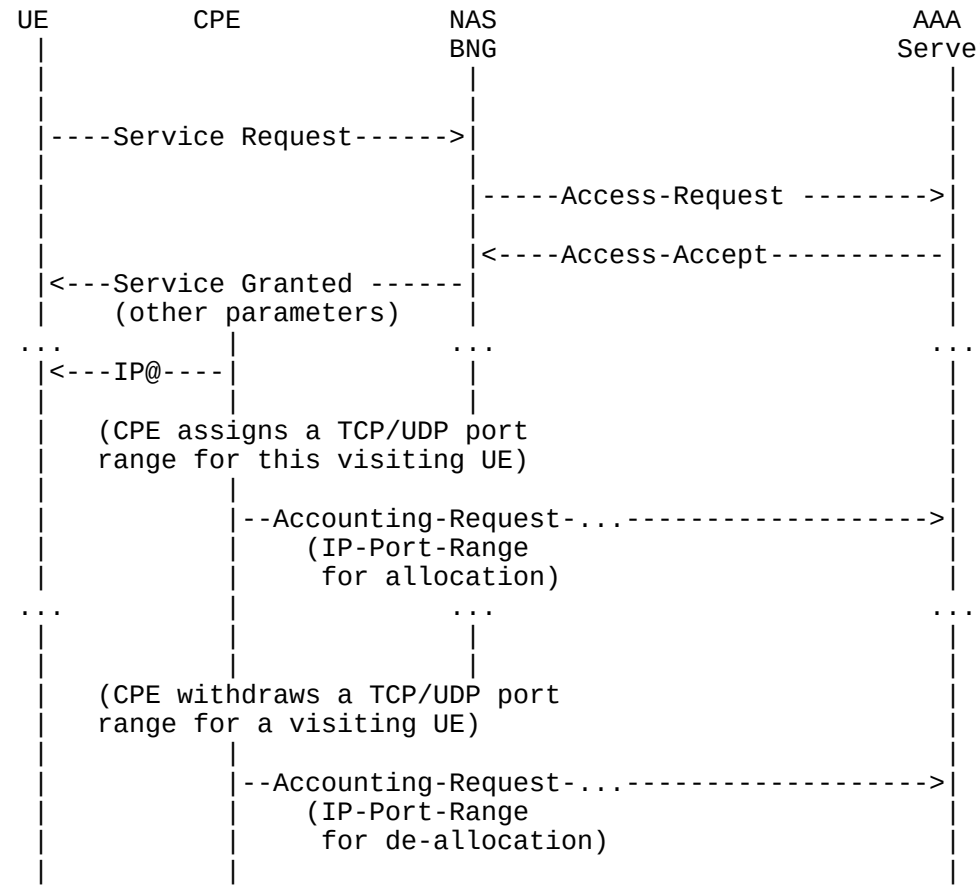


Figure 21: RADIUS Message Flow for reporting CPE allocation/de-allocation of a port set to a visiting UE

5. Table of Attributes

This document proposes three new RADIUS attributes and their formats are as follows:

- IP-Port-Limit: TBA1.TBA2.TBA3.[TBA4, {TBA5}]
- IP-Port-Range: TBA1.TBA2.TBA3.[TBA10, {TBA11, TBA12}, {TBA5}, {TBA13}].
- IP-Port-Forwarding-Map: TBA1.TBA2.TBA3.[TBA8, TBA9, {TBA6 | TBA7}, {TBA5}]

The following table provides a guide as what type of RADIUS packets that may contain these attributes, and in what quantity.

Request	Accept	Reject	Challenge	Acct. Request	#	Attribute
0+	0+	0	0	0+	TBA	IP-Port-Limit
0	0	0	0	0+	TBA	IP-Port-Range
0+	0+	0	0	0+	TBA	IP-Port-Forwarding-Map

The following table defines the meaning of the above table entries.

0 This attribute MUST NOT be present in packet.
0+ Zero or more instances of this attribute MAY be present in packet.

6. Security Considerations

This document does not introduce any security issue than what has been identified in [RFC2865].

7. IANA Considerations

This document requires new code point assignments for both IPFIX Elements and RADIUS attributes as explained in the following sections.

7.1. IANA Considerations on New IPFIX Elements

The following are code point assignments for new IPFIX Elements as requested by this document:

- transportType (refer to Section 3.2.1): The identifier of this IPFIX Element is TBAx1. The data type of this IPFIX Element is unsigned8, and the Element’s value indicates TCP/UDP ports and ICMP Identifiers (1), TCP/UDP ports (2), TCP ports (3), UDP ports (4) or ICMP identifiers (5).

- natTransportLimit (refer to Section 3.2.2): The identifier of this IPFIX Element is TBAx2. The data type of this IPFIX Element is unsigned16, and the Element’s value is the max number of IP transport ports to be assigned to an end user associated with one or more IPv4 addresses.

- localID (refer to Section 3.2.11): The identifier of this IPFIX Element is TBAx3. The data type of this IPFIX Element is string, and the Element’s value is an IPv4 or IPv6 address, a MAC address, a VLAN ID, etc.

7.2. IANA Considerations on New RADIUS Attributes

The following are new code point assignment for RADIUS extensions as requested by this document

- TBA1 (refer to Section 3.1.1): This value is for the Radius Type field and should be allocated from the number space of Extended-Type-1 (241), Extended-Type-2 (242), Extended-Type-3 (243), or Extended-Type-4 (244) per [RFC6929].

- TBA2 (refer to Section 3.1.1): This value is for the Extended-Type field and should be allocated from the Short Extended Space per [RFC6929].

- TBA3 (refer to Section 3.2.1): This value is for the Type field of IP-Port-Type TLV. It should be allocated as TLV data type. It is within the TBA2 container and it extends the attribute tree as TBA1.TBA2.TBA3.[...]. Also, this value uniquely refers to IPFIX Element ID transportType (TBAx1).

- TBA4 (refer to Section 3.2.2): This value is for the Type field of IP-Port-Limit TLV. It should be allocated as TLV data type and it extends the attribute tree as TBA1.TBA2.TBA3.[TBA4...]. Also, this value uniquely refers to IPFIX Element ID natTransportLimit(TBAx2).

- TBA5 (refer to Section 3.2.3): This value is for the Type field of IP-Port-Ext-IPv4-Addr TLV. It should be allocated as TLV data type and it extends the attribute tree as TBA1.TBA2.TBA3.[..TBA5...]. Also, this value uniquely refers to IPFIX Element ID postNATSourceIPv4Address(225).

- TBA6 (refer to Section 3.2.4): This value is for the Type field of IP-Port-Int-IPv4-Addr TLV. It should be allocated as TLV data type and it extends the attribute tree as TBA1.TBA2.TBA3.[...TBA6...]. Also, this value uniquely refers to IPFIX Element ID sourceIPv4Address(8).

- TBA7 (refer to Section 3.2.5): This value is for the Type field of IP-Port-Int-IPv6-Addr TLV. It should be allocated as TLV data type and it extends the attribute tree as TBA1.TBA2.TBA3.[...TBA7...]. Also, this value uniquely refers to IPFIX Element ID sourceIPv6Address(27).

- TBA8 (refer to Section 3.2.6): This value is for the Type field of IP-Port-Int-Port TLV. It should be allocated as TLV data type and it extends the attribute tree as TBA1.TBA2.TBA3.[...TBA8...].

Also, this value uniquely refers to IPFIX Element ID sourceTransportPort(7).

- TBA9 (refer to Section 3.2.7): This value is for the Type field of IP-Port-Ext-port TLV. It should be allocated as TLV data type and it extends the attribute tree as TBA1.TBA2.TBA3.[...TBA9...]. Also, this value uniquely refers to IPFIX Element ID postNAPTSourceTransportPort(227).

- TBA10 (refer to Section 3.2.8): This value is for the Type field of IP-Port-Alloc TLV. It should be allocated as TLV data type and it extends the attribute tree as TBA1.TBA2.TBA3.[...TBA10...]. Also, this value uniquely refers to IPFIX Element ID natEvent(230).

- TBA11 (refer to Section 3.2.9): This value is for the Type field of IP-Port-Range-Start TLV. It should be allocated as TLV data type and it extends the attribute tree as TBA1.TBA2.TBA3.[...TBA11...]. Also, this value uniquely refers to IPFIX Element ID portRangeStart(361).

- TBA12 (refer to Section 3.2.10): This value is for the Type field of IP-Port-Range-End TLV. It should be allocated as TLV data type and it extends the attribute tree as TBA1.TBA2.TBA3.[...TBA12...]. Also, this value uniquely refers to IPFIX Element ID portRangeEnd(362).

- TBA13 (refer to Section 3.2.11): This value is for the Type field of IP-Port-Local-Id TLV. It should be allocated as TLV data type and it extends the attribute tree as TBA1.TBA2.TBA3.[...TBA13...]. Also, this value uniquely refers to IPFIX Element ID localID(TBAx3).

8. Acknowledgements

Many thanks to Dan Wing, Roberta Maglione, Daniel Derksen, David Thaler, Alan Dekok, Lionel Morand, and Peter Deacon for their useful comments and suggestions.

9. References

9.1. Normative References

[IPFIX] IANA, "IP Flow Information Export (IPFIX) Entities", <http://www.iana.org/assignments/ipfix/ipfix.xhtml>.

[RFC1918] Rekhter, Y., Moskowitz, R., Karrenberg, D., Groot, G., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, February 1996.

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

[RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629, June 1999.

[RFC2865] Rigney, C., Willens, S., Rubens, A., and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, June 2000.

[RFC5176] Chiba, M., Dommety, G., Eklund, M., Mitton, D., and B. Aboba, "Dynamic Authorization Extensions to Remote Authentication Dial In User Service (RADIUS)", RFC 5176, January 2008.

[RFC6929] DeKok, A. and A. Lior, "Remote Authentication Dial In User Service (RADIUS) Protocol Extensions", RFC 6929, April 2013.

[RFC7012] Claise, B. and B. Trammell, "Information Model for IP Flow Information Export (IPFIX)", RFC 7012, September 2013.

[TR-146] Broadband Forum, "TR-146: Subscriber Sessions", <http://www.broadband-forum.org/technical/download/TR-146.pdf>.

9.2. Informative References

[I-D.gundavelli-v6ops-community-wifi-svcs]
Gundavelli, S., Grayson, M., Seite, P., and Y. Lee, "Service Provider Wi-Fi Services Over Residential Architectures", draft-gundavelli-v6ops-community-wifi-svcs-06 (work in progress), April 2013.

[I-D.ietf-softwire-lw4over6]
Cui, Y., Qiong, Q., Boucadair, M., Tsou, T., Lee, Y., and I. Farrer, "Lightweight 4over6: An Extension to the DS-Lite Architecture", draft-ietf-softwire-lw4over6-13 (work in progress), November 2014.

[I-D.miles-behave-l2nat]
Miles, D. and M. Townsley, "Layer2-Aware NAT", draft-miles-behave-l2nat-00 (work in progress), March 2009.

[RFC3022] Srisuresh, P. and K. Egevang, "Traditional IP Network Address Translator (Traditional NAT)", RFC 3022, January 2001.

[RFC6146] Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", RFC 6146, April 2011.

[RFC6269] Ford, M., Boucadair, M., Durand, A., Levis, P., and P. Roberts, "Issues with IP Address Sharing", RFC 6269, June 2011.

[RFC6333] Durand, A., Droms, R., Woodyatt, J., and Y. Lee, "Dual-Stack Lite Broadband Deployments Following IPv4 Exhaustion", RFC 6333, August 2011.

[RFC6619] Arkko, J., Eggert, L., and M. Townsley, "Scalable Operation of Address Translators with Per-Interface Bindings", RFC 6619, June 2012.

[RFC6887] Wing, D., Cheshire, S., Boucadair, M., Penno, R., and P. Selkirk, "Port Control Protocol (PCP)", RFC 6887, April 2013.

[RFC6888] Perreault, S., Yamagata, I., Miyakawa, S., Nakagawa, A., and H. Ashida, "Common Requirements for Carrier-Grade NATs (CGNs)", BCP 127, RFC 6888, April 2013.

[RFC6967] Boucadair, M., Touch, J., Levis, P., and R. Penno, "Analysis of Potential Solutions for Revealing a Host Identifier (HOST_ID) in Shared Address Deployments", RFC 6967, June 2013.

Authors' Addresses

Dean Cheng
Huawei
2330 Central Expressway
Santa Clara, California 95050
USA

Email: dean.cheng@huawei.com

Jouni Korhonen
Broadcom
Porkkalankatu 24
FIN-00180 Helsinki
Finland

Email: jouni.nospam@gmail.com

Mohamed Boucadair
France Telecom
Rennes
France

Email: mohamed.boucadair@orange.com

Senthil Sivakumar
Cisco Systems
7100-8 Kit Creek Road
Research Triangle Park, North Carolina
USA

Email: ssenthil@cisco.com

RADIUS Extensions Working Group
Internet-Draft
Intended status: Best Current Practice
Expires: April 30, 2015

S. Winter
RESTENA

October 27, 2014

Considerations regarding the correct use of EAP-Response/Identity draft-winter-radext-populating-eapidentity-01

Abstract

There are some subtle considerations for an EAP peer regarding the content of the EAP-Response/Identity packet when authenticating with EAP to an EAP server. This document describes two such considerations and suggests workarounds to the associated problems.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 30, 2015.

Table of Contents

1. Introduction

1.1. Problem Statement

An Extensible Authentication Protocol (EAP, [RFC3748]) conversation between an EAP peer and an EAP server starts with an (optional) request for identity information by the EAP server (EAP-Request/ Identity) followed by the peer's response with identity information (EAP-Response/Identity). Only after this identity exchange are EAP types negotiated.

EAP-Response/Identity is sent before EAP type negotiation takes place, but it is not independent of the later-negotiated EAP type. Two entanglements between EAP-Response/Identity and EAP methods' notions of a user identifier are described in this document.

1. The choice of identity to send in EAP-Response/Identity may have detrimental effects on the subsequent EAP type negotiation.

2. Using identity information from the preferred EAP type without thoughtful conversion of character encoding may have detrimental effects on the outcome of the authentication.

The following two chapters describe each of these issues in detail. The last chapter contains recommendations for implementers of EAP peers to avoid these issues.

1.2. Requirements Language

In this document, several words are used to signify the requirements of the specification. 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 RFC 2119. [RFC2119]

2. EAP-Response/Identity: Effects on EAP type negotiation

Assuming the EAP peer's EAP type selection is not the trivial case (i.e. it has more than one configured EAP type for a given network or application, and needs to make a decision which one to use), an issue arises when the configured EAP types are not all configured with the same user identifier.

Issue: if the user identifiers in the set of configured EAP types differ (e.g. have a different [RFC4282] "realm" portion), and the authenticator does not send identity selection hints as per [RFC4284], then EAP type negotiation may be limited to those EAP types which are terminated in the same EAP server. The reason for that is because the information in the EAP-Response/Identity is used for request routing decisions and thus determines the EAP server - a given user identifier may be routed to a server which exclusively serves the matching EAP type. Negotiating another EAP type from the set of configured EAP types during the running EAP conversation is then not possible.

Example:

Assume an EAP peer is configured to support two EAP types:

- EAP-AKA' [RFC5448] with user identifier imsi@mnc123.mcc123.3gpp-network.org
- EAP-TTLS [RFC5281] with user identifier john@realm.example

The user connects to hotspot of a roaming consortium which could authenticate him with EAP-TTLS and his john@realm.example identity. The hotspot operator has no business relationship at all with the 3GPP consortium; incoming authentication requests for realms ending in 3gppnetwork.org will be immediately rejected. Identity selection hints are not sent.

Consequence: If the EAP peer consistently chooses the imsi@mnc123.mcc123.3gpp-network.org user identifier as choice for its initial EAP-Response/Identity, the user will be consistently and perpetually rejected, even though in possession of a valid credential for the hotspot.

An EAP peer should always try all options to authenticate. As the example above shows, it may not be sufficient to rely on EAP method negotiation alone to iterate through all configured EAP types and

come to a conclusive outcome of the authentication attempt. Multiple new EAP authentications, each using a different user identifier from the set of configured user identities, may be required to fully iterate through the list of usable identities.

3. Character (re-)encoding may be required

The user identifier as configured in the EAP method configuration is not always suited as user identifier to choose as EAP-Response/ Identity. This is trivially true when using tunneled EAP types and configuring anonymous outer identity for the tunneling EAP type. There is at least one additional, non-trivial, case to consider however:

EAP methods define the encoding of their user identifiers; in particular, the encoding of the user identifiers as defined the EAP method may or may not be UTF-8; some EAP methods are even known not to put any encoding restrictions on their user identifiers at all.

It is not the intention of EAP, as a mere method-agnostic container which simply carries EAP types, to restrict an EAP method's choice of encoding of a user identifier. However, there are restrictions in what should be contained in the EAP-Response/Identity: EAP is very often carried over a AAA protocol (e.g over RADIUS as per [RFC3579]). The typical use for the contents of EAP-Response/Identity inside AAA protocols like RADIUS [RFC2865] and Diameter [RFC6733] is to copy the content of EAP-Response/Identity into a "User-Name" attribute; the encoding of the User-Name attribute is required to be UTF-8. EAP-Response/Identity does not carry encoding information itself, so a conversion between a non-UTF-8 encoding and UTF-8 is not possible for the AAA entity doing the EAP-Response/Identity to User-Name copying.

Consequence: If an EAP method's user identifier is not encoded in UTF-8, and the EAP peer verbatimly uses that method's notion of a user identifier for its EAP-Response/Identity field, then the AAA entity is forced to violate its own specification because it has to, but can not use UTF-8 for its own User-Name attribute. If the EAP method configuration sets an outer identity in a non UTF-8 character set, and the EAP peer verbatimly uses that outer identity for its EAP-Response/Identity field, then the same violation occurs.

This jeopardizes the subsequent EAP authentication as a whole; request routing may fail, lead to a wrong destination or introduce routing loops due to differing interpretations of the User-Name in EAP pass-through authenticators and AAA proxies.

4. Recommendations for EAP peer implementations

Where user identifiers between configured EAP types in an EAP peer differ, the EAP peer can not rely on the EAP type negotiation mechanism alone to provide useful results. If an EAP authentication gets rejected, the EAP peer SHOULD re-try the authentication using a different EAP-Response/Identity than before. The EAP peer SHOULD try all user identifiers from the entire set of configured EAP types before declaring final authentication failure.

EAP peers need to maintain state on the encoding of the user identifiers which are used in their locally configured EAP types. When constructing an EAP-Response/Identity from that user identifier, they MUST (re-)encode that user identifier as UTF-8 and use the resulting value for the EAP-Response/Identity. If the EAP type is configured for the use of anonymous outer identities, the desired outer identity MUST also be (re-)encoded in UTF-8 encoding before being put into the EAP-Response/Identity.

5. Privacy Considerations

Because the EAP-Response/Identity content is not encrypted, the backtracking to a new EAP-Response/Identity will systematically reveal all configured identities to intermediate passive listeners on the path between the EAP peer and the EAP server (until one authentication round succeeds).

This additional leakage of identity information is not very significant though because where privacy is considered important, the additional option for identity privacy which is present in most modern EAP methods can be used.

If the EAP peer implementation is certain that all EAP types will be terminated at the same EAP server (e.g. with a corresponding configuration option) then the iteration over all identities can be avoided, because the EAP type negotiation is then sufficient.

If a choice of which identity information to disclose needs to be made by the EAP peer, when iterating through the list of identities the EAP peer SHOULD

in first priority honour a manually configured order of preference of EAP types, if any

in second priority try EAP types in order of less leakage first; that is, EAP types with a configured outer identity should be tried before other EAP types which would reveal actual user identities.

6. Security Considerations

The security of an EAP conversation is determined by the EAP method which is used to authenticate. This document does not change the actual authentication with an EAP method, and all the security properties of the chosen EAP method remain. The format requirements (character encoding) and operational considerations (re-try EAP with a different EAP-Response/Identity) do not lead to new or different security properties.

7. IANA Considerations

There are no IANA actions in this document.

8. References

8.1. Normative References

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

8.2. Informative References

[RFC2865] Rigney, C., Willens, S., Rubens, A., and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, June 2000.

[RFC3579] Aboba, B. and P. Calhoun, "RADIUS (Remote Authentication Dial In User Service) Support For Extensible Authentication Protocol (EAP)", RFC 3579, September 2003.

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, "Extensible Authentication Protocol (EAP)", RFC 3748, June 2004.

[RFC4282] Aboba, B., Beadles, M., Arkko, J., and P. Eronen, "The Network Access Identifier", RFC 4282, December 2005.

[RFC4284] Adrangi, F., Lortz, V., Bari, F., and P. Eronen, "Identity Selection Hints for the Extensible Authentication Protocol (EAP)", RFC 4284, January 2006.

[RFC5281] Funk, P. and S. Blake-Wilson, "Extensible Authentication Protocol Tunneled Transport Layer Security Authenticated Protocol Version 0 (EAP-TTLSv0)", RFC 5281, August 2008.

[RFC5448] Arkko, J., Lehtovirta, V., and P. Eronen, "Improved Extensible Authentication Protocol Method for 3rd Generation Authentication and Key Agreement (EAP-AKA')", RFC 5448, May 2009.

[RFC6733] Fajardo, V., Arkko, J., Loughney, J., and G. Zorn, "Diameter Base Protocol", RFC 6733, October 2012.

Author's Address

Stefan Winter
Fondation RESTENA
6, rue Richard Coudenhove-Kalergi
Luxembourg 1359
LUXEMBOURG

Phone: +352 424409 1
Fax: +352 422473
EMail: stefan.winter@restena.lu
URI: http://www.restena.lu.