Workgroup: RADEXT Working Group
Internet-Draft: draft-dekok-radext-sradius-00
Published: 13 October 2022
Intended Status: Standards Track
Expires: 16 April 2023
Authors: A. DeKok
FreeRADIUS

# Secure RADIUS

## Abstract

This document defines Secure RADIUS (SRADIUS), which is a transport profile for RADIUS. There are three changes from traditional RADIUS transport protocols. First, TLS transport is required and insecure transports are forbidden. Second, the shared secret is no longer used, and all MD5-based packet signing and attribute obfuscation methods are therefore no longer necessary. Finally, the now unused Authenticator field is repurposed to contain an explict request / response identifier, called a Token.

SRADIUS connections can transport all RADIUS attributes. Implementation of SRADIUS requires only minor changes to packet encoder and decoder functionality. Nothing else is changed from traditional RADIUS.

## About This Document

This note is to be removed before publishing as an RFC.

Status information for this document may be found at https://datatracker.ietf.org/doc/draft-dekok-radext-sradius/.

Discussion of this document takes place on the RADEXT Working Group mailing list (mailto:radext@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/radext/.

Source for this draft and an issue tracker can be found at https://github.com/freeradius/sradius.git.

## Status of This Memo

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

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

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

This Internet-Draft will expire on 16 April 2023.

## Copyright Notice



## Table of Contents

## 1. Introduction

The RADIUS protocol [RFC2865] uses MD5 [RFC1321] to sign packets, and to obfuscate certain attributes. As noted in [RFC6151], MD5 is insecure and should no longer be used. In addition, the dependency on MD5 makes it impossible to use RADIUS in a FIPS-140 compliant system.

This document proposes Secure RADIUS (SRADIUS), which is a transport profile for RADIUS. Systems which implement SRADIUS are therefore capable of being FIPS-140 compliant.

The changes from traditional RADIUS transports are as follows:

- *TLS or DTLS transport is required,
- *TLS 1.3 or later is required,
- *As the security of the protocol now depends on TLS, all uses of the shared secret have been removed,
- *The now-unused Request and Response Authenticator field can be repurposed to carry an opaque Token which identifies requests and responses,
- *The Message-Authenticator attribute ([RFC3579] Section 3.2) is not sent in any packet, and if received is ignored,
- *Attributes such as User-Password, Tunnel-Password, and MS-MPPE keys are sent without the previous MD5-based obfuscation, as the contents are protected by TLS,
- *All other attributes including CHAP, MS-CHAP, and MS-CHAPv2 can still be carried inside of SRADIUS.

If a home server chooses to implement SRADIUS, it can also choose to also require full FIPS-140 compliance. In which case the home server will not support CHAP or MS-CHAP. However, it is still possible for a FIPS-140 compliant home server to accept authentication methods which depend on MD4 or MD5, so long as those methods are passed somehow to a secondary server which supports them.

We note that the decision to support (or not) any authentication method is entirely site local, and is not a requirement of the SRADIUS transport. As a transport profile for RADIUS, SRADIUS explicitly does not modify the content or meaning of any RADIUS attribute.

We also note that any proxies which accept or originate SRADIUS connections are able to transport CHAP and MS-CHAP without issue.

Unless otherwise described in this document, all RADIUS requirements apply to SRADIUS. That is, SRADIUS is a transport profile for RADIUS. It is not a new protocol, and it is not an extension to the RADIUS protocol. SRADIUS does not change the RADIUS packet format, attribute format, etc.

## 2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

*RADIUS

The Remote Authentication Dial-In User Service protocol, as defined in [RFC2865], [RFC2865], and [RFC5176] among others.

*RADIUS/UDP

RADIUS over the User Datagram Protocol as define above.

*RADIUS/TCP

RADIUS over the Transmission Control Protocol [RFC6613]

*RADIUS/TLS

RADIUS over the Transport Layer Security protocol [RFC6614]

*RADIUS/DTLS

RADIUS over the Datagram Transport Layer Security protocol [RFC7360]

*SRADIUS

The Secure RADIUS protocol ("S" RADIUS), as defined in this document. We use SRADIUS interchangeable for TLS and for DTLS transport.

*TLS

the Transport Layer Security protocol. Generally when we refer to TLS in this document, we are referring to TLS or DTLS transport.

## 3. The SRADIUS Transport profile for RADIUS

SRADIUS is a transport profile for RADIUS. In addition to defining the transport, we also discuss how the encoding of some attributes is changed when transported in SRADIUS. Any field or attribute not mentioned here is unchanged from RADIUS.

### 3.1. Transport

SRADIUS connections MUST use TLS [RFC6614] or DTLS [RFC7360] transport protocols. The insecure UDP [RFC2865] and TCP [RFC6613] transport protocols MUST NOT be used.

SRADIUS implementations MUST require TLS version 1.3 or later. There is no reason to use any earlier version of TLS.

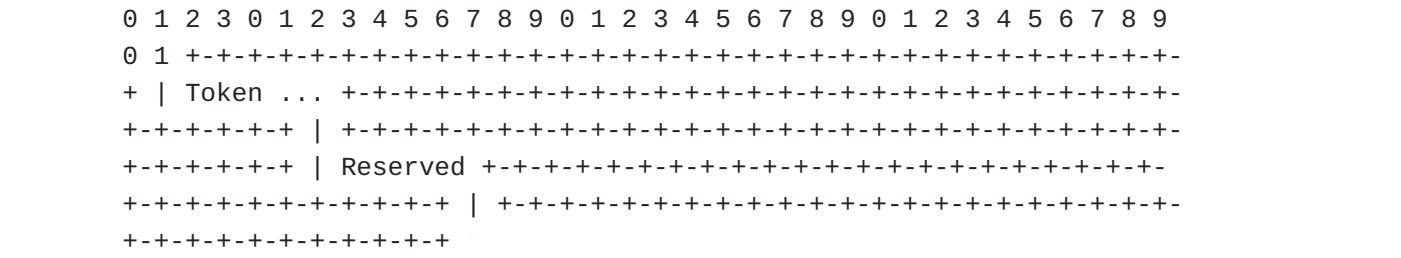
SRADIUS implementations MUST support TLS-PSK. The default profile is to have as few changes as possible from RADIUS.

### 3.2. Request and Response Authenticator fields

The Request and Response Authenticator fields MUST NOT be calculated as described in any previous RADIUS specification. Instead, those fields are not used to sign packets. That 16-octet portion of the packet header is now repurposed as two logical subfields:

- 8 octets of opaque Token used to match requests and responses,

- 8 octets of Reserved. These octets MUST be set to zero when sending any SRADIUS packet. These octets MUST be ignored when receiving any SRADIUS packet. These octets are reserved for future protocol extensions.




The Token field MUST be different for every unique packet sent over a particular SRADIUS connection. This unique value can be used to match responses to requests, and to identify duplicate requests. Other than those two requirements, there is no special meaning for the Token field.

### 3.2.1. Sending Packets

The Token field MUST change for every new SRADIUS packet which is sent. For DTLS transport, it is possible to retransmit duplicate packets, in which case the Token MUST NOT be changed when a duplicate packet is sent.

The Token MUST be different for different packets on the same connection.

It is RECOMMENDED that the Token field be a implemented as a 64-bit counter. Such a counter SHOULD be initialized from a random number generator whenever a client reboots. It is NOT RECOMMENDED to use the current time as the seed for the random number generator, or for the initial Token value unless that time is carried forward across reboots via a hardware clock. Without a hardware clock, the systems value for the current time is likely to reset to a pre-set fixed value.

The counter SHOULD be unique per connection, and SHOULD be initialized to a different value for each connection. The counter may be globally unique to an implementation, but having a unique counter per connection is acceptable.

### 3.2.2. Recieving Packets

A system which recieves SRADIUS packets MUST perform packet deduplication for all situations where it is required by RADIUS. Where RADIUS does not require deduplication (e.g. TLS transport), the SRADIUS system SHOULD NOT do deduplication.

In normal RADIUS, the Identifier field can be the same for different types of packets on the same connection, e.g. for Access-Request and Accounting-Request. This overlap leads to increased complexity for RADIUS clients and servers, as the Identifier field is not, in fact, a unique identifier. Implementations of RADIUS therefore need do deduplication across multiple fields of the RADIUS packet header, which can be complex.

For SRADIUS, implementations MUST do deduplication solely on the Token field on a per-connection basis. This change from RADIUS simplifies implementations. In addition, a unique 64-bit counter is more than sufficient to uniquely identify packets.

This change from RADIUS means that the Identifier field is no longer useful. It is RECOMMENDED that the Identifier field be set to zero for all SRADIUS packets. In order to stay close to RADIUS, replies MUST use the same Identifier as was seen in the corresponding request. There is no reason to make major changes to the RADIUS packet header.

## 4. Attribute handling in SRADIUS

Most attributes in RADIUS have no special encoding "on the wire", or any special meaning between client and server. Unless discussed in this section, all RADIUS attributes are unchanged in SRADIUS. This requirement includes attributes which contain a tag [RFC2868].

### 4.1. Obfuscated attributes such as User-Password and Tunnel-Password

Attributes which are obfuscated with MD5 no longer have the obfuscation step applied in SRADIUS. Instead, there are simply encoded as their values, as with any other attribute. Their encoding method MUST follow the encoding for the underlying data type, with any encryption / obfuscation step omitted.

We note that there is often concern in RADIUS that passwords are sent "in cleartext" across the network. This allegation was never true for RADIUS, and is still untrue for SRADIUS. While passwords are encoded in packets as strings, the packets (and thus passwords) are protected by TLS. The same TLS which protects passwords used for web logins, e-mail reception and sending, etc. As a result, any claims that passwords are sent "in the clear" are false.

The User-Password attribute ([RFC2865] Section 5.2) MUST be encoded the same as any other attribute of data type 'text' ([RFC8044] Section 3.4), e.g. User-Name ([RFC2865] Section 5.1).

The Tunnel-Password attribute ([RFC2868] Section 3.5) MUST be encoded the same as any other attribute of data type 'text' which contains a tag, such as Tunnel-Client-Endpoint ([RFC2868] Section 3.3). Since the attribute is no longer obfuscated, there is no need for a salt field or Data-Length fields as described in [RFC2868] Section 3,5, and the textual value can simply be encoded as-is.

Any Vendor-Specific attribute which uses similar obfuscation MUST be encoded as per their base data type. Specifically, the MS-MPPE-Send-Key and MS-MPPE-Recv-Key attributes ([RFC2548] Section 2.4) MUST be encoded as any other attribute of data type 'string' ([RFC8044] Section 3.5).

### 4.2. Message-Authenticator

The Message-Authenticator attribute ([RFC3579] Section 3.2) MUST NOT be sent over an SRADIUS connection. It is no longer used or needed.

If the Message-Authenticator attribute is received over an SRADIUS connection, the attribute MUST be silently discarded, or treated an as "invalid attribute", as defined in [RFC6929], Section 2.8. That is, the Message-Authenticator attribute is no longer used to sign packets. Its existence (or not) is meaningless in SRADIUS.

However, any packet which contains a Message-Authenticator attribute can still be processed. There is no need to discard the entire packet when a Message-Authenticator attribute is received.

#### 4.2.1. Message-Authentication-Code

Similarly, the Message-Authentication-Code attribute defined in [RFC6218] Section 3.3 MUST NOT be sent over an SRADIUS connection. It MUST be treated the same was as Message-Authenticator, above.

As the Message-Authentication-Code attribute is no longer used, the related MAC-Randomizer attribute [RFC6218] Section 3.2 is also no longer used. It MUST be treated the same was as Message-Authenticator, above.

### 4.3. CHAP, MS-CHAP, etc.

While some attributes such as CHAP-Password, etc. depend on insecure cryptographic primitives such as MD5, these attributes are treated as opaque blobs when sent between a RADIUS client and server. The attributes are not obfuscated, and they do not depend on the shared secret.

As a result, these attributes are unaffected by SRADIUS. We reiterate that SRADIUS is a transport profile for RADIUS. Other than Message-Authenticator, the meaning of all attributes in SRADIUS is identical to their meaning in RADIUS. Only the "on the wire" encoding of some attributes change, and then only for attributes which are obfuscated using the shared secret. Those obfuscated attributes are now protected by the modern cryptography in TLS, instead of an "ad hoc" approach using MD5.

An SRADIUS server can proxy CHAP, MS-CHAP, etc. without any issue. An SRADIUS home server can authenticate CHAP, MS-CHAP, etc. without any issue.

## 5. Proxies

We reiterate that SRADIUS is a transport profile for RADIUS. A RADIUS proxy normally decodes, and then re-encodes attributes, included obfuscated ones. A RADIUS proxy will not generally rewrite the content of the attributes it proxies. While some attributes may be modified due to administrative / policy rules on the proxy, the proxy will generally not rewrite the contents of User-Password, CHAP-Password, MS-CHAP-Password, MS-MPPE keys, etc.

The same requirement applies to a proxy which uses SRADIUS. The proxy may receive RADIUS or SRADIUS, and it may send RADIUS or SRADIUS, in any combination. As a result, SRADIUS is fully compatible with all past, present, and future RADIUS attributes.

## 6. Crypto-Agility

The crypto-agility requirements of [RFC6421] are addressed in [RFC6614] Appendix C, and in Section 10.1 of [RFC7360]. SRADIUS makes no changes from, or additions to, those specifications.

This document adds the requirement that any new RADIUS or SRADIUS specification MUST NOT introduce new cryptographic primitives as was done with User-Password and Tunnel-Password. There is insufficient expertise in the RADIUS community to securely design new cryptography.

## 7. Implementation Status

SRADIUS is implemented in a branch of FreeRADIUS which is hosted on GitHub.

## 8. Privacy Considerations

SRADIUS requires secure transport for RADIUS, and this has all of the privacy benefits of RADIUS/TLS [RFC6614] and RADIUS/DTLS [RFC7360]. All of the insecure uses of RADIUS bave been removed.

## 9. Security Considerations

The primary focus of this document is addressing security considerations for RADIUS.

## 10. IANA Considerations

IANA is request to allocate two new ports:

SRADIUS/UDP - TBD

SRADIUS/TCP - TBD

## 11. Acknowledgements

In hindsight, the decision to retain MD5 for RADIUS/TLS was likely wrong. It was an easy decision in the short term, but it has caused ongoing problems which this document addresses.

## 12. Changelog

## 13. References

### 13.1. Normative References

**[BCP14]**

Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC2865] Rigney, C., Willens, S., Rubens, A., and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, DOI 10.17487/RFC2865, June 2000, <https://www.rfc-editor.org/info/rfc2865>.

[RFC6421] Nelson, D., Ed., "Crypto-Agility Requirements for Remote Authentication Dial-In User Service (RADIUS)", RFC 6421, DOI 10.17487/RFC6421, November 2011, <https://www.rfc-editor.org/info/rfc6421>.

[RFC6929] DeKok, A. and A. Lior, "Remote Authentication Dial In User Service (RADIUS) Protocol Extensions", RFC 6929, DOI 10.17487/RFC6929, April 2013, <https://www.rfc-editor.org/info/rfc6929>.

[RFC8044] DeKok, A., "Data Types in RADIUS", RFC 8044, DOI 10.17487/RFC8044, January 2017, <https://www.rfc-editor.org/info/rfc8044>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

### 13.2. Informative References

[RFC1321] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, DOI 10.17487/RFC1321, April 1992, <https://www.rfc-editor.org/info/rfc1321>.

[RFC2548] Zorn, G., "Microsoft Vendor-specific RADIUS Attributes", RFC 2548, DOI 10.17487/RFC2548, March 1999, <https://www.rfc-editor.org/info/rfc2548>.

[RFC2868] Zorn, G., Leifer, D., Rubens, A., Shriver, J., Holdrege, M., and I. Goyret, "RADIUS Attributes for Tunnel Protocol Support", RFC 2868, DOI 10.17487/RFC2868, June 2000, <https://www.rfc-editor.org/info/rfc2868>.

[RFC3579] Aboba, B. and P. Calhoun, "RADIUS (Remote Authentication Dial In User Service) Support For Extensible Authentication Protocol (EAP)", RFC 3579, DOI 10.17487/

RFC3579, September 2003, <https://www.rfc-editor.org/info/rfc3579>.

**[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, DOI 10.17487/RFC5176, January 2008, <https://www.rfc-editor.org/info/rfc5176>.

**[RFC6151]** Turner, S. and L. Chen, "Updated Security Considerations for the MD5 Message-Digest and the HMAC-MD5 Algorithms", RFC 6151, DOI 10.17487/RFC6151, March 2011, <https://www.rfc-editor.org/info/rfc6151>.

**[RFC6218]** Zorn, G., Zhang, T., Walker, J., and J. Salowey, "Cisco Vendor-Specific RADIUS Attributes for the Delivery of Keying Material", RFC 6218, DOI 10.17487/RFC6218, April 2011, <https://www.rfc-editor.org/info/rfc6218>.

**[RFC6613]** DeKok, A., "RADIUS over TCP", RFC 6613, DOI 10.17487/RFC6613, May 2012, <https://www.rfc-editor.org/info/rfc6613>.

**[RFC6614]** Winter, S., McCauley, M., Venaas, S., and K. Wierenga, "Transport Layer Security (TLS) Encryption for RADIUS", RFC 6614, DOI 10.17487/RFC6614, May 2012, <https://www.rfc-editor.org/info/rfc6614>.

**[RFC7360]** DeKok, A., "Datagram Transport Layer Security (DTLS) as a Transport Layer for RADIUS", RFC 7360, DOI 10.17487/RFC7360, September 2014, <https://www.rfc-editor.org/info/rfc7360>.

## Author's Address

Alan DeKok
FreeRADIUS

Email: aland@freeradius.org