Internet Engineering Task Force (IETF)
Request for Comments: 6158
BCP: 158
Category: Best Current Practice
ISSN: 2070-1721

A. DeKok, Ed.
FreeRADIUS
G. Weber
Individual Contributor

March 2011

RADIUS Design Guidelines

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, within the IETF as well as other Standards Development Organizations (SDOs).

Status of This Memo

This memo documents an Internet Best Current Practice.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on BCPs is available in Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc6158.

Table of Contents

1. Introduction

This document provides guidelines for the design of Remote Authentication Dial In User Service (RADIUS) attributes within the IETF as well as within other Standards Development Organizations (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 authors will check their document against the guidelines in this document prior to publication or requesting review (such as an "Expert Review" described in [RFC3575]). Similarly, it is expected that this document will be used by reviewers (such as WG participants or the Authentication, Authorization, and Accounting (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. Therefore, in addition to covering the most frequently encountered issues, this document explains some of the considerations motivating the guidelines. These considerations include complexity trade-offs that make it difficult to provide "hard and fast" rules for attribute design. This document explains those trade-offs through reviews of current attribute usage.

The rest of the document is organized as follows. Section 1 discusses the applicability of the guidelines and defines a recommended review process for RADIUS specifications. Section 2 defines the design guidelines in terms of what is "RECOMMENDED" and "NOT RECOMMENDED". Section 3 gives a longer explanation of the rationale behind the guidelines given in the previous section. Appendix A repeats the guidelines in a "checklist" format. Appendix B discusses previously defined attributes that do not follow the guidelines.

Authors of new RADIUS specifications can be compliant with the design guidelines by working through the checklists given in Appendix A. Reviewers of RADIUS specifications are expected to be familiar with the entire document.

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 accounting (AAA) server is an entity that provides one or more AAA services to a NAS.

Standard space
: Codes in the RADIUS Attribute Type Space that are allocated by IANA and that follow the format defined in Section 5 of RFC 2865 [RFC2865].

Vendor space
: The contents of the Vendor-Specific Attribute (VSA), as defined in [RFC2865], Section 5.26. These attributes provide a unique attribute type space in the "String" field for each vendor (identified by the Vendor-Type field), which they can self-allocate.

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

The advice in this document applies to RADIUS attributes used to encode service-provisioning, authentication, or accounting data based on the attribute encodings and data formats defined in RFC 2865 [RFC2865], RFC 2866 [RFC2866], and subsequent RADIUS RFCs.

Since this document represents a Best Current Practice, it does not update or deprecate existing standards. As a result, uses of the terms "MUST" and "MUST NOT" are limited to requirements already present in existing documents.

It is RECOMMENDED that these guidelines be followed for all new RADIUS specifications, whether they originate from a vendor, an SDO, or the IETF. Doing so will ensure the widest possible applicability and interoperability of the specifications, while requiring minimal changes to existing systems. In particular, it is expected that RADIUS specifications requesting allocation within the standard space will follow these guidelines and will explain why this is not possible if they cannot.

However, there are situations in which vendors or SDOs can choose not to follow these guidelines without major consequences. As noted in Section 5.26 of [RFC2865], Vendor-Specific Attributes (VSAs) are "available to allow vendors to support their own extended Attributes not suitable for general usage". Where vendors or SDOs develop specifications "not suitable for general usage", limited interoperability and inability to use existing implementations may be acceptable, and, in these situations, vendors and SDOs MAY choose not to conform to these guidelines.

Note that the RADEXT WG is currently (as of 2011) involved in developing updates to RADIUS. Those updates will provide their own usage guidelines that may modify some of the guidelines defined here, such as defining new data types, practices, etc.

RADIUS protocol changes, or specification of attributes (such as Service-Type), that can, in effect, provide new RADIUS commands require greater expertise and deeper review, as do changes to the RADIUS operational model. As a result, such changes are outside the scope of this document and MUST NOT be undertaken outside the IETF.

1.3.1. Reviews

For specifications utilizing attributes within the standard space, conformance with the design guidelines in this document is expected unless a good case can be made for an exception. Reviewers SHOULD use the design guidelines as a review checklist.

While not required, IETF review may also be beneficial for specifications utilizing the vendor space. Experience has shown that attributes not originally designed for general usage can subsequently garner wide-spread deployment. An example is the Vendor-Specific Attributes defined in [RFC2548], which have been widely implemented within IEEE 802.11 Access Points.

In order to assist in the development of specifications conforming to these guidelines, authors can request review by sending an email to the AAA Doctors [DOCTORS] or equivalent mailing list. The IETF Operations & Management Area Directors will then arrange for the review to be completed and posted to the AAA Doctors mailing list [DOCTORS], RADEXT WG mailing list, or other IETF mailing lists. Since reviews are handled by volunteers, responses are provided on a best-effort basis, with no service-level guarantees. Authors are encouraged to seek review as early as possible, so as to avoid potential delays.

As reviewers require access to the specification, vendors and SDOs are encouraged to make it publicly available. Where the RADIUS specification is embedded within a larger document that cannot be made public, the RADIUS attribute and value definitions can be made available on a public web site or can be published as an Informational RFC, as with [RFC4679].

The review process requires neither allocation of attributes within the standard space nor publication of an RFC. Requiring SDOs or vendors to rehost VSAs into the standard space solely for the purpose of obtaining review would put pressure on the standard space and may be harmful to interoperability since it would 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 or vendor specifications.

Similarly, vendors are encouraged to make their specifications publicly available, for maximum interoperability. However, it is not necessary for a vendor to request publication of a VSA specification as an RFC.

2. Guidelines

The RADIUS protocol as defined in [RFC2865] and [RFC2866] uses elements known as attributes in order to represent authentication, authorization, and accounting data.

Unlike Simple Network Management Protocol (SNMP), first defined in [RFC1157] and [RFC1155], RADIUS does not define a formal data definition language. The data type of RADIUS attributes is not transported on the wire. Rather, the data type of a RADIUS attribute is fixed when an 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.

To explain the implications of this early RADIUS design decision, we distinguish two kinds of data types, namely "basic" and "complex". Basic data types use one of the existing RADIUS data types as defined in Section 2.1, encapsulated in a [RFC2865] RADIUS attribute or in a [RFC2865] RADIUS VSA. All other data formats are "complex types".

RADIUS attributes can be classified into one of three broad categories:

* Attributes that are of interest to a single vendor, e.g., for a product or product line. Minimal cross-vendor interoperability is needed.

  Vendor-Specific Attributes (VSAs) are appropriate for use in this situation. Code-point allocation is managed by the vendor with the vendor space defined by their Private Enterprise Number (PEN), as given in the Vendor-Id field.

* Attributes that are of interest to an industry segment, where an SDO defines the attributes for that industry. Multi-vendor interoperability within an industry segment is expected.

  Vendor-Specific Attributes (VSAs) MUST be used. Code-point allocation is managed by the SDO with the vendor space defined by the SDO’s PEN rather than the PEN of an individual vendor.

* Attributes that are of broad interest to the Internet community. Multi-vendor interoperability is expected.

  Attributes within the standard space are appropriate for this purpose and are allocated via IANA as described in [RFC3575]. Since the standard space represents a finite resource, and is the only attribute space available for use by IETF working groups, vendors, and SDOs are encouraged to utilize the vendor space rather than request allocation of attributes from the standard space. Usage of attribute type codes reserved for standard attributes is considered antisocial behavior and is strongly discouraged.

2.1. Data Types

RADIUS defines a limited set of data types, defined as "basic data types". The following data qualifies as "basic 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).

* IPv4 address in network byte order.

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

* IPv6 address in network byte order.

* Interface-Id (8-octet string in network byte order).

* IPv6 prefix.

* String (i.e., binary data), totaling 253 octets or less in length. This includes the opaque encapsulation of data structures defined outside of RADIUS. See also Appendix A.1.3 for additional discussion.

* UTF-8 text [RFC3629], totaling 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 encoding.

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

* Attributes grouped into a logical container using the [RFC2868] tagging mechanism. This approach is NOT RECOMMENDED (see Section 3.2.2) but is permissible where the alternatives are worse.

* Attributes requiring the transport of more than 253 octets of Text or String data. This includes the opaque encapsulation of data structures defined outside of RADIUS, e.g., EAP-Message.

All other data formats (including nested attributes) are defined to be "complex data types" and are NOT RECOMMENDED for normal use. Complex data types MAY be used in situations where they reduce complexity in non-RADIUS systems or where using the basic data types would be awkward (such as where grouping would be required in order

to link related attributes). Since there are no "hard and fast" rules for where complexity is best located, each situation has to be decided on a case-by-case basis. Examples of this trade-off are discussed in Appendix B. Where a complex data type is selected, an explanation SHOULD be offered as to why this was necessary.

2.2. Vendor Space

The Vendor space is defined to be the contents of the Vendor-Specific Attribute ([RFC2865], Section 5.26) where the Vendor-Id defines the space for a particular vendor, and the contents of the "String" field define a unique attribute type space for that vendor. As discussed there, it is intended for vendors and SDOs to support their own attributes not suitable for general use.

While the encoding of attributes within the vendor space is under the control of vendors and SDOs, following the guidelines described here is advantageous since it enables maximum interoperability with minimal changes to existing systems.

For example, RADIUS server support for new attributes using "basic data types" can typically be accomplished by editing a RADIUS dictionary, whereas "complex data types" typically require RADIUS server code changes, which can add complexity and delays in implementation.

Vendor RADIUS Attribute specifications SHOULD self-allocate attributes from the vendor space rather than request an allocation from within the standard space.

VSA encodings that do not follow the [RFC2865], Section 5.26 encoding scheme are NOT RECOMMENDED. 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 encoding of a VSA. Vendors therefore SHOULD NOT use multiple encodings for VSAs that are associated with a particular Vendor Id. A vendor wishing to use multiple VSA encodings SHOULD request one Vendor Id for each VSA encoding that they will use.

2.3. Service Definitions and RADIUS

RADIUS specifications define how an existing service or protocol can be provisioned using RADIUS, usually via the Service-Type Attribute. 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)

Requirements for allocation of new commands (i.e., the Code field in the packet header) and new attributes within the standard space are described in [RFC3575], Section 2.1.

2.4. Translation of Vendor Specifications

[RFC2865], Section 5.26 defines Vendor-Specific Attributes as follows:

> 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 space. We do not expect that all RADIUS attribute specifications requiring interoperability will be developed within the IETF, and allocated from the standard 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.

It is therefore NOT RECOMMENDED that specifications intended solely for use by a vendor or SDO be translated into the standard space.

3. Rationale

This section outlines the rationale behind the above recommendations.

3.1. 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 or Disconnect-Request.

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

* The protocol provides 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 that require stateful protocol behavior without use of the State Attribute are inherently incompatible with RADIUS as defined in [RFC2865] and MUST be redesigned. See [RFC5080], Section 2.1.1 for additional discussion surrounding 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 or

Disconnect-Request. Documents that include provisioning of services within an Access-Reject or Disconnect-Request are inherently incompatible with RADIUS and need to be redesigned.

[RFC5176], Section 3 notes the following:

> A Disconnect-Request MUST contain only NAS and session identification attributes. If other attributes are included in a Disconnect-Request, implementations MUST send a Disconnect-NAK; an Error-Cause Attribute with value "Unsupported Attribute" MAY be included.

As a result, documents that include provisioning of services within a Disconnect-Request 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 request packet ([RFC2865], [RFC2866], or [RFC5176]) will solicit in response at most a single response packet, sent to the IP address and port of the RADIUS client that originated the request. Changes to this model are likely to require major revisions to existing implementations, and 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 signaled 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 reassemble 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.

3.2. Data Model Issues

While [RFC2865], Section 5 defines basic data types, later specifications did not follow this practice. This problem has led implementations to define their own names for data types, resulting in non-standard names for those types.

In addition, the number of vendors and SDOs creating new attributes within the vendor space has grown, and this has led to some divergence in approaches to RADIUS attribute design. For example, vendors and SDOs have evolved the data model to support functions such as new data types along with attribute grouping and attribute fragmentation, with different groups taking different approaches. These approaches are often incompatible, leading to additional complexity in RADIUS implementations.

In order to avoid repeating old mistakes, this section describes the history of the RADIUS data model and attempts to codify existing practices.

3.2.1. Issues with Definitions of Types

[RFC2865], Section 5 explicitly defines five data types: text, string, address, integer, and time. Both the names and interpretations of the types are given.

Subsequent RADIUS specifications defined attributes by using type names not defined in [RFC2865], without defining the new names as done in [RFC2865]. They did not consistently indicate the format of the value field using the same conventions as [RFC2865]. As a result, the data type is ambiguous in some cases and may not be consistent among different implementations.

It is out of the scope of this document to resolve all potential ambiguities within existing RADIUS specifications. However, in order to prevent future ambiguities, it is RECOMMENDED that future RADIUS attribute specifications explicitly define newly created data types at the beginning of the document and indicate clearly the data type to be used for each attribute.

For example, [RFC3162] utilizes, but does not explicitly define, a type that encapsulates an IPv6 address (Sections 2.1 and 2.4) and another type that encapsulates an IPv6 prefix (Section 2.3). The IPv6 address attributes confusingly are referenced as type "Address" in the document. This is a similar name as the "address" type defined in [RFC2865], which was defined to refer solely to IPv4 addresses.

While the Framed-Interface-Id Attribute defined in [RFC3162], Section 2.2 included a value field of 8 octets, the data type was not explicitly indicated; therefore, there is controversy over whether the format of the data was intended to be an 8-octet String or whether a special Interface-Id type was intended.

Given that attributes encapsulating an IPv6 address and an IPv6 prefix are already in use, it is RECOMMENDED that RADIUS server implementations include support for these as basic types, in addition to the types defined in [RFC2865]. Where the intent is to represent a specific IPv6 address, an "IPv6 address" type SHOULD be used. Although it is possible to use an "IPv6 Prefix" type with a prefix length of 128 to represent an IPv6 address, this usage is NOT RECOMMENDED. Implementations supporting the Framed-Interface-Id Attribute may select a data type of their choosing (most likely an 8-octet String or a special "Interface Id" data type).

It is worth noting that since RADIUS only supports unsigned integers of 32 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.

3.2.2. 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.

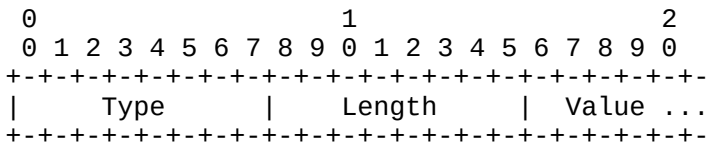
This tagging mechanism has some drawbacks. There are a limited number of unique tags (31). The tags are not well suited for use with arbitrary binary data values because it is not always possible to tell if the first byte after the Length is the tag or the first byte of the untagged value (assuming the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

3.2.3. Complex Data Types

As described in this section, the creation of complex types can lead to interoperability and deployment issues, so they need to be introduced with care. For example, the RADIUS attribute encoding is summarized in [RFC2865]:



However, some standard attributes pack multiple sub-fields into the "Value" field, resulting in the creation a non-standard, i.e., complex, type. Separating these sub-fields into different attributes, each with its own type and length, would have the following benefits:

* When manual data entry is required, it is easier for an administrator to enter the data as well-known types rather than as 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 other RADIUS server code changes.

Code changes can also be required in policy management systems and in the RADIUS server’s receive path. These changes are due to limitations in RADIUS server policy languages, which commonly provide for limited operations (such as comparisons or arithmetic operations) on the existing data types. Many existing RADIUS policy languages typically are not capable of parsing sub-elements or providing more sophisticated matching functionality.

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 limitations, the introduction of new types can require code changes on the RADIUS server, which would be unnecessary if basic data types had been used instead. In addition, if "ad hoc" types are used, attribute-specific parsing is required, which 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 reduce reliability and introduce deployment barriers.

3.2.4. Complex Data Type Exceptions

As described in Section 2.1, the introduction of complex data types is discouraged where viable alternatives are available. A potential exception is attributes that inherently require code changes on both the client and server. For example, as described in Appendix B, complex attributes have been used in situations involving authentication and security attributes, which need to be dynamically computed and verified. 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.

Another 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 Extensible Authentication Protocol (EAP) packet. Since these complex types do not need to be parsed by the RADIUS server, the issues arising from server 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 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 to RADIUS 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 to RADIUS, 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 in a specification. 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 definition.

3.3. 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 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 encoding and assign Vendor types within their own vendor space, as defined by their unique Vendor-Id.

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 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 space is limited to 255 unique attributes. Of these, only about half remain available for allocation. In the vendor space, the number of attributes available is a function of the encoding of the attribute (the size of the Vendor type field).

3.3.1. Interoperability Considerations

Vendors and SDOs are reminded that the standard 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. In the past, some vendors and SDOs have assigned vendor-specific meaning to "unused" values from the standard space. This process results in interoperability issues and is counterproductive. 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, which MUST be in their own vendor space as defined by their unique Vendor-Id, as discussed in Sections 3.3.2 and 3.3.3.

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 VSAs for multi-vendor usage.

3.3.2. Vendor Allocations

As noted in [RFC3575], Section 2.1, vendors are encouraged to utilize VSAs to define functions "specific only to one vendor’s implementation of RADIUS, where no interoperability is deemed useful. For functions specific only to one vendor’s implementation of RADIUS, the use of that should be encouraged instead of the allocation of global attribute types".

The recommendation for vendors to allocate attributes from a vendor space rather than via the IETF process is a recognition that vendors desire to assert change control over their own RADIUS specifications. This change control can be obtained by requesting a PEN from the Internet Assigned Number Authority (IANA) for use as a Vendor-Id within a Vendor-Specific Attribute. The vendor can then allocate attributes within the vendor space defined by that Vendor-Id at their sole discretion. Similarly, the use of data types (complex or otherwise) within that vendor space is solely under the discretion of the vendor.

3.3.3. SDO Allocations

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. It is therefore RECOMMENDED

that SDO RADIUS Attribute specifications allocate attributes from the vendor space rather than request an allocation from the RADIUS standard 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 allocation process defined in [RFC3575].

The recommendation for SDOs to allocate attributes from a vendor space rather than via the IETF process is a recognition that SDOs desire to assert change control over their own RADIUS specifications. This change control can be obtained by requesting a PEN from the Internet Assigned Number Authority (IANA) for use as a Vendor-Id within a Vendor-Specific Attribute. The SDO can then allocate attributes within the vendor space defined by that Vendor-Id at their sole discretion. Similarly, the use of data types (complex or otherwise) within that vendor space is solely under the discretion of the SDO.

3.4. 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.

4. IANA Considerations

This document has no action items for IANA. However, it does provide guidelines for Expert Reviewers appointed as described in [RFC3575].

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 5.1 SHOULD be addressed.

Message authentication in RADIUS is provided largely via the Message-Authenticator attribute. See Section 3.2 of [RFC3579] and also Section 2.2.2 of [RFC5080], which say 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 Section 4 of [RFC3579] and Section 5 of [RFC3580] for a more in-depth explanation of these issues.

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

5.1. New Data Types and Complex Attributes

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.

Some systems permit complex attributes to be defined via a method that is more capable than traditional RADIUS dictionaries. These systems can reduce the security threat of new types significantly, but they do not remove it entirely.

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 system 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.

Any system consuming opaque data that originates from a RADIUS system SHOULD be properly isolated from that RADIUS system and SHOULD run with minimal privileges. Any potential vulnerabilities in the non-RADIUS system will then have minimal impact on the security of the system as a whole.

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)", RFC 1157, May 1990.

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

[RFC2548] Zorn, G., "Microsoft Vendor-specific RADIUS Attributes", RFC 2548, March 1999.

[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., and J. Roese, "IEEE 802.1X Remote Authentication Dial In User Service (RADIUS) Usage Guidelines", RFC 3580, September 2003.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

[RFC4181] Heard, C., Ed., "Guidelines for Authors and Reviewers of MIB Documents", BCP 111, 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., Zorn, G., Arberg, P., and R. Rennison, "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 J. Heffner, "Packetization Layer Path MTU Discovery", RFC 4821, March 2007.

[RFC4849] Congdon, P., Sanchez, M., and B. Aboba, "RADIUS Filter Rule Attribute", RFC 4849, April 2007.

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

[RFC5090] Sterman, B., Sadolevsky, D., Schwartz, D., Williams, D., and W. Beck, "RADIUS Extension for Digest Authentication", RFC 5090, February 2008.

[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.

[DOCTORS] AAA Doctors Mailing List, www.ietf.org/mail-archive/web/aaa-doctors.

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

[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., Adrangi, F., Jones, M., Lior, A., and B. Aboba, "Carrying Location Objects in RADIUS and Diameter", RFC 5580, August 2009.

[AAA-SIP] Sterman, B., Sadolevsky, D., Schwartz, D., Williams, D., and W. Beck, "RADIUS Extension for Digest Authentication", Work in Progress, November 2004.

Appendix A. Design Guidelines Checklist

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 Basic Data Types

Does the data fit within the basic data types described in Section 2.1? 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.

A.1.2. Transport of Authentication and Security Data

Does the data provide authentication and/or security capabilities for the RADIUS protocol as outlined below? If so, use of a complex data type is acceptable under the following circumstances:

* Complex data types that carry authentication methods that 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 in Section A.1.3.

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, for reasons discussed in Section 3.2.3.

A.1.4. Pre-Existing Data Types

There is a trade-off in design between reusing existing formats for historical compatibility or choosing new formats for a "better" design. This trade-off does not always require the "better" design to be used. As a result, pre-existing complex data types described in Appendix B MAY be used.

A.2. Improper Data Types

This section suggests alternatives to data types that do not fall within the "basic data type" definition. Section A.2.1 describes simple data types, which should be replaced by basic data types. Section A.2.2 describes more complex data types, which should be replaced by multiple attributes using the basic data types.

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 bits.
  SHOULD be replaced by an unsigned integer of 32 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 bits in network. There is no reason to transport integers in any format other than network byte order.

* 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 attribute-value pairs (AVPs).
  Attributes should be defined in a flat typespace.

A.2.2. More Complex Data Types

Does the attribute:

* define a complex data type not described in Appendix B?

* that a RADIUS server and/or client is expected to parse, validate, or create the contents of via a dynamic computation (i.e., a type that cannot be treated as opaque data (Section A.1.3))?

* involve functionality that could be implemented without code changes on both the client and server (i.e., a type that doesn't require dynamic computation and verification, such as those performed for authentication or security attributes)?

If so, this data type SHOULD be replaced with simpler types, as discussed in Appendix A.2.1. See also Section 2.1 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 VSA encoding should be replaced with the [RFC2865], Section 5.26 encoding 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 encoding suggested in Section 5.26 of [RFC2865]. Vendor extensions to non-standard encodings 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 the following:

* 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 in Section 3.2.3. 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 that 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.

* Provisioning of service in a Disconnect-Request.
  Such provisioning is not permitted and MUST NOT be used. If limited access needs to be provided, then a CoA-Request [RFC5176] 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 SHOULD 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 its 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 3.1.

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 space rather than requesting allocation from the standard 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. It is RECOMMENDED, however, that the RADIUS data model be used, even when 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 vendor space.

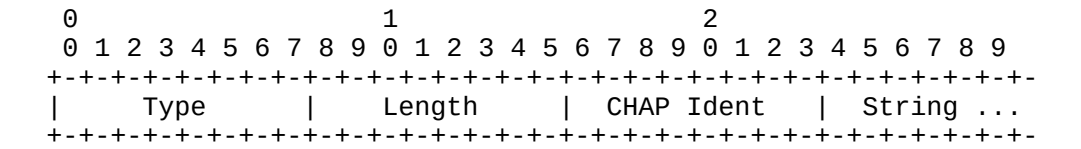
Appendix B. Complex Attributes

This appendix 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

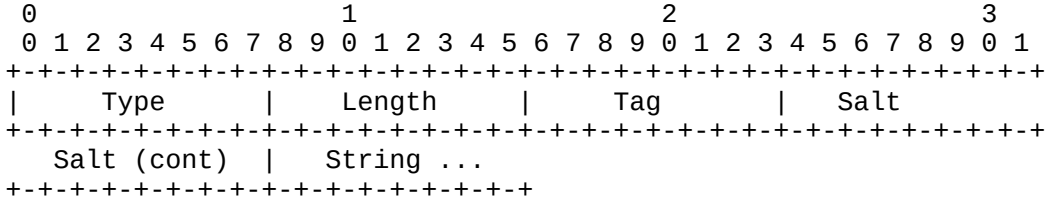
[RFC2865], Section 5.40 defines the CHAP-Challenge Attribute, which is sent from the RADIUS client to the RADIUS server in an Access-Request. While the data type of the CHAP Identifier is given, the text also says:

> 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

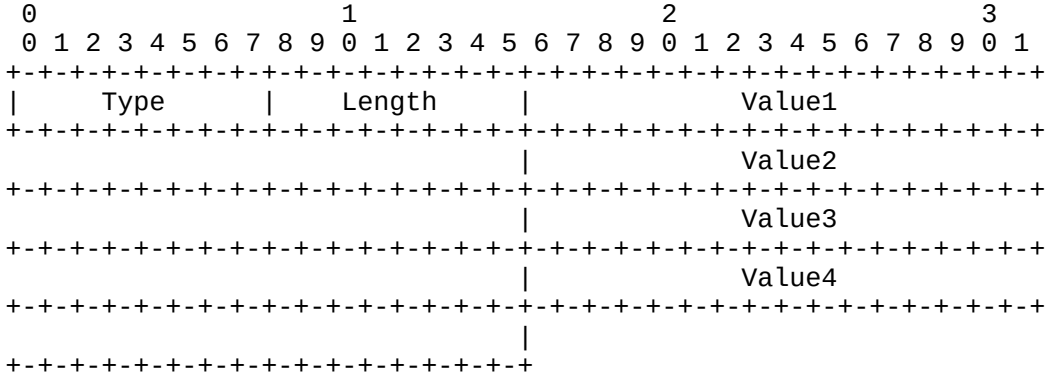
[RFC2868], Section 3.5 defines the Tunnel-Password Attribute, which is sent from the RADIUS server to the client in an Access-Accept. This attribute includes Tag and Salt fields, as well as a String field that consists of three logical sub-fields: the Data-Length (required and one octet), Password sub-fields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



Since this is a security attribute, code changes are required on the RADIUS client and server to support it, regardless of the attribute format. However, while use of a complex data type is acceptable in this situation, the design of the Tunnel-Password Attribute is problematic from a security perspective since it uses MD5 as a cipher and provides a password to a NAS, potentially without proper authorization.

B.4. ARAP-Password

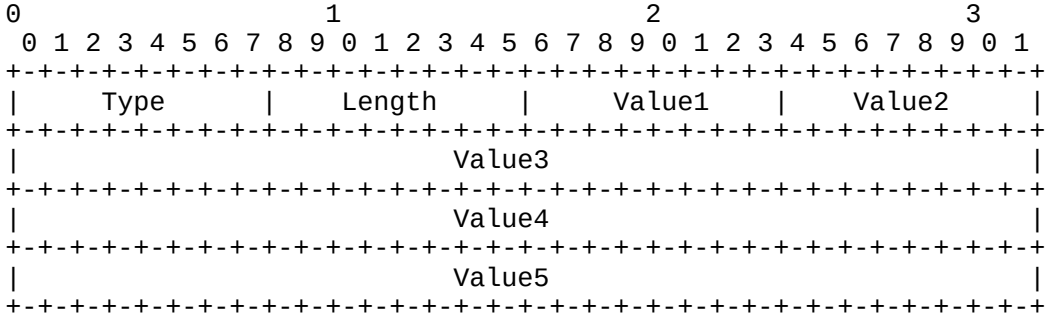
[RFC2869], Section 5.4 defines the ARAP-Password Attribute, which is sent from the RADIUS client to the server in an Access-Request. It contains four 4-octet values instead of having a single Value field:



As with the CHAP-Password Attribute, this is an authentication attribute that 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 Apple Remote Access Protocol (ARAP):

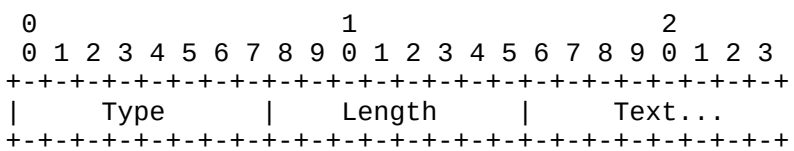


Unlike the previous attributes, this attribute contains no encrypted component, nor is it directly involved in authentication. The individual sub-fields therefore could have been encapsulated in separate attributes.

While the contents of this attribute are 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:

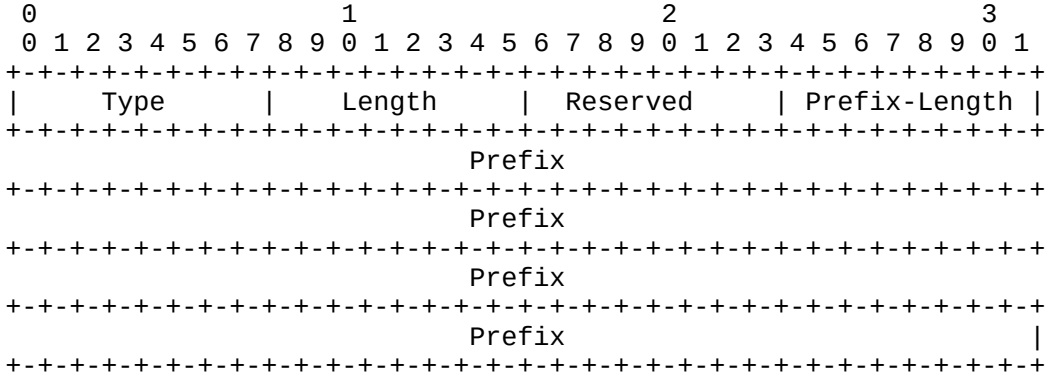
> The Text field consists of UTF-8 encoded 10646 [8] characters. The connection speed SHOULD be included at the beginning of the first Connect-Info attribute in the packet. If the transmit and receive connection speeds differ, they may both be included in the first attribute with the transmit speed first (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 not directly involved in authentication. The individual sub-fields could therefore have been encapsulated in separate attributes.

However, since the definition refers to potential standardization activity within ITU, the Connect-Info Attribute can also be thought of as opaque data whose definition is provided elsewhere. The Connect-Info Attribute could therefore qualify for an exception as described in Section 3.2.4.

B.7. Framed-IPv6-Prefix

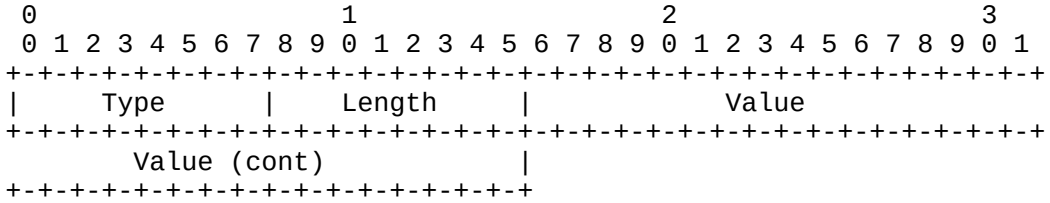
Section 2.3 of [RFC3162] defines the Framed-IPv6-Prefix Attribute, and Section 3 of [RFC4818] reuses this format for the Delegated-IPv6-Prefix Attribute; these attributes are sent from the RADIUS server to the client in an Access-Accept.



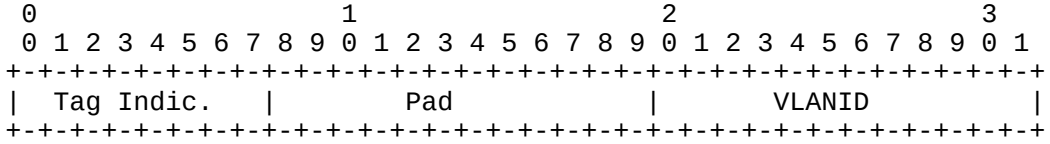
The sub-fields encoded in these attributes are strongly related, and there was no previous definition of this data structure that could be referenced. Support for this attribute requires code changes on both the client and server, due to a new data type being defined. In this case, it appears to be acceptable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

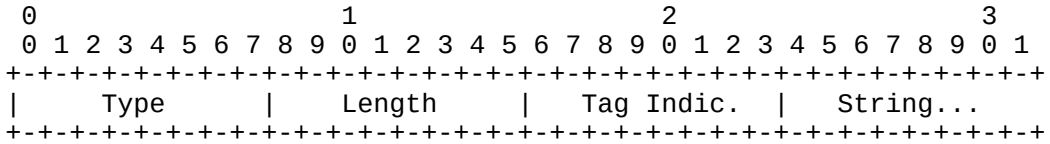


The length of the VLANID field is defined by the [IEEE-802.1Q] specification. The Tag Indicator field is either 0x31 or 0x32, for compatibility with the Egress-VLAN-Name, as discussed below. The complex structure of Egress-VLANID overlaps with that of the base Integer data type, meaning that no code 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.

B.10. Digest-*

[RFC5090] attempts to standardize the functionality provided by an expired Internet-Draft [AAA-SIP], which improperly uses two attributes from the standard space without having been assigned them by IANA. This self-allocation is forbidden, as described in Section 2. In addition, the document uses nested attributes, which are discouraged in Section 2.1. The updated document uses basic data types and allocates nearly 20 attributes in the process.

However, the document has seen wide-spread implementation, but [RFC5090] has not. One explanation may be that implementors disagreed with the trade-offs made in the updated specification. It may have been better to simply document the existing format and request IANA allocation of two attributes. The resulting design would have used nested attributes but may have gained more wide-spread implementation.

Acknowledgments

We would like to acknowledge David Nelson, Bernard Aboba, Emile van Bergen, Barney Wolff, Glen Zorn, Avi Lior, and Hannes Tschofenig for contributions to this document.

Authors' Addresses

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

EMail: aland@freeradius.org

Greg Weber
Knoxville, TN 37932
USA

EMail: gdweber@gmail.com