RADIUS EXTensions Working Group
Internet-Draft
Intended status: Experimental
Expires: August 4, 2012

A. Perez-Mendez
R. Marin-Lopez
F. Pereniguez-Garcia
G. Lopez-Millan
University of Murcia
D. Lopez
Telefonica I+D
A. DeKok
Network RADIUS
Feb 2012

Support of fragmentation of RADIUS packets
draft-perez-radext-radius-fragmentation-01

Abstract

This document describes a mechanism providing fragmentation support of RADIUS packets that exceed the 4 KB limit.

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 August 4, 2012.

Table of Contents

1. Introduction

RADIUS [RFC2865] is a protocol for carrying authentication, authorization, and configuration information between a Network Access Server (NAS) which desires to authenticate its links and a shared Authentication Server (AS). Information is exchanged between the NAS and the AS through packets. Each RADIUS packet can transport several RADIUS attributes, to convey the necessary information to the other peer, up to a maximum size of 4 KB of total data (including RADIUS packet headers). RADIUS attributes have a maximum size of 253 bytes of payload.

RADIUS has been extensively used along the years. Along this time, the need of sending RADIUS attributes larger than 253 bytes has become a reality. An immediate alternative to overcome this issue consists in splitting the data into a group of RADIUS attributes of the same type, and then insert them into the RADIUS packet in order. At the destination, the content of these attributes is extracted and joined to rebuild the original data. This scheme is followed, for example, by RADIUS-EAP [RFC3579]. A more advanced solution is given in [I-D.ietf-radext-radius-extensions], where extended attributes can be marked with a flag to indicate fragmentation. A reference-based mechanism is also proposed in [RFC6158], where attribute can be obtained through an out-of-band protocol.

However, there are no proposals to deal with fragmentation at a packet level, when the total length exceeds the 4 KB limit imposed by the RADIUS specification. As the usage of RADIUS is being considered in more complex AAA scenarios, including the exchange of richer data, like SAML assertions or JWT tokens, exceeding this limit becomes more likely, thus making necessary the availability of mechanisms for dealing with this situation.

This document defines a mechanism to allow RADIUS peers to exchange packets that exceed the 4 KB limit, by fragmenting them across several exchanges. This proposal tries to maintain compatibility with any intra-packet fragmentation mechanism and with the existing RADIUS deployments.

1.1. 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].

2. Overview

When a RADIUS peer needs to send a packet that exceeds the 4 KB, the following mechanism is used. First, the large packet is split into several smaller RADIUS packets (i.e. chunks) of the same type (e.g. Access-Request). The first chunk contains the first "n" RADIUS attributes of the original packet (in the same order), until a limit of 4092 bytes (4096 - 4) is reached. It can be less, depending on the specific length of the attributes, but never more. If there are still attributes from the original packet that have not been yet included into any chunk, a new attribute called More-Data-Pending is appended into the chunk. This process is repeated until all the RADIUS attributes from the original packet have been included into some chunk.

Then the first chunk is sent to the peer, which identifies the packet as a chunk (the More-Data-Pending attribute is present), and requests for the rest of the chunks. Once all the chunks have been received by the peer, the original packet is reconstructed and processed as if it had been received in one piece.

When a packet is truncated into chunks, a special situation may occur in combination with Extended Type attributes as defined in [I-D.ietf-radext-radius-extensions]. If the truncation occurs in the middle of a fragmented attribute, the last attribute of the chunk will be an Extended Type with Flags, with flag M enabled. This situation is specifically forbidden in [I-D.ietf-radext-radius-extensions]. To indicate that this situation is provoked by a truncation and hence MUST be allowed, a new flag "T" (indicating truncation) MUST be set into that Extended-Type-Flag attribute. The combination of the flags "M" and "T" indicates that the attribute is fragmented (flag M), but that all the fragments are not available in this chunk (flag T).

Indeed, this situation will be the most usual. When packet fragmentation is required, usually it will be motivated by the inclusion of one or more large attribute that makes use of attribute fragmentation. Hence, the truncation will probably split the large attribute into two (or more) pieces. The rest of possibilities, where the truncation point does not split a fragmented attribute, does not require any special treatment.

3. More-Data-Pending attribute

This document proposes the definition of a new extended type attribute, called More-Data-Pending. The format of this attribute follows the indications of an Extended Type attribute defined in

[I-D.ietf-radext-radius-extensions]. The presence of this attribute indicates that the received RADIUS packet is not complete (i.e. it is a chunk), and more data MUST be received to regenerate the original packet. The following figure represents the format of the More-Data-Pending attribute.

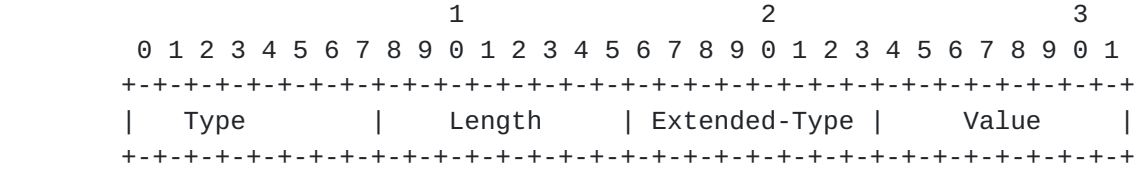

Figure 1: More-Data-Pending format

Type

To be assigned (TBA)

Length

4

Extended-Type

To be assigned (TBA).

Value

1 byte. Not defined yet.

This attribute MAY be present in Access-Request and Acess-Challenge packets. It MUST not be included in Acess-Accept packets.

4. Fragmentation of Access-Request packets

When the NAS desires to send a RADIUS packet that exceeds the 4 KB limit, the packet can be split into smaller packets (chunks) and sent over different exchanges. This fact is indicated by including a More-Data-Pending attribute on each chunk (except the last one of the series). The process is described in detail using the following example. In this example, the attributes "Data" and "Other" are Extended Type with Flags, as defined in [I-D.ietf-radext-radius-extensions].

In order to make the example simpler, it is assumed that each RADIUS packet can include up to 8 RADIUS attributes, instead of using bytes. Flag M is indicated as [M]. Flag T is indicated as [T]. Presence of

both is indicated as [MT]. Data1, Data2, Data3... indicate successive fragments of the attribute "Data".

- The RADIUS client wants to send the following RADIUS packet:

  Access-Request = User-Name, Calling-Station-Id, Data1[M], Data2[M], Data3[M], Data4[M], Data5[M], Data6[M], Data7[M], Data8[M], Data9[M], Data10, Other1[M], Other2[M], Other3

- As the RADIUS packet exceeds the maximum allowed length (8 attributes), the RADIUS client truncates the packet to generate the first chunk, including the More-Data-Pending attribute. Flag "T" is activated into the fragment "Data5", as it is the last of the packet (chunk), but not the last of the fragmented attribute.

  Access-Request-1 = User-Name, Calling-Station-Id, Data1[M], Data2[M], Data3[M], Data4[M], Data5[MT], More-Data-Pending

- When the server receives the RADIUS packet containing the More-Data-Pending attribute, the processing of the packet is delayed until all the pending data is received. The pending data is requested by means of an Access-Challenge packet, using the State attribute to tie together this response with the subsequent request from the client.

  Access-Challenge-1 = State1

- The client continues including attributes until another RADIUS packet (i.e. chunk) is completed, appending again the More-Data-Pending attribute. The State attribute received in the Access-Challenge is also included in this chunk. Again, flag "T" is enabled in the last fragment of the chunk to indicate that a truncation took place.

  Acess-Request-2 = State1, Data6[M], Data7[M], Data8[M], Data9[M], Data10, Other1[MT], More-Data-Pending

- As the received request contains the More-Data-Pending, the server stores the attributes into the state associated to State1 and replies with another Access-Challenge. The challenge contains a new State attribute that refers to this conversation.

  Access-Challenge-2 = State2

- Finally, the client sends the last chunk of the original packet, including the received State attribute.

Access-Request-3 = State2, Other2[M], Other3

o On reception of this last chunk (no More-Data-Pending attribute present), the server can process the totality of the received attributes as if they all had been received into a single RADIUS packet larger than 4 KB.

The following figure depicts the exchange of chucks between the NAS and the AS.

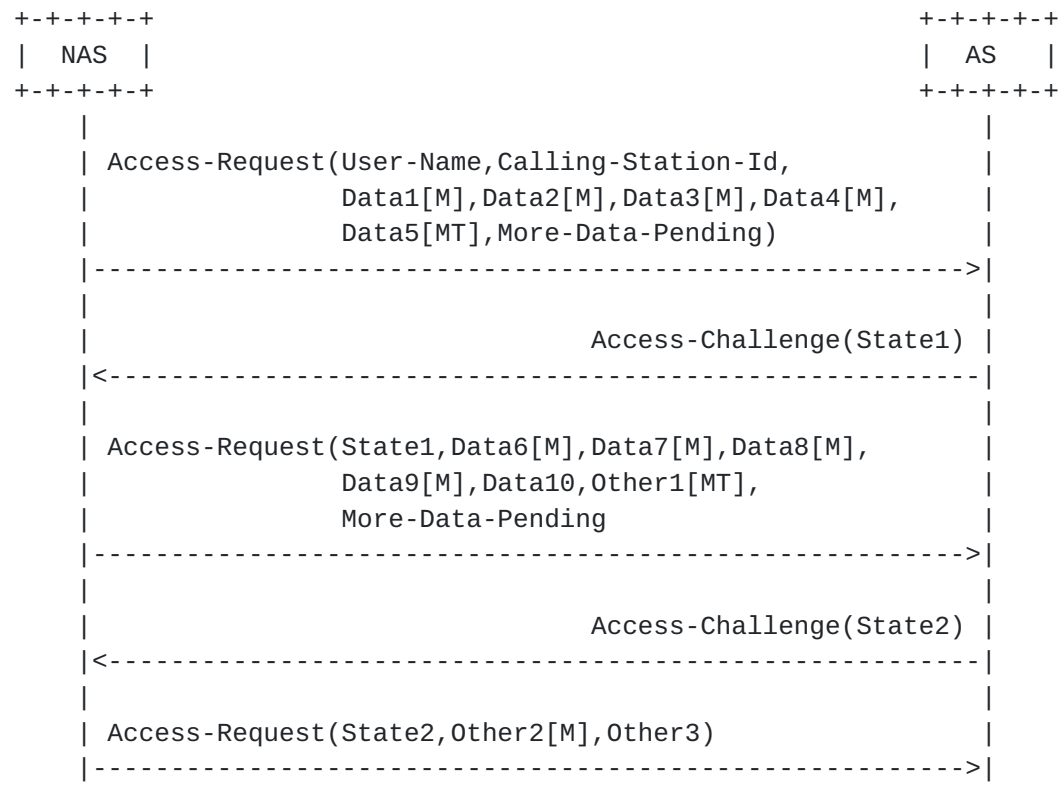

Figure 2: Fragmented Access-Request packet

5. Fragmentation of Access-Challenge packets

When is the server (AS) the one who wants to send a large RADIUS packet, the solution is very similar to the previous one. The one difference is that in this scenario, the AS includes a State attribute along with the More-Data-Required.

The following figure depicts how the message exchange would be if the AS wanted to send a large packet to the NAS. Specifically, it wants to send the following challenge:

Access-Challenge = Data1[M], Data2[M], Data3[M], Data4[M],
Data5[M], Data6[M], Data7[M], Data8[M], Data9[M], Data10,
Other1[M], Other2[M], Other3

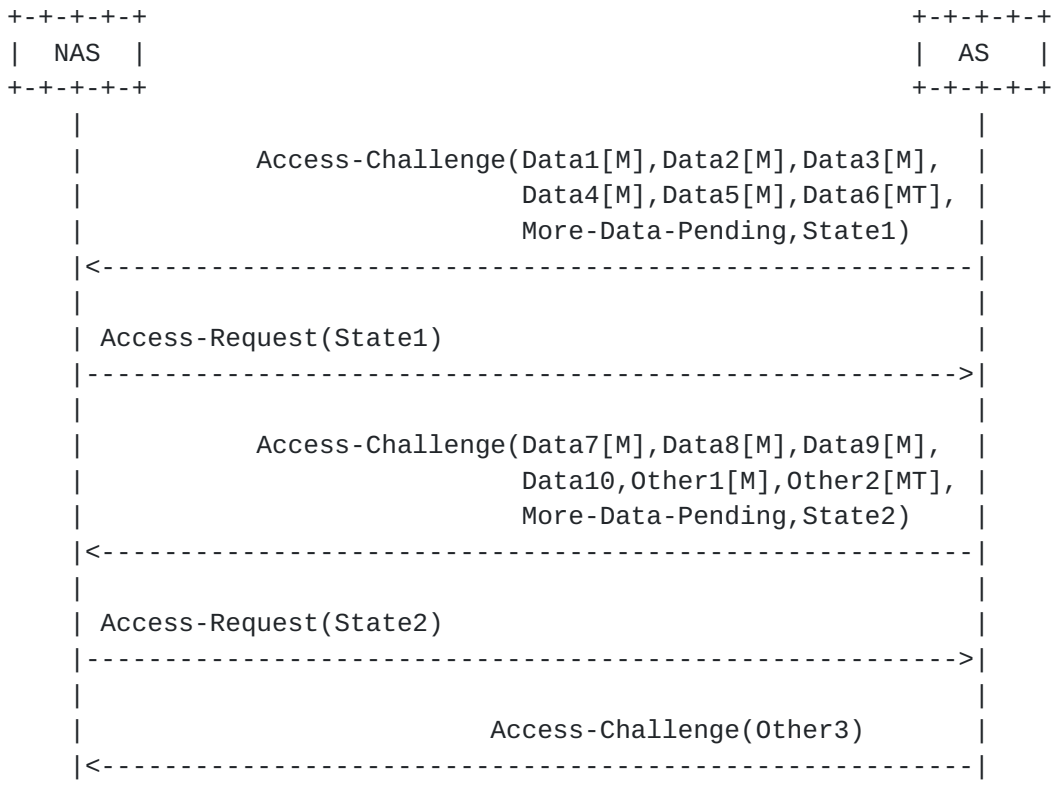

Figure 3: Fragmented Access-Challenge packet

6. Fragmentation of Access-Accept packets

If the AS wants to send an Access-Accept packet that exceeds the 4 KB limit, the operation is slightly different. As some attributes are allowed to appear in Access-Accept packets, but not in Access-Challenge packets, the solution described in the previous sections is not directly applicable. Instead, the AS MUST send an Access-Accept packet to the NAS containing all the attributes that can not be included in Access-Challenge packets, and indicating "Authorize-Only" as the Service-Type. The additional attributes are received via a series of Access-Request/Access-Challenge exchanges, as described in the previous section. Finally, the last packet from the AS to the NAS would be an Access-Accept containing the real Service-Type for the user. For simpliciy, Service-Type[X] indicates a Service-Type attribute of value X.

- The AS wants to send the following Access-Accept packet:

  Access-Accept = User-Name, Service-Type[X], Framed-IP-Address, Data1[M], Data2[M], Data3[M], Data4[M], Data5[M], Data6[M], Data7[M], Data8[M], Data9[M], Data10

- As the RADIUS packet exceeds the maximum allowed length (8 attributes), the AS truncates the packet to generate the first chunk. This chunk only includes the attributes that cannot be included in Access-Challenge packets. In this example they are the User-Name, the Service-Type and the Framed-IP-Address. The Service-Type is changed to "Authorize-Only", and a State attribute is included.

  Access-Accept-1 = User-Name, Service-Type[Authorize-Only], Framed-IP-Addres, State1

- When the NAS receives the Access-Accept, it determines, based on the Service-Type=Authorize-Only, that an additional exchange is required. Thus, it generates a new Access-Request packet containing the received State attribute.

  Access-Request-1 = State1

- The AS then generates a new chunk with part of the remaining attributes to be sent. As they do not fit into a single chunk, a More-Data-Pending attribute and a new State attribute are also included.

  Access-Challenge-1 = Data1[M], Data2[M], Data3[M], Data4[M], Data5[M], Data6[MT], More-Data-Pending, State2

- The NAS determines the received packet is part of a larger one (i.e. it is a chunk) due to the presence of the More-Data-Pending attribute, hence it requests the rest of the data by sending a new Access-Request packet including the received State attribute.

  Access-Request-2 = State2

- Finally, the AS includes the rest of the attributes into the final Access-Accept packet. This packet also includes the original Service-Type for the user.

  Access-Accept-2 = Data7[M], Data8[M], Data9[M], Data10, Service-Type[X]

o On reception of this last packet, the NAS can process the totality of the received attributes as if they all had been received into a single RADIUS packet larger than 4 KB.

The following figure depicts the exchange of chucks between the NAS and the AS.

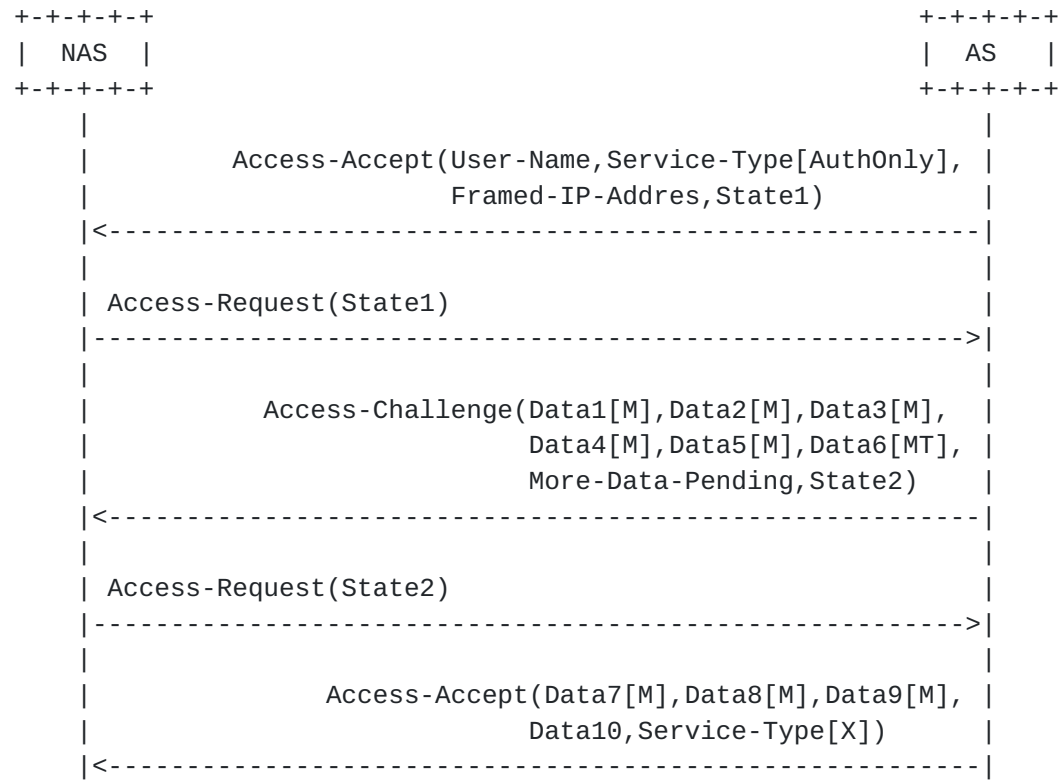

Figure 4: Fragmented Access-Accept packet

7. Security Considerations

8. IANA Considerations

This document has no actions for IANA.

9. Normative References

[I-D.ietf-radext-radius-extensions]
DeKok, A. and A. Lior, "Remote Authentication Dial In User Service (RADIUS) Protocol Extensions", draft-ietf-radext-radius-extensions-04 (work in progress), January 2012.

[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.

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

[RFC6158] DeKok, A. and G. Weber, "RADIUS Design Guidelines", BCP 158, RFC 6158, March 2011.

Authors' Addresses

Alejandro Perez-Mendez (Ed.)
University of Murcia
Campus de Espinardo S/N, Faculty of Computer Science
Murcia, 30100
Spain

Phone: +34 868 88 46 44
Email: alex@um.es

Rafa Marin-Lopez
University of Murcia
Campus de Espinardo S/N, Faculty of Computer Science
Murcia, 30100
Spain

Phone: +34 868 88 85 01
Email: rafa@um.es

Fernando Pereniguez-Garcia
University of Murcia
Campus de Espinardo S/N, Faculty of Computer Science
Murcia, 30100
Spain

Phone: +34 868 88 78 82
Email: pereniguez@um.es

Gabriel Lopez-Millan
University of Murcia
Campus de Espinardo S/N, Faculty of Computer Science
Murcia, 30100
Spain

Phone: +34 868 88 85 04
Email: gabilm@um.es

Diego R. Lopez
Telefonica I+D
Don Ramon de la Cruz, 84
Madrid, 28006
Spain

Phone: +34 913 129 041
Email: diego@tid.es

Alan DeKok
Network RADIUS
15 av du Granier
Meylan, 38240
France

Phone: +34 913 129 041
Email: aland@networkradius.com
URI: http://networkradius.com