Network Working Group
INTERNET-DRAFT
Category: Best Current Practice
<draft-ietf-radext-design-09.txt>
Expires: April 12, 2010
12 October 2009

Alan DeKok (ed.)
FreeRADIUS
G. Weber
Individual Contributor

RADIUS Design Guidelines
draft-ietf-radext-design-09

Copyright Notice

Abstract

This document provides guidelines for the design of attributes used by the Remote Authentication Dial In User Service (RADIUS) protocol. It is expected that these guidelines will prove useful to authors and reviewers of future RADIUS attribute specifications, both within the IETF as well as other Standards Development Organizations (SDOs).

Table of Contents

1. Introduction

This document provides guidelines for the design of RADIUS attributes both within the IETF as well as within other SDOs. By articulating RADIUS design guidelines, it is hoped that this document will encourage the development and publication of high quality RADIUS attribute specifications.

However, the advice in this document will not be helpful unless it is put to use. As with "Guidelines for Authors and Reviewers of MIB Documents" [RFC4181], it is expected that this document will be used by authors to check their document against the guidelines prior to requesting review (such as an "Expert Review" described in [RFC3575]). Similarly, it is expected that this document will used by reviewers (such as WG participants or the AAA Doctors [DOCTORS]), resulting in an improvement in the consistency of reviews.

In order to meet these objectives, this document needs to cover not only the science of attribute design, but also the art. As a result, in addition to covering the most frequently encountered issues, this document attempts to provide some of the considerations motivating the guidelines.

In order to characterize current attribute usage, both the basic and complex data types defined in the existing RADIUS RFCs are reviewed.

1.1. Terminology

This document uses the following terms:

Network Access Server (NAS)
: A device that provides an access service for a user to a network.

RADIUS server
: A RADIUS authentication, authorization, and/or accounting (AAA) server is an entity that provides one or more AAA services to a NAS.

RADIUS proxy
: A RADIUS proxy acts as a RADIUS server to the NAS, and a RADIUS client to the RADIUS server.

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].

1.3. Applicability

As RADIUS has become more widely accepted as a management protocol, its usage has become more prevalent, both within the IETF as well as within other SDOs. Given the expanded utilization of RADIUS, it has become apparent that requiring SDOs to accomplish all their RADIUS work within the IETF is inherently inefficient and unscalable. By articulating guidelines for RADIUS attribute design, this document enables SDOs out of the IETF to design their own RADIUS attributes within the Vendor-Specific Attribute (VSA) space.

It is RECOMMENDED that SDOs follow the guidelines articulated in this document. Doing so will ensure the widest possible applicability and interoperability of the specifications, while requiring minimal changes to existing systems. Specifications that do not follow the guidelines articulated herein are NOT RECOMMENDED. However, we recognize that there are some situations where SDOs or vendors require the creation of specifications not following these guidelines. We do not forbid these specifications, but it is RECOMMENDED that they are created only if they have a limited scope of applicability, and all attributes defined in those specifications are VSAs, as discussed Appendix A.5, below.

It is RECOMMENDED that SDOs and vendors seek review of RADIUS attribute specifications from the IETF. However, when specifications are SDO specific, re-use existing data types, and follow these guidelines, they do not require IETF review.

In order to enable IETF review of such specifications, the authors recommend that:

* SDOs make their RADIUS attribute specifications publicly available;

* SDOs request review of RADIUS attribute specifications by sending email to the AAA Doctors [DOCTORS] or equivalent mailing list;

* IETF and SDO RADIUS attribute specifications are reviewed according to the guidelines proposed in this document;

* Reviews of specifications are posted to the RADEXT WG mailing list, the AAA Doctors mailing list [DOCTORS] or another IETF mailing list suggested by the Operations & Management Area Directors of the IETF.

These reviews can assist with creation of specifications that meet the SDO requirements, and which are also compatible with the

traditional data model and uses of RADIUS. While these reviews require access to public specifications, the review process does not require publication of an IETF RFC.

The advice in this document applies to attributes used to encode service-provisioning or authentication data. RADIUS protocol changes, or specification of attributes (such as Service-Type) that can be used to, in effect, provide new RADIUS commands require greater expertise and deeper review, as do changes to the RADIUS operational model as discussed below in Section 3.3. Such changes MUST NOT be undertaken outside the IETF and when handled within the IETF require "IETF Consensus" for adoption, as noted in [RFC3575] Section 2.1.

2. RADIUS Data Model

The Remote Authentication Dial In User Service (RADIUS) defined in [RFC2865] and [RFC2866] uses elements known as attributes in order to represent authentication, authorization and accounting data.

Unlike SNMP, first defined in [RFC1157] and [RFC1155], RADIUS does not define a formal data definition language. A handful of basic data types are in common use, and a data type is associated with an attribute when the attribute is defined.

Two distinct attribute spaces are defined: the standard space, and a Vendor-Specific space. Attributes in the standard space generally are composed of a type, length, value (TLV) triplet, although complex attributes have also been defined. The Vendor-Specific space is encapsulated within a single attribute type (Vendor-Specific Attribute). The format of this space is defined by individual vendors, but the same TLV encoding used by the standard space is recommended in [RFC2865] Section 5.26. The similarity between attribute formats has enabled implementations to leverage common parsing functionality, although in some cases the attributes in the Vendor-Specific space have begun to diverge from the common format.

2.1. Standard Space

The following subsections describe common data types and formats within the RADIUS standard attribute space. Common exceptions are identified.

2.1.1. Basic Data Types

The data type of RADIUS attributes is not transported on the wire. Rather, the data type of a RADIUS attribute is fixed when that attribute is defined. Based on the RADIUS attribute type code,

RADIUS clients and servers can determine the data type based on pre-configured entries within a data dictionary.

[RFC2865] defines the following data types:

text 1-253 octets containing UTF-8 encoded 10646 [RFC3629] characters. Text of length zero (0) MUST NOT be sent; omit the entire attribute instead.

string	1-253 octets containing binary data (values 0 through 255 decimal, inclusive). Strings of length zero (0) MUST NOT be sent; omit the entire attribute instead.
IPv4 address	32 bit value, in network byte order.
integer	32 bit unsigned value, in network byte order.
time	32 bit unsigned value, in network byte order. -- seconds since 00:00:00 UTC, January 1, 1970.

In addition to these data types, follow-on RADIUS specifications define attributes using the following additional types:

IPv6 address	128 bit value, in network byte order.
IPv6 prefix	8 bits of reserved, 8 bits of prefix length, up to 128 bits of value, in network byte order.
integer64	64 bit unsigned value, in network byte order This type has also been used to represent an IPv6 interface identifier.

Examples of the IPv6 address type include NAS-IPv6-Address defined in [RFC3162] Section 2.1 and Login-IPv6-Host defined in [RFC3162] Section 2.4. The IPv6 prefix type is used in [RFC3162] Section 2.3, and in [RFC4818] Section 3. The integer64 type is used for the ARAP-Challenge-Response Attribute defined in [RFC2869] Section 5.15, and the Framed-Interface-Id Attribute defined in [RFC3162] Section 2.2. [RFC4675] Section 2.4 defines User-Priority-Table as 64-bits in length, but denotes it as type String.

Given that attributes of type IPv6 address, IPv6 prefix, and integer64 are already in use, it is RECOMMENDED that RADIUS server implementations include support for these additional basic types, in addition to the types defined in [RFC2865].

Where the intent is to represent a specific IPv6 address, the IPv6 address type SHOULD be used. Although it is possible to use the IPv6 IPv6 Prefix type with a prefix length of 128 to represent an IPv6 address, this usage is NOT RECOMMENDED.

It is worth noting that since RADIUS only supports unsigned integers of 32 or 64 bits, attributes using signed integer data types or unsigned integer types of other sizes will require code changes, and

SHOULD be avoided.

For [RFC2865] RADIUS VSAs, the length limitation of the String and Text types is 247 octets instead of 253 octets, due to the additional overhead of the Vendor-Specific Attribute.

2.1.2. Tagging Mechanism

[RFC2868] defines an attribute grouping mechanism based on the use of a one octet tag value. Tunnel attributes that refer to the same tunnel are grouped together by virtue of using the same tag value.

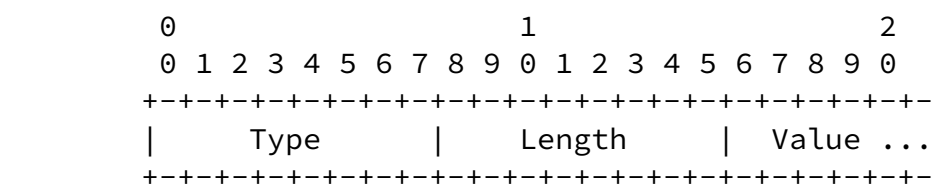
This tagging mechanism has some drawbacks. There are a limited number of unique tags (31). The tags are not well suited for use with arbitrary binary data values, because it is not always possible to tell if the first byte after the Length is the tag or the first byte of the untagged value (assuming the tag is optional).

Other limitations of the tagging mechanism are that when integer values are tagged, the value portion is reduced to three bytes meaning only 24-bit numbers can be represented. The tagging mechanism does not offer an ability to create nested groups of attributes. Some RADIUS implementations treat tagged attributes as having additional data types tagged-string and tagged-integer. These types increase the complexity of implementing and managing RADIUS systems.

For these reasons, the tagging scheme described in RFC 2868 is NOT RECOMMENDED for use as a generic grouping mechanism.

2.1.3. Complex Attribute Usage

The RADIUS attribute encoding is summarized in [RFC2865]:



However, some standard attributes do not follow this format. Attributes that use sub-fields instead of using a basic data type are known as "complex attributes". As described below, the definition of complex attributes can lead to interoperability and deployment issues, so they need to be introduced with care.

In general, complex attributes sent from the RADIUS server to the client can be supported by concatenating the values into a String

data type field. However, separating these values into different attributes, each with its own type and length, would have the following benefits:

* it is easier for the user to enter the data as well-known types, rather than complex structures;

* it enables additional error checking by leveraging the parsing and validation routines for well-known types;

* it simplifies implementations by eliminating special-case attribute-specific parsing.

One of the fundamental goals of the RADIUS protocol design was to allow RADIUS servers to be configured to support new attributes without requiring server code changes. RADIUS server implementations typically provide support for basic data types, and define attributes in a data dictionary. This architecture enables a new attribute to be supported by the addition of a dictionary entry, without requiring RADIUS server code changes.

On the RADIUS client, code changes are typically required in order to implement a new attribute. The RADIUS client typically has to compose the attribute dynamically when sending. When receiving, a RADIUS client needs to be able to parse the attribute and carry out the requested service. As a result, a detailed understanding of the new attribute is required on clients, and data dictionaries are less useful on clients than on servers.

Given these considerations, the introduction of a new basic or complex attribute will typically require code changes on the RADIUS client. The magnitude of changes for the complex attribute could be greater, due to the potential need for custom parsing logic.

The RADIUS server can be configured to send a new static attribute by entering its type and data format in the RADIUS server dictionary, and then filling in the value within a policy based on the attribute name, data type and type-specific value. For complex attribute types not supported by RADIUS server dictionaries, changes to the dictionary code can be required in order to allow the new attribute to be supported by and configured on the RADIUS server.

Code changes can also be required in policy management and in the RADIUS server's receive path. These changes are due to limitations in RADIUS server policy languages, which typically only provide for limited operations (such as comparisons or arithmetic operations) on

the basic data types. Many existing RADIUS policy languages typically are not capable of parsing sub-elements, or providing

sophisticated matching functionality.

Given these limitations, the introduction of complex attributes can require code changes on the RADIUS server which would be unnecessary if basic data types had been used instead. In addition, attribute-specific parsing means more complex software to develop and maintain. More complexity can lead to more error prone implementations, interoperability problems, and even security vulnerabilities. These issues can increase costs to network administrators as well as reducing reliability and introducing deployment barriers. As a result, the introduction of new complex data types within RADIUS attribute specifications SHOULD be avoided, except in the case of complex attributes involving authentication or security functionality.

As can be seen in Appendix B, most of the existing complex attributes involve authentication or security functionality. Supporting this functionality requires code changes on both the RADIUS client and server, regardless of the attribute format. As a result, in most cases, the use of complex attributes to represent these methods is acceptable, and does not create additional interoperability or deployment issues.

The only other exception to the recommendation against complex types is for types that can be treated as opaque data by the RADIUS server. For example, the EAP-Message attribute, defined in [RFC3579] Section 3.1 contains a complex data type that is an EAP packet. Since these complex types do not need to be parsed by the RADIUS server, the issues arising from policy language limitations do not arise. Similarly, since attributes of these complex types can be configured on the server using a data type of String, dictionary limitations are also not encountered. Appendix A.1 below includes a series of checklists that may be used to analyze a design for RECOMMENDED and NOT RECOMMENDED behavior in relation to complex types.

If the RADIUS Server simply passes the contents of an attribute to some non-RADIUS portion of the network, then the data is opaque, and SHOULD be defined to be of type String. A concrete way of judging this requirement is whether or not the attribute definition in the

RADIUS document contains delineated fields for sub-parts of the data. If those fields need to be delineated in RADIUS, then the data is not opaque, and it SHOULD be separated into individual RADIUS attributes.

An examination of existing RADIUS RFCs discloses a number of complex attributes that have already been defined. Appendix B includes a listing of complex attributes used within [RFC2865], [RFC2868], [RFC2869], [RFC3162], [RFC4818], and [RFC4675]. The discussion of these attributes includes reasons why a complex type is acceptable,

or suggestions for how the attribute could have been defined to follow the RADIUS data model.

In other cases, the data in the complex type are described textually. This is possible because the data types are not sent within the attributes, but are a matter for endpoint interpretation. An implementation can define additional data types, and use these data types today by matching them to the attribute's textual description.

2.1.4. Complex Attributes and Security

The introduction of complex data types brings the potential for the introduction of new security vulnerabilities. Experience shows that the common data types have few security vulnerabilities, or else that all known issues have been found and fixed. New data types require new code, which may introduce new bugs, and therefore new attack vectors.

RADIUS servers are highly valued targets, as they control network access and interact with databases that store usernames and passwords. An extreme outcome of a vulnerability due to a new, complex type would be that an attacker is capable of taking complete control over the RADIUS server.

The use of attributes representing opaque data does not reduce this threat. The threat merely moves from the RADIUS server to the application that consumes that opaque data.

The threat is particularly severe when the opaque data originates from the user, and is not validated by the NAS. In those cases, the RADIUS server is potentially exposed to attack by malware residing on an unauthenticated host. Applications consuming opaque data that

reside on the RADIUS server SHOULD be properly isolated from the RADIUS server, and SHOULD run with minimal privileges. Any potential vulnerabilities in that application will then have minimal impact on the security of the system as a whole.

2.1.5. Service definitions and RADIUS

RADIUS specifications define how an existing service or protocol can be provisioned using RADIUS. Therefore, it is expected that a RADIUS attribute specification will reference documents defining the protocol or service to be provisioned. Within the IETF, a RADIUS attribute specification SHOULD NOT be used to define the protocol or service being provisioned. New services using RADIUS for provisioning SHOULD be defined elsewhere and referenced in the RADIUS specification.

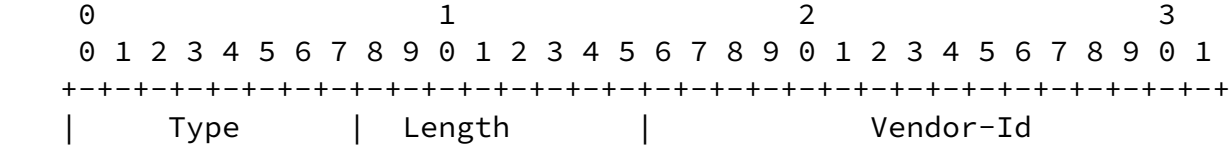
New attributes, or new values of existing attributes, SHOULD NOT be used to define new RADIUS commands. RADIUS attributes are intended to:

* authenticate users
* authorize users (i.e., service provisioning or changes to provisioning)
* account for user activity (i.e., logging of session activity)

New commands (i.e., the Code field in the packet header) are allocated only through "IETF Consensus" as noted in [RFC3575] Section 2.1. Specifications also SHOULD NOT use new attributes to modify the interpretation of existing RADIUS commands.

2.2. Vendor Space

As noted in [RFC2865] Section 5.26, the VSA format is defined as follows:



The high-order octet of the Vendor-Id field is 0 and the low-order 3 octets are the Structure of Management Information (SMI) Network Management Private Enterprise Code (PEC) of the Vendor in network byte order.

While the format of the String field is defined by the vendor, [RFC2865] Section 5.26 notes:

> It SHOULD be encoded as a sequence of vendor type / vendor length / value fields, as follows. The Attribute-Specific field is dependent on the vendor's definition of that attribute. An example encoding of the Vendor-Specific attribute using this method follows:

> Multiple sub-attributes MAY be encoded within a single Vendor-Specific attribute, although they do not have to be.

Note that the Vendor type field in the recommended VSA format is only a single octet, like the RADIUS type field. While this limitation results in an efficient encoding, there are situations in which a vendor or SDO will eventually wish to define more than 255 attributes. Also, an SDO can be comprised of multiple subgroups, each of whom can desire autonomy over the definition of attributes within their group. The most interoperable way to address these issues is for the vendor or SDO to request allocation of multiple Vendor identifiers.

However, instead of doing this, vendors have defined the following non-standard VSA formats:

* Vendor types of 16 bits, followed by an 8 bit length and then attribute-specific data.

* Vendor types of 32 bits, followed by no length field, and then attribute-specific data.

* Vendor types of the RFC format, but where some VSAs are defined as "grouped" or TLV attributes. These attributes are then used to carry sub-attributes.

* "Bare" ASCII strings that immediately follow the Vendor-Id, without using a Vendor type or Vendor length.

All VSA schemes that do not follow the [RFC2865] recommendations are NOT RECOMMENDED. These non-standard formats will typically not be implementable without RADIUS server code changes. This includes all the above formats, as well as Vendor types of more than 8 bits, vendor lengths of less than 8 bits, vendor lengths of more than 8 bits and Vendor-Specific contents that are not in Type-Length-Value format.

Although [RFC2865] does not mandate it, implementations commonly assume that the Vendor Id can be used as a key to determine the on-the-wire format of a VSA. Vendors therefore SHOULD NOT use multiple formats for VSAs that are associated with a particular Vendor Id. A vendor wishing to use multiple VSA formats SHOULD request one Vendor Id for each VSA format that they will use.

3. Data Model Issues

Since the closure of the RADIUS Working Group, the popularity and prevalence of RADIUS has continued to grow. In addition to increasing demand for allocation of attributes within the RADIUS standard attribute space, the number of vendors and SDOs creating new attributes within the Vendor-Specific attribute space has grown, and this has lead to some divergence in approaches to RADIUS attribute design.

In general, standard RADIUS attributes have a more constrained data

model than attributes within the vendor space. For example, vendors and SDOs have evolved the data model to support new functions such as attribute grouping and attribute fragmentation, with different groups taking different approaches.

Given these enhancements, it has become difficult for vendors or SDOs to translate attributes from the vendor space to the more stringent standards space. For example, a Vendor-Specific attribute using sub-elements could require allocation of several standard space attributes using basic data types. In this case not only would translation require substantial additional work, it would further deplete the RADIUS standard attribute space. Given these limitations, translation of vendor attributes to the standards space is not necessarily desirable, particularly if the VSA specification is publicly available and can be implemented within existing RADIUS clients and servers. In such situations, the costs may substantially outweigh the benefits. It is possible that some of the enhancements made within the vendor space may eventually become available within the standard attribute space. However, the divergence of the standard and vendor attribute spaces is most likely a permanent feature, and should be recognized as such.

3.1. Vendor Space

The usage model for RADIUS VSAs is described in [RFC2865] Section 6.2:

> Note that RADIUS defines a mechanism for Vendor-Specific extensions (Attribute 26) and the use of that should be encouraged instead of allocation of global attribute types, for functions specific only to one vendor's implementation of RADIUS, where no interoperability is deemed useful.

Nevertheless, many new attributes have been defined in the vendor specific space in situations where interoperability is not only useful, but is required. For example, SDOs outside the IETF (such as the IEEE 802 and the 3rd Generation Partnership Project (3GPP)) have

been assigned Vendor-Ids, enabling them to define their own VSA format and assign Vendor types within their own space.

The use of VSAs by SDOs outside the IETF has gained in popularity for

several reasons:

Efficiency
: As with SNMP, which defines an "Enterprise" Object Identifier (OID) space suitable for use by vendors as well as other SDOs, the definition of Vendor-Specific RADIUS attributes has become a common occurrence as part of standards activity outside the IETF. For reasons of efficiency, it is easiest if the RADIUS attributes required to manage a standard are developed within the same SDO that develops the standard itself. As noted in "Transferring MIB Work from IETF Bridge MIB WG to IEEE 802.1 WG" [RFC4663], today few vendors are willing to simultaneously fund individuals to participate within an SDO to complete a standard, as well as to participate in the IETF in order to complete the associated RADIUS attributes specification.

Attribute scarcity
: The standard RADIUS attribute space is limited to 255 unique attributes. Of these, only about half remain available for allocation. In the Vendor-Specific space, the number of attributes available is a function of the format of the attribute (the size of the Vendor type field).

Along with these advantages, some limitations of VSA usage are noted in [RFC2865] Section 5.26:

> This Attribute is available to allow vendors to support their own extended Attributes not suitable for general usage. It MUST NOT affect the operation of the RADIUS protocol.
>
> Servers not equipped to interpret the vendor-specific information sent by a client MUST ignore it (although it may be reported). Clients which do not receive desired vendor-specific information SHOULD make an attempt to operate without it, although they may do so (and report they are doing so) in a degraded mode.

The limitation on changes to the RADIUS protocol effectively prohibits VSAs from changing fundamental aspects of RADIUS operation, such as modifying RADIUS packet sequences, or adding new commands. However, the requirement for clients and servers to be able to operate in the absence of VSAs has proven to be less of a constraint, since it is still possible for a RADIUS client and server to mutually indicate support for VSAs, after which behavior expectations can be reset.

Therefore, RFC 2865 provides considerable latitude for development of new attributes within the vendor space, while prohibiting development of protocol variants. This flexibility implies that RADIUS attributes can often be developed within the vendor space without loss (and possibly even with gain) in functionality.

As a result, translation of RADIUS attributes developed within the vendor space into the standard space may provide only modest benefits, while accelerating the exhaustion of the standard attribute space. We do not expect that all RADIUS attribute specifications requiring interoperability will be developed within the IETF, and allocated from the standards space. A more scalable approach is to recognize the flexibility of the vendor space, while working toward improvements in the quality and availability of RADIUS attribute specifications, regardless of where they are developed.

3.1.1. Interoperability Considerations

Vendors and SDOs are reminded that the standard RADIUS attribute space, and the enumerated value space for enumerated attributes, are reserved for allocation through work published via the IETF, as noted in [RFC3575] Section 2.1. Some vendors and SDOs have in the past performed self-allocation by assigning vendor-specific meaning to "unused" values from the standard RADIUS attribute ID or enumerated value space. This self-allocation results in interoperability issues, and is counter-productive. Similarly, the Vendor-Specific enumeration practice discussed in [RFC2882] Section 2.2.1 is NOT RECOMMENDED.

If it is not possible to follow the IETF process, vendors and SDOs SHOULD self-allocate an attribute from their Vendor-Specific space, and define an appropriate value for it.

As a side note, [RFC2865] Section 5.26 uses the term "Vendor-Specific Attribute" to refer to an encoding format which can be used by individual vendors to define attributes not suitable for general usage. However, since then VSAs have also become widely used by SDOs defining attributes intended for multi-vendor interoperability. As such, these attributes are not specific to any single vendor, and the term "Vendor-Specific" may be misleading. An alternate term which better describes this use case is SDO-Specific Attribute (SSA).

The design and specification of VSAs for multi-vendor usage SHOULD be undertaken with the same level of care as standard RADIUS attributes. Specifically, the provisions of this document that apply to standard RADIUS attributes also apply to SSAs or VSAs for multi-vendor usage.

3.1.2. Vendor Allocations

Vendor RADIUS Attribute specifications SHOULD allocate attributes from the vendor space, rather than requesting an allocation from the RADIUS standard attribute space.

As discussed in [RFC2865] Section 5.26, the vendor space is intended for vendors to support their own Attributes not suitable for general use. However, it is RECOMMENDED that vendors follow the guidelines outlined here, which are intended to enable maximum interoperability with minimal changes to existing systems.

Following these guidelines means that RADIUS servers can be updated to support the vendor's equipment by editing a RADIUS dictionary. If these guidelines are not followed, then the vendor's equipment can only be supported via code changes in RADIUS server implementations. Such code changes add complexity and delay to implementations.

3.1.3. SDO Allocations

SDO RADIUS Attribute specifications SHOULD allocate attributes from the vendor space, rather than requesting an allocation from the RADIUS standard attribute space, for attributes matching any of the following criteria:

* attributes relying on data types not defined within RADIUS

* attributes intended primarily for use within an SDO

* attributes intended primarily for use within a group of SDOs.

Any new RADIUS attributes or values intended for interoperable use across a broad spectrum of the Internet Community SHOULD follow the normal IETF process, and SHOULD result in allocations from the RADIUS standard space.

The recommendation for SDOs to allocate attributes from a vendor space rather than via the IETF process is a recognition that SDOs may desire to assert change control over their own RADIUS specifications. This change control can be obtained by requesting a PEC from the

Internet Assigned Number Authority (IANA), for use as a Vendor-Id within a Vendor-Specific attribute. Further allocation of attributes inside of the VSA space defined by that Vendor-Id is subject solely to the discretion of the SDO. Similarly, the use of data types (complex or not) within that VSA space is solely under the discretion of the SDO. It is RECOMMENDED that SDOs follow the guidelines outlined here, which are intended to enable maximum interoperability with minimal changes to existing systems.

It should be understood that SDOs do not have to rehost VSAs into the standards space solely for the purpose of obtaining IETF review. Rehosting puts pressure on the standards space, and may be harmful to interoperability, since it can create two ways to provision the same service. Rehosting may also require changes to the RADIUS data model which will affect implementations that do not intend to support the SDO specifications.

3.1.4. Publication of specifications

SDOs are encouraged to seek early review of SSA specifications by the IETF. This review may be initiated by sending mail to the AAA Doctors list [DOCTORS], with the understanding that this review is a voluntary-based service offered on best-effort basis. Since the IETF is not a membership organization, in order to enable the RADIUS SSA specification to be reviewed, it is RECOMMENDED that it be made publicly available; this also encourages interoperability. Where the RADIUS SSA specification is embedded within a larger document which cannot be made public, the RADIUS attribute and value definitions SHOULD be published instead as an Informational RFC, as with [RFC4679]. This process SHOULD be followed instead of requesting IANA allocations from within the standard RADIUS attribute space.

Similarly, vendors are encouraged to make their specifications publicly available, for maximum interoperability. However, it is not necessary for them to request publication of their VSA specifications as Informational RFCs by the IETF.

All other specifications, including new authentication, authorization, and/or security mechanisms SHOULD be allocated via the standard RADIUS space, as noted in [RFC3575] Section 2.1.

3.2. Polymorphic Attributes

A polymorphic attribute is one whose format or meaning is dynamic. For example, rather than using a fixed data format, an attribute's format might change based on the contents of another attribute. Or, the meaning of an attribute may depend on earlier packets in a sequence.

RADIUS server dictionary entries are typically static, enabling the user to enter the contents of an attribute without support for changing the format based on dynamic conditions. However, this limitation on static types does not prevent implementations from implementing policies that return different attributes based on the contents of received attributes; this is a common feature of existing RADIUS implementations.

In general, polymorphism is NOT RECOMMENDED. Polymorphism rarely enables capabilities that would not be available through use of multiple attributes. Polymorphism requires code changes in the RADIUS server in situations where attributes with fixed formats would not require such changes. Thus, polymorphism increases complexity while decreasing generality, without delivering any corresponding benefits.

Note that changing an attribute's format dynamically is not the same thing as using a fixed format and computing the attribute itself dynamically. RADIUS authentication attributes such as User-Password, EAP-Message, etc. while being computed dynamically, use a fixed format.

3.3. RADIUS Operational Model

The RADIUS operational model includes several assumptions:

* The RADIUS protocol is stateless;

* Provisioning of services is not possible within an Access-Reject;

* There is a distinction between authorization checks and user authentication;

* The protocol provices for authentication and integrity protection of packets;

* The RADIUS protocol is a Request/Response protocol;

* The protocol defines packet length restrictions.

While RADIUS server implementations may keep state, the RADIUS protocol is stateless, although information may be passed from one protocol transaction to another via the State Attribute. As a result, documents which require stateful protocol behavior without use of the State Attribute are inherently incompatible with RADIUS as defined in [RFC2865], and need to be redesigned. See [RFC5080] Section 2.1.1 for a more in-depth discussion of the use of the State Attribute.

As noted in [RFC5080] Section 2.6, the intent of an Access-Reject is to deny access to the requested service. As a result, RADIUS does not allow the provisioning of services within an Access-Reject. Documents which include provisioning of services within an Access-Reject are inherently incompatible with RADIUS, and need to be redesigned.

As noted in [RFC5080] Section 2.1.1, a RADIUS Access-Request may not contain user authentication attributes or a State Attribute linking the Access-Request to an earlier user authentication. Such an Access-Request, known as an authorization check, provides no assurance that it corresponds to a live user. RADIUS specifications defining attributes containing confidential information (such as Tunnel-Password) should be careful to prohibit such attributes from being returned in response to an authorization check. Also, [RFC5080] Section 2.1.1 notes that authentication mechanisms need to tie a sequence of Access-Request/Access-Challenge packets together into one authentication session. The State Attribute is RECOMMENDED for this purpose.

While [RFC2865] did not require authentication and integrity protection of RADIUS Access-Request packets, subsequent authentication mechanism specifications such as RADIUS/EAP [RFC3579] and Digest Authentication [RFC5090] have mandated authentication and integrity protection for certain RADIUS packets. [RFC5080] Section 2.1.1 makes this behavior RECOMMENDED for all Access-Request packets,

including Access-Request packets performing authorization checks. It is expected that specifications for new RADIUS authentication mechanisms will continue this practice.

The RADIUS protocol as defined in [RFC2865] is a request-response protocol spoken between RADIUS clients and servers. A single RADIUS Access-Request packet will solicit in response at most a single Access-Accept, Access-Reject or Access-Challenge packet, sent to the IP address and port of the RADIUS Client that originated the Access-Request. Similarly, a single Change-of-Authorization (CoA)-Request packet [RFC5176] will solicit in response at most a single CoA-ACK or CoA-NAK packet, sent to the IP address and port of the Dynamic Authorization Client (DAC) that originated the CoA-Request. A single Disconnect-Request packet will solicit in response at most a single Disconnect-ACK or Disconnect-NAK packet, sent to the IP address and port of the Dynamic Authorization Client (DAC) that originated the Disconnect-Request. Changes to this model are likely to require major revisions to existing implementations and so this practice is NOT RECOMMENDED.

The Length field in the RADIUS packet header is defined in [RFC2865] Section 3. It is noted there that the maximum length of a RADIUS packet is 4096 octets. As a result, attribute designers SHOULD NOT assume that a RADIUS implementation can successfully process RADIUS packets larger than 4096 octets.

Even when packets are less than 4096 octets, they may be larger than the Path Maximum Transmission Unit (PMTU). Any packet larger than the PMTU will be fragmented, making communications more brittle as

firewalls and filtering devices often discard fragments. Transport of fragmented UDP packets appears to be a poorly tested code path on network devices. Some devices appear to be incapable of transporting fragmented UDP packets, making it difficult to deploy RADIUS in a network where those devices are deployed. We RECOMMEND that RADIUS messages be kept as small possible.

If a situation is envisaged where it may be necessary to carry authentication, authorization or accounting data in a packet larger than 4096 octets, then one of the following approaches is RECOMMENDED:

1. Utilization of a sequence of packets.
   For RADIUS authentication, a sequence of Access-Request/ Access-Challenge packets would be used. For this to be feasible, attribute designers need to enable inclusion of attributes that can consume considerable space within Access-Challenge packets. To maintain compatibility with existing NASes, either the use of Access-Challenge packets needs to be permissible (as with RADIUS/EAP, defined in [RFC3579]), or support for receipt of an Access-Challenge needs to be indicated by the NAS (as in RADIUS Location [RFC5580]). Also, the specification needs to clearly describe how attribute splitting is to be signalled and how attributes included within the sequence are to be interpreted, without requiring stateful operation. Unfortunately, previous specifications have not always exhibited the required foresight. For example, even though very large filter rules are conceivable, the NAS-Filter-Rule Attribute defined in [RFC4849] is not permitted in an Access-Challenge packet, nor is a mechanism specified to allow a set of NAS-Filter-Rule attributes to be split across an Access-Request/Access-Challenge sequence.

   In the case of RADIUS accounting, transporting large amounts of data would require a sequence of Accounting-Request packets. This is a non-trivial change to RADIUS, since RADIUS accounting clients would need to be modified to split the attribute stream across multiple Accounting-Requests, and billing servers would need to be modified to re-assemble and interpret the attribute stream.

2. Utilization of names rather than values.
   Where an attribute relates to a policy that could conceivably be pre-provisioned on the NAS, then the name of the pre-provisioned policy can be transmitted in an attribute, rather than the policy itself, which could be quite large. An example of this is the Filter-Id Attribute defined in [RFC2865] Section 5.11, which enables a set of pre-provisioned filter rules to be referenced by name.

3. Utilization of Packetization Layer Path MTU Discovery techniques, as specified in [RFC4821]. As a last resort, where the above techniques cannot be made to work, it may be possible to apply the techniques described in [RFC4821] to discover the maximum supported RADIUS packet size on the path between a

RADIUS client and a home server. While such an approach can avoid the complexity of utilization of a sequence of packets, dynamic discovery is likely to be time consuming and cannot be guaranteed to work with existing RADIUS implementations. As a result, this technique is not generally applicable.

4. IANA Considerations

This document does not require that the IANA update any existing registry or create any new registry, but includes information that affects the IANA, which:

* includes specific guidelines for Expert Reviewers appointed under the IANA considerations of [RFC3575]

* includes guidelines that recommend against self allocation from the RADIUS standard attribute space in other SDO RADIUS Attribute specifications.

* recommends that SDOs request a Private Enterprise Code (PEC) from the IANA, for use as a Vendor-Id within a Vendor-Specific attribute.

5. Security Considerations

This specification provides guidelines for the design of RADIUS attributes used in authentication, authorization and accounting. Threats and security issues for this application are described in [RFC3579] and [RFC3580]; security issues encountered in roaming are described in [RFC2607].

Obfuscation of RADIUS attributes on a per-attribute basis is necessary in some cases. The current standard mechanism for this is described in [RFC2865] Section 5.2 (for obscuring User-Password values) and is based on the MD5 algorithm specified in [RFC1321]. The MD5 and SHA-1 algorithms have recently become a focus of scrutiny and concern in security circles, and as a result, the use of these algorithms in new attributes is NOT RECOMMENDED. In addition, previous documents referred to this method as generating "encrypted" data. This terminology is no longer accepted within the cryptographic community.

Where new RADIUS attributes use cryptographic algorithms, algorithm

negotiation SHOULD be supported. Specification of a mandatory-to-implement algorithm is REQUIRED, and it is RECOMMENDED that the mandatory-to-implement algorithm be certifiable under FIPS 140 [FIPS].

Where new RADIUS attributes encapsulate complex data types, or transport opaque data, the security considerations discussed in Section 2.1.4 SHOULD be addressed.

Message authentication in RADIUS is provided largely via the Message-Authenticator attribute. See [RFC3579] Section 3.2, and also [RFC5080] 2.2.2, which says that client implementations SHOULD include a Message-Authenticator attribute in every Access-Request.

In general, the security of the RADIUS protocol is poor. Robust deployments SHOULD support a secure communications protocol such as IPSec. See [RFC3579] Section 4, and [RFC3580] Section 5 for a more in-depth explanation of these issues.

Implementations not following the suggestions outlined in this document may be subject to a problems such as ambiguous protocol decoding, packet loss leading to loss of billing information, and denial of service attacks.

6. References

6.1. 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.

6.2. Informative References

[RFC1155] Rose, M. and K. McCloghrie, "Structure and identification of management information for TCP/IP-based internets", STD 16, RFC 1155, May 1990.

[RFC1157] Case, J., Fedor, M., Schoffstall, M., and J. Davin, "Simple Network Management Protocol (SNMP)", STD 15, RFC 1157, May 1990.

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

[RFC2607] Aboba, B. and J. Vollbrecht, "Proxy Chaining and Policy Implementation in Roaming", RFC 2607, June 1999.

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

[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., Willats, W., and P. Calhoun, "RADIUS Extensions", RFC 2869, June 2000.

[RFC2882] Mitton, D, "Network Access Servers Requirements: Extended RADIUS Practices", RFC 2882, July 2000.

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

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

[RFC3580] Congdon, P., Aboba, B., Smith, A., Zorn, G., Roese, J., "IEEE 802.1X Remote Authentication Dial In User Service (RADIUS) Usage Guidelines", RFC3580, September 2003.

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

[RFC4181] Heard, C., "Guidelines for Authors and Reviewers of MIB Documents", RFC 4181, September 2005.

[RFC4663] Harrington, D., "Transferring MIB Work from IETF Bridge MIB WG to IEEE 802.1 WG", RFC 4663, September 2006.

[RFC4675] Congdon, P., Sanchez, M. and B. Aboba, "RADIUS Attributes for Virtual LAN and Priority Support", RFC 4675, September 2006.

[RFC4679] Mammoliti, V., et al., "DSL Forum Vendor-Specific RADIUS

Attributes", RFC 4679, September 2006.

[RFC4818] Salowey, J. and R. Droms, "RADIUS Delegated-IPv6-Prefix Attribute", RFC 4818, April 2007.

[RFC4821] Mathis, M. and Heffner, J, "Packetization Layer Path MTU Discovery", RFC 4821, March 2007.

[RFC4849] Congdon, P. et al, "RADIUS Filter-Rule Attribute", RFC 4849, April 2007.

[RFC5080] Nelson, D. and DeKok, A, "Common Remote Authentication Dial In User Service (RADIUS) Implementation Issues and Suggested Fixes", RFC 5080, December 2007.

[RFC5090] Sterman, B. et al., "RADIUS Extension for Digest Authentication", RFC 5090, February 2008.

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

[DOCTORS] AAA Doctors Mailing list <aaa-doctors@ops.ietf.org>

[FIPS] FIPS 140-3 (DRAFT), "Security Requirements for Cryptographic Modules", http://csrc.nist.gov/publications/fips/fips140-3/

[IEEE-802.1Q]
IEEE Standards for Local and Metropolitan Area Networks: Draft Standard for Virtual Bridged Local Area Networks, P802.1Q-2003, January 2003.

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

Appendix A - Design Guidelines

The following text provides guidelines for the design of attributes used by the RADIUS protocol. Specifications that follow these guidelines are expected to achieve maximum interoperability with minimal changes to existing systems.

A.1. Types matching the RADIUS data model

A.1.1. Transport of simple data

Does the data fit within the existing RADIUS data model, as outlined below? If so, it SHOULD be encapsulated in a [RFC2865] format RADIUS attribute, or in a [RFC2865] format RADIUS VSA that uses one of the existing RADIUS data types.

* 32-bit unsigned integer, in network byte order.

* Enumerated data types, represented as a 32-bit unsigned integer with a list of name to value mappings. (e.g., Service-Type)

* 64-bit unsigned integer, in network byte order.

* IPv4 address in network byte order.

* IPv6 address in network byte order.

* IPv6 prefix.

* time as 32 bit unsigned value, in network byte order, and in seconds since 00:00:00 UTC, January 1, 1970.

* string (i.e., binary data), totalling 253 octets or less in length. This includes the opaque encapsulation of data structures defined outside of RADIUS. See also Appendix A.1.3, below.

* UTF-8 text, totalling 253 octets or less in length.

Note that the length limitations for VSAs of type String and Text are less than 253 octets, due to the additional overhead of the Vendor-Specific format.

The following data also qualifies as "simple data types":

* Attributes grouped into a logical container, using the [RFC2868] tagging mechanism. This approach is NOT RECOMMENDED (see Section 2.1.2), but is permissible where

the alternatives are worse.

* Attributes requiring the transport of more than 247 octets of Text or String data. This includes the opaque encapsulation of data structures defined outside of RADIUS. See also Section A.1.3, below.

Nested groups or attributes do not qualify as "simple data types", and SHOULD NOT be used.

A.1.2. Transport of Authentication and Security Data

Does the data provide authentication and/or security capabilities, as outlined below? If so, it SHOULD be encapsulated in a [RFC2865] format RADIUS attribute. It SHOULD NOT be encapsulated in a [RFC2865] format RADIUS VSA.

* Complex data types that carry authentication methods which RADIUS servers are expected to parse and verify as part of an authentication process.

* Complex data types that carry security information intended to increase the security of the RADIUS protocol itself.

Any data type carrying authentication and/or security data that is not meant to be parsed by a RADIUS server is an "opaque data type", as defined below.

A.1.3. Opaque data types

Does the attribute encapsulate an existing data structure defined outside of the RADIUS specifications? Can the attribute be treated as opaque data by RADIUS servers (including proxies?) If both questions can be answered affirmatively, a complex structure MAY be used in a RADIUS specification.

The specification of the attribute SHOULD define the encapsulating attribute to be of type String. The specification SHOULD refer to an external document defining the structure. The specification SHOULD NOT define or describe the structure, as discussed above in Section 2.1.3.

A.2. Improper Data Types

All data types other than the ones described above in Appendix A.1 SHOULD NOT be used. This section describes in detail a number of data types that are NOT RECOMMENDED in new RADIUS specifications. Where possible, replacement data types are suggested.

A.2.1. Simple Data Types

Does the attribute use any of the following data types? If so, the data type SHOULD be replaced with the suggested alternatives, or it SHOULD NOT be used at all.

* Signed integers of any size.
  SHOULD NOT be used. SHOULD be replaced with one or more unsigned integer attributes. The definition of the attribute can contain information that would otherwise go into the sign value of the integer.

* 8 bit unsigned integers.
  SHOULD be replaced with 32-bit unsigned integer. There is

insufficient justification to save three bytes.

* 16 bit unsigned integers.
  SHOULD be replaced with 32-bit unsigned integer. There is insufficient justification to save two bytes.

* Unsigned integers of size other than 32 or 64.
  SHOULD be replaced by an unsigned integer of 32 or 64 bits. There is insufficient justification to define a new size of integer.

* Integers of any size in non-network byte order
  SHOULD be replaced by unsigned integer of 32 or 64 bits, in network byte order. There is no reason to transport integers in any format other than network byte order.

* Tagged data types as described in [RFC2868].
  These data types SHOULD NOT be used in new specifications.

* Complex data structures defined only within RADIUS.
  SHOULD NOT be used. This recommendation does not apply to new attributes for authentication or security, as described below in Section A.2.2.

* Multi-field text strings.
  Each field SHOULD be encapsulated in a separate attribute.

* Polymorphic attributes.
  Multiple attributes, each with a static data type SHOULD be defined instead.

* Nested AVPs.
  Attributes should be defined in a flat typespace, possibly using a 16-bit Vendor type field (see Section 2.3).

A.2.2. Complex Data Types

Does the attribute:

* define a complex data type

* That a RADIUS server and/or client is expected to parse?

i.e. A type that cannot be treated as opaque data (Section A.1.3)

* for purposes other than authentication or security?

If so, this data type SHOULD be replaced with simpler types, as discussed above in Appendix A.2.1. Also see Section 2.1.3 for a discussion of why complex types are problematic.

A.3. Vendor-Specific formats

Does the specification contain Vendor-Specific attributes that match any of the following criteria? If so, the data type should be replaced with the suggested alternatives, or should not be used at all.

* Vendor types of more than 8 bits.
  SHOULD NOT be used. Vendor types of 8 bits SHOULD be used instead.

* Vendor lengths of less than 8 bits. (i.e., zero bits)
  SHOULD NOT be used. Vendor lengths of 8 bits SHOULD be used instead.

* Vendor lengths of more than 8 bits.
  SHOULD NOT be used. Vendor lengths of 8 bits SHOULD be used instead.

* Vendor-Specific contents that are not in Type-Length-Value format.
  SHOULD NOT be used. Vendor-Specific attributes SHOULD be in Type-Length-Value format.

In general, Vendor-Specific attributes SHOULD follow the [RFC2865] suggested format. Vendor extensions to non-standard formats are NOT RECOMMENDED as they can negatively affect interoperability.

A.4. Changes to the RADIUS Operational Model

Does the specification change the RADIUS operation model, as outlined in the list below? If so, then another method of achieving the design objectives SHOULD be used. Potential problem areas include:

* Defining new commands in RADIUS using attributes.
  The addition of new commands to RADIUS MUST be handled via allocation of a new Code, and not by the use of an attribute. This restriction includes new commands created by overloading the Service-Type attribute to define new values that modify the functionality of Access-Request packets.

* Using RADIUS as a transport protocol for data unrelated to authentication, authorization, or accounting. Using RADIUS to transport authentication methods such as EAP is explicitly permitted, even if those methods require the transport of relatively large amounts of data. Transport of opaque data relating to AAA is also permitted, as discussed above in Section 2.1.3. However, if the specification does not relate to AAA, then RADIUS SHOULD NOT be used.

* Assuming support for packet lengths greater than 4096 octets.
  Attribute designers cannot assume that RADIUS implementations can successfully handle packets larger than 4096 octets. If a specification could lead to a RADIUS packet larger than 4096 octets, then the alternatives described in Section 3.3 SHOULD be considered.

* Stateless operation. The RADIUS protocol is stateless, and documents which require stateful protocol behavior without the use of the State Attribute need to be redesigned.

* Provisioning of service in an Access-Reject. Such provisioning is not permitted, and MUST NOT be used. If limited access needs to be provided, then an Access-Accept with appropriate authorizations can be used instead.

* Lack of user authentication or authorization restrictions.
  In an authorization check, where there is no demonstration of a live user, confidential data cannot be returned. Where there is a link to a previous user authentication, the State attribute needs to be present.

* Lack of per-packet integrity and authentication.
  It is expected that documents will support per-packet integrity and authentication.

* Modification of RADIUS packet sequences.
  In RADIUS, each request is encapsulated in it's own packet, and elicits a single response that is sent to the requester. Since changes to this paradigm are likely to require major modifications to RADIUS client and server implementations, they SHOULD be avoided if possible.

For further details, see Section 3.3.

A.5. Allocation of attributes

Does the attribute have a limited scope of applicability, as outlined below? If so, then the attributes SHOULD be allocated from the Vendor-Specific space.

* attributes intended for a vendor to support their own systems, and not suitable for general usage

* attributes relying on data types not defined within RADIUS

* attributes intended primarily for use within an SDO

* attributes intended primarily for use within a group of SDOs.

Note that the points listed above do not relax the recommendations discussed in this document. Instead, they recognize that the RADIUS data model has limitations. In certain situations where interoperability can be strongly constrained by the SDO or vendor, an expanded data model MAY be used. We recommend, however, that the RADIUS data model SHOULD be used, even if it is marginally less efficient than alternatives.

When attributes are used primarily within a group of SDOs, and are not applicable to the wider Internet community, we expect that one SDO will be responsible for allocation from their own private space.

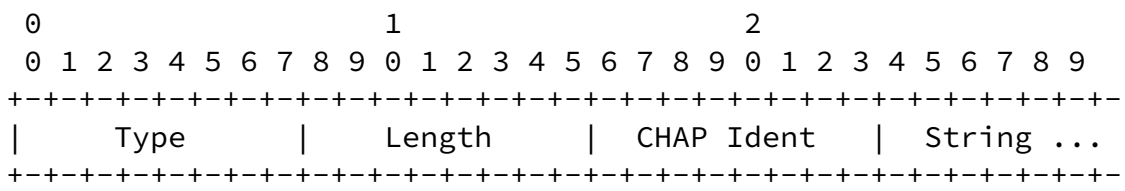
Appendix B - Complex Attributes

This section summarizes RADIUS attributes with complex data types that are defined in existing RFCs.

This appendix is published for informational purposes only, and reflects the usage of attributes with complex data types at the time of the publication of this document.

B.1. CHAP-Password

[RFC2865] Section 5.3 defines the CHAP-Password Attribute which is sent from the RADIUS client to the RADIUS server in an Access-Request. The data type of the CHAP Identifier is not given, only the one octet length:



Since this is an authentication attribute, code changes are required on the RADIUS client and server to support it, regardless of the attribute format. Therefore, this complex data type is acceptable in this situation.

B.2. CHAP-Challenge

[RFC2865] Section 5.40 defines the CHAP-Challenge Attribute which is sent from the RADIUS client to the RADIUS server in an Access-Request. While the data type of the CHAP Identifier is given, the text also says:

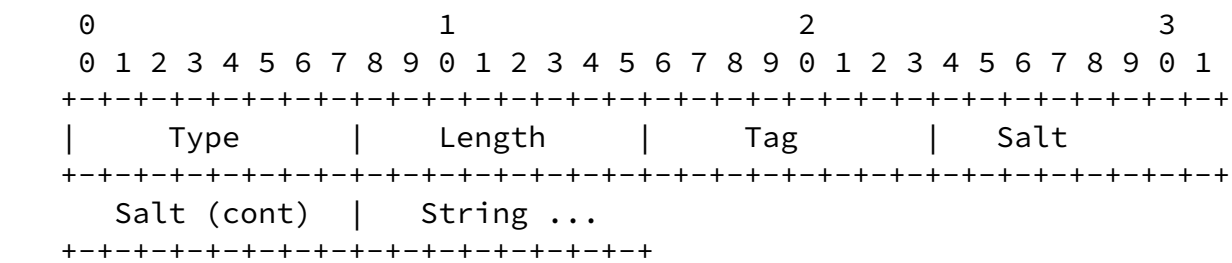
> If the CHAP challenge value is 16 octets long it MAY be placed in the Request Authenticator field instead of using this attribute.

Defining attributes to contain values taken from the RADIUS packet header is NOT RECOMMENDED. Attributes should have values that are packed into a RADIUS AVP.

B.3. Tunnel-Password

[RFC2868] Section 3.5 defines the Tunnel-Password Attribute, which is sent from the RADIUS server to the client in an Access-Accept. This attribute includes Tag and Salt fields, as well as a string field which consists of three logical sub-fields: the Data-Length (one octet) and Password sub-fields (both of which are required), and the

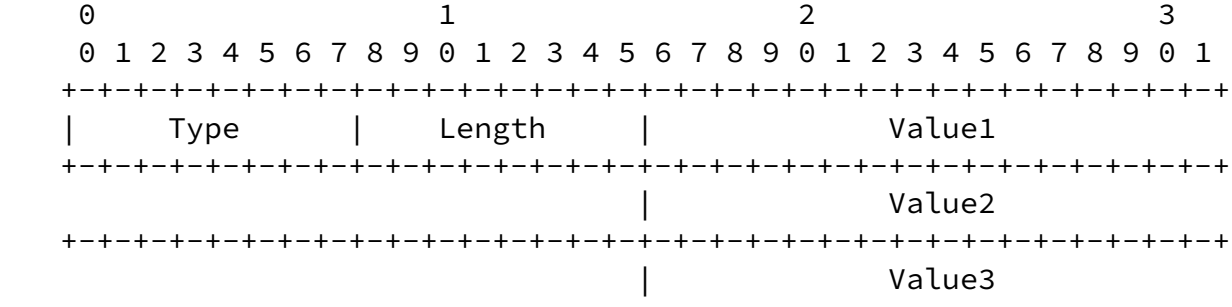
optional Padding sub-field. The attribute appears as follows:

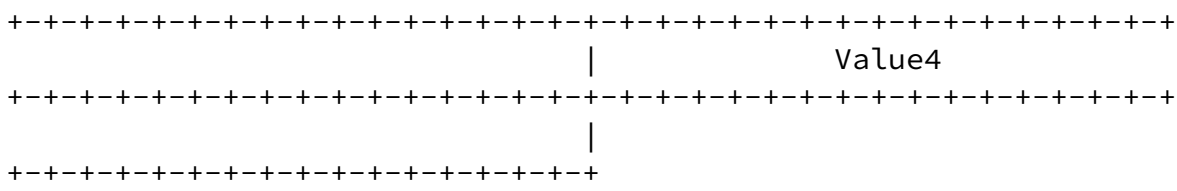


Since this is a security attribute and is encrypted, code changes are required on the RADIUS client and server to support it, regardless of the attribute format. Therefore, this complex data type is acceptable in this situation.

B.4. ARAP-Password

[RFC2869] Section 5.4 defines the ARAP-Password attribute, which is sent from the RADIUS client to the server in an Access-Request. It contains four 4 octet values, instead of having a single Value field:

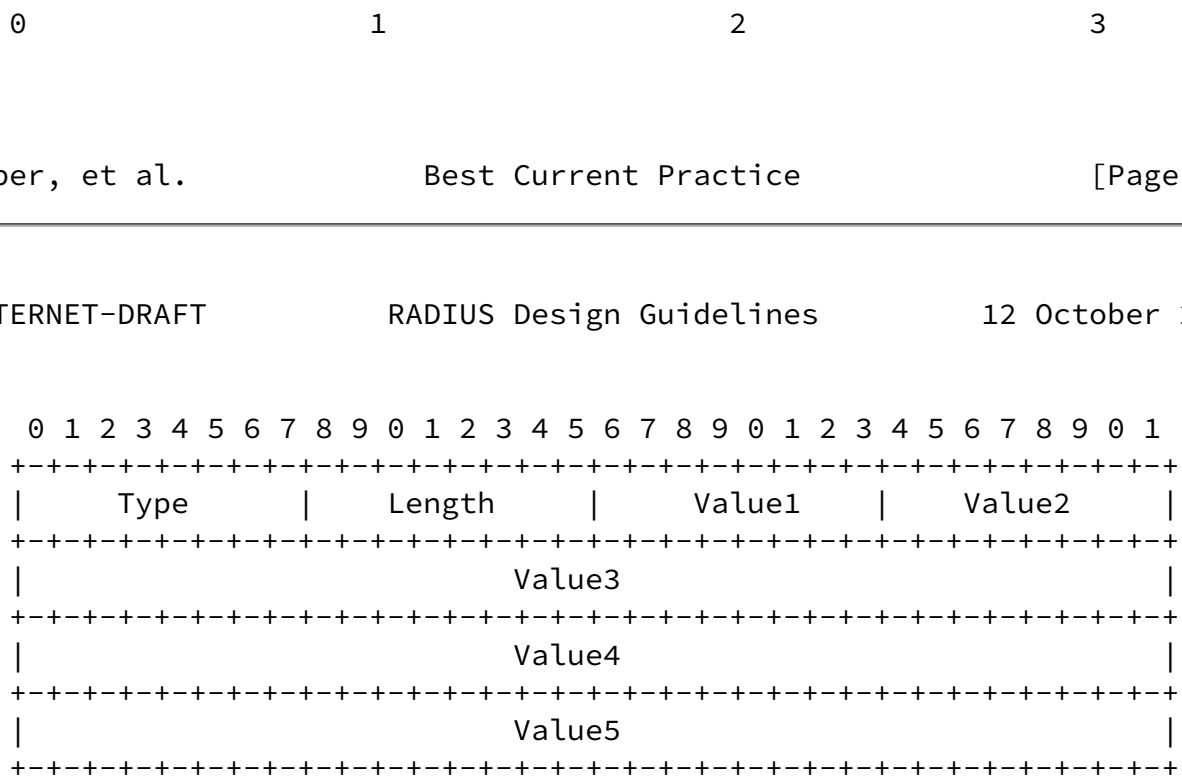




As with the CHAP-Password attribute, this is an authentication attribute which would have required code changes on the RADIUS client and server regardless of format.

B.5. ARAP-Features

[RFC2869] Section 5.5 defines the ARAP-Features Attribute, which is sent from the RADIUS server to the client in an Access-Accept or Access-Challenge. It contains a compound string of two single octet values, plus three 4-octet values, which the RADIUS client encapsulates in a feature flags packet in the ARAP protocol:



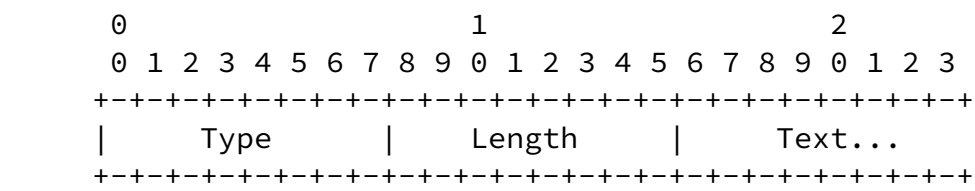
Unlike the previous attributes, this attribute contains no encrypted component, nor is it directly involved in authentication. The individual sub-fields therefore could have been encapsulated in separate attributes.

While the contents of this attribute is intended to be placed in an ARAP packet, the fields need to be set by the RADIUS server. Using standard RADIUS data types would have simplified RADIUS server implementations, and subsequent management. The current form of the attribute requires either the RADIUS server implementation, or the RADIUS server administrator, to understand the internals of the ARAP

protocol.

B.6. Connect-Info

[RFC2869] Section 5.11 defines the Connect-Info attribute, which is used to indicate the nature of the connection.



Even though the type is Text, the rest of the description indicates that it is a complex attribute:

> The Text field consists of UTF-8 encoded 10646 _8_ characters. The connection speed SHOULD be included at the beginning of the first Connect-Info attribute in the packet. If the transmit and receive connection speeds differ, they may both be included in the first attribute with the transmit speed first (the speed the NAS modem transmits at), a slash (/), the receive speed, then optionally other information.
> For example, "28800 V42BIS/LAPM" or "52000/31200 V90"
>
> More than one Connect-Info attribute may be present in an

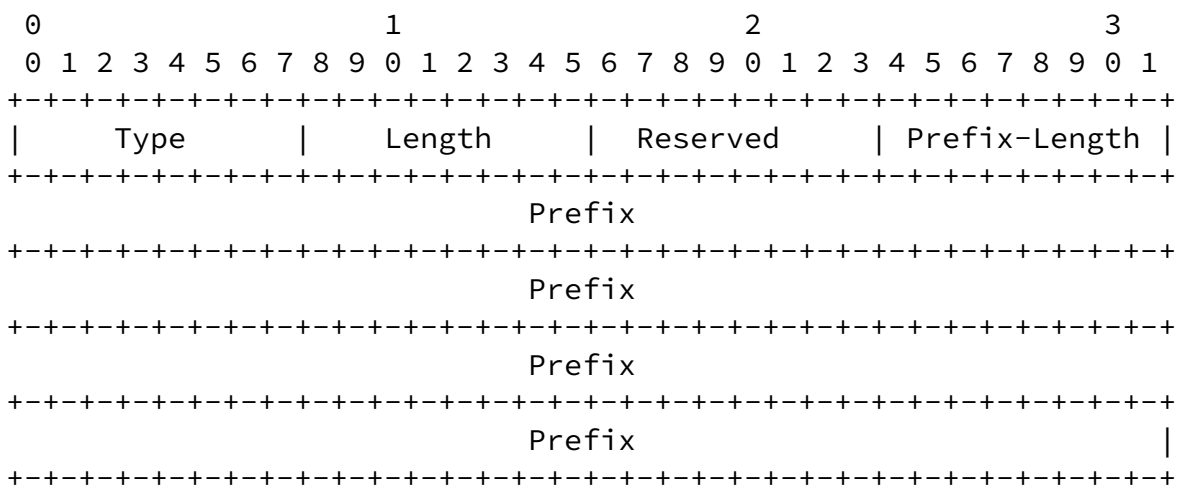
> Accounting-Request packet to accommodate expected efforts by ITU to have modems report more connection information in a standard format that might exceed 252 octets.

This attribute contains no encrypted component, and is it not directly involved in authentication. The individual sub-fields could therefore have been encapsulated in separate attributes.

Since the form of the text string is well defined, there is no benefit in using a text string. Instead, an integer attribute should have been assigned for each of the transmit speed and the receive speed. A separate enumerated integer should have been assigned for the additional information, as is done for the NAS-Port-Type attribute.

B.7. Framed-IPv6-Prefix

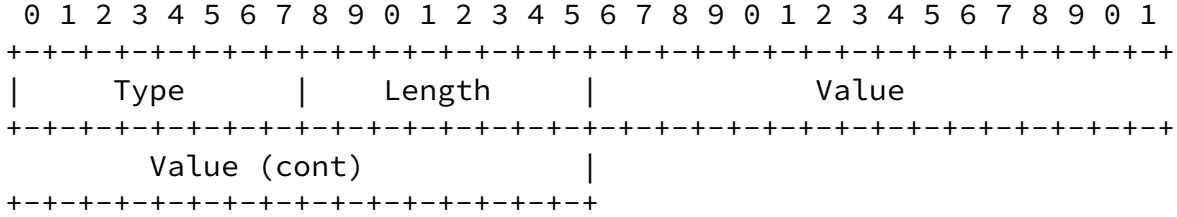
[RFC3162] Section 2.3 defines the Framed-IPv6-Prefix Attribute and [RFC4818] Section 3 reuses this format for the Delegated-IPv6-Prefix Attribute; these attributes are sent from the RADIUS server to the client in an Access-Accept.



The sub-fields encoded in these attributes are strongly related, and there was no previous definition of this data structure that could be referenced. Support for this attribute requires code changes on both the client and server, due to a new data type being defined. In this case it appears to be acceptable to encode them in one attribute.

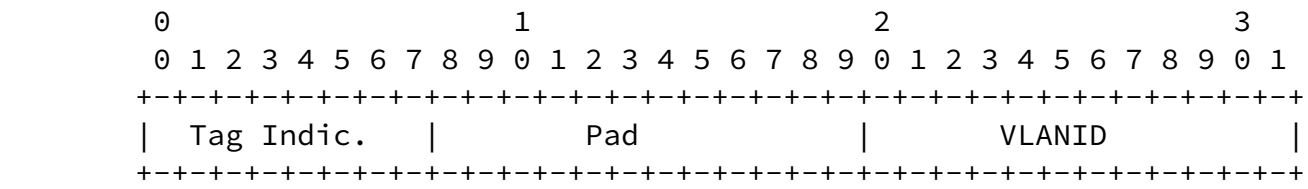
B.8. Egress-VLANID

[RFC4675] Section 2.1 defines the Egress-VLANID Attribute which can be sent by a RADIUS client or server.



While it appears superficially to be of type Integer, the Value field

is actually a packed structure, as follows:

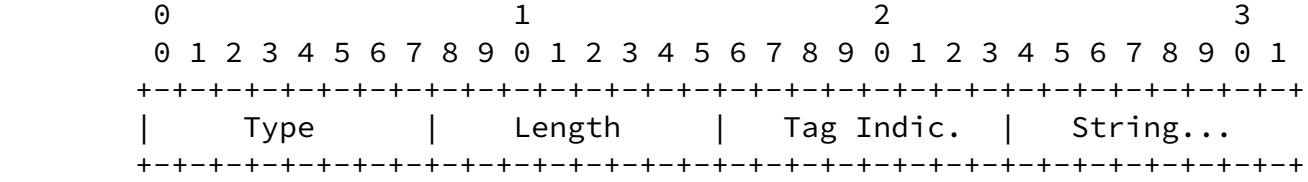


The length of the VLANID field is defined by the [IEEE-802.1Q] specification. The Tag indicator field is either 0x31 or 0x32, for compatibility with the Egress-VLAN-Name, as discussed below. The complex structure of Egress-VLANID overlaps with that of the base Integer data type, meaning that no code changes are required for a RADIUS server to support this attribute. Code changes are required on the NAS, if only to implement the VLAN ID enforcement.

Given the IEEE VLAN requirements and the limited data model of RADIUS, the chosen method is likely the best of the possible alternatives.

B.9. Egress-VLAN-Name

[RFC4675] Section 2.3 defines the Egress-VLAN-Name Attribute which can be sent by a RADIUS client or server.



The Tag Indicator is either the character '1' or '2', which in ASCII map to the identical values for Tag Indicator in Egress-VLANID, above. The complex structure of this attribute is acceptable for reasons identical to those given for Egress-VLANID.

Acknowledgments

We would like to acknowledge David Nelson, Bernard Aboba, Emile van Bergen, Barney Wolff and Glen Zorn for contributions to this

document.

Authors' Addresses

Greg Weber
Knoxville, TN 37932
USA

Email: gdweber@gmail.com

Alan DeKok
The FreeRADIUS Server Project
http://freeradius.org/

Email: aland@freeradius.org