Workgroup: RADEXT Working Group
Internet-Draft:
draft-dekok-radext-deprecating-radius-00
Published: 4 October 2022
Intended Status: Standards Track
Expires: 7 April 2023
A A. DeKok
uFreeRADIUS
t
h
o
r
s
:

Deprecating RADIUS/UDP and RADIUS/TCP

Abstract

RADIUS crypto-agility was first mandated as future work by RFC 6421. The outcome of that work was the publication of RADIUS over TLS (RFC 6614) and RADIUS over DTLS (RFC 7360) as experimental documents. Those transport protocols have been in wide-spread use for many years in a wide range of networks. They have proven their utility as replacements for the previous UDP (RFC 2865) and TCP (RFC 6613) transports. With that knowledge, the continued use of insecure transports for RADIUS has serious and negative implications for privacy and security.

This document formally deprecates the use of the User Datagram Protocol (UDP) and of the Transport Congestion Protocol (TCP) as transport protocols for RADIUS. These transports are permitted inside of secure networks, but their use even in that environment is strongly discouraged. For all other environments, the use of TLS-based transports is mandated.

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-deprecating-radius/.

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/deprecating-radius.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 7 April 2023.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction

The RADIUS protocol [RFC2865] was first standardized in 1997, though its roots go back much earlier to 1993. The protocol uses MD5

[RFC1321] to sign some packets types, and to obfsucate certain attributes such as User-Password. As originally designed, Access-Request packets were entirely unauthenticated, and could be trivially spoofed as discussed in [RFC3579] Section 4.3.2. In order to prevent such spoofing, that specification defined the Message-Authenticator attribute ([RFC3579] Section 3.2) which allowed for packets to carry a signature based on HMAC-MD5.

The state of MD5 security was discussed in [RFC6151], which led to the state of RADIUS security being reviewed in [RFC6421] Section 3. The outcome of that review was the remainder of [RFC6421], which created crypto-agility requirements for RADIUS.

RADIUS/TLS [RFC6614] and RADIUS/DTLS [RFC7360] were then defined in order to meet the crypto-agility requirements of [RFC6421]. RADIUS/TLS has been in wide-spread use for about a decade, including Eduroam, and more recently OpenRoaming. RADIUS/DTLS has seen less use across the public Internet, but it nonetheless has multiple implementations.

As of the writing of this specification, RADIUS/UDP is still widely used, even though it depends on MD5 and "ad hoc" constructions for security. While MD5 has been broken, it is a testament to the design of RADIUS that there have been (as yet) no attacks on RADIUS Authenticator signatures which are stronger than brute-force.

However, the probmens with MD5 means that if a someone can view unencrypted RADIUS traffic, even a hobbyist attacker can crack all possible RADIUS shared secrets of eight characters or less. Such attacks can also result in compromise of all passwords carried in the User-Password attribute.

Even if a stronger packet signature method was used as in [RFC6218], it would not fully address the issues with RADIUS. Most information in RADIUS is sent in cleartext, and only a few attributes are hidden via obfuscation methods which rely on more "ad hoc" MD5 constructions. The privacy implications of this openness are severe.

Any observer of non-TLS RADIUS traffic is able to tell who is logging in to the network, what devices they are using, where they are logging in from, and their approximate location (usually city). With location-based attributes as defined in [RFC5580], a users location may be determined to within 15 or so meters. An observer can use RADIUS accounting packets to determine how long a user is online, and to track a summary of their total traffic (upload and download totals).

When RADIUS/UDP is used across the public Internet, the location of corporate executives can potentially be tracked in real-time (usually 10 minute intervals), to within 15 meters. Their devices can be identified, and tracked. Any passwords they send via the User-Password attribute can be be compromised. The negative implications for security and individual safety are obvious.

These issues are only partly mitigated when the attributes RADIUS is carrying define their own increased security and privacy. For example, some authentication methods such EAP-TLS, EAP-TTLS, etc. allow for User-Name privacy and for more secure transport of

passwords. The use of MAC address randomization can limit device informationidentification to a particular manufacterer, instead of to a unique device.

However, these authentication methods are not always used or are not always available. Even if these methods were used ubiquitously, they do not protect all of the information which is publicly available when RADIUS/UDP or RADIUS/TCP is used.

It is no longer acceptable for RADIUS to rely on MD5 for security. It is no longer acceptable to send device or location information in clear text. This document therefore deprecates insecure uses of RADIUS, and mandates the use of secure TLS-based transport layers.

The use of TLS-based transport ensures complete privacy and security for all RADIUS traffic. An observer is limited to knowing rough activity levels of a client or server. That is, an observer can tell if there are a few users on a NAS, or many users on a NAS. All other information is hidden from all observers.

1.1. Overview

The rest of this document begins a summary of issues with RADIUS, and shows just how trivial it is to crack RADIUS/UDP security. We then mandate the use of TLS transport, and describe what that means. We give recommendations on how current systems can be migrated to using TLS. We conclude with privacy and security considerations.

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 Transport Congestion Protocol [RFC6613]

*RADIUS/TLS

RADIUS over the Transport Layer Security protocol [RFC6614]

*RADIUS/DTLS

RADIUS over the Datagram Transport Layer Security protocol [RFC7360]

*TLS

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

*NAS

Network Access Server, which is a RADIUS client.

3. Overview of issues with RADIUS

There are a number of issues with RADIUS. For one, RADIUS sends most information "in the clear", with obvious privacy implications.

Further, MD5 has been broken for over a decade, as summarized in [RFC6151]. No protocol should be using MD5 for anything. Even if MD5 was not broken, computers have gotten substantially faster in the past thirty years. This speed increase makes it possible for the average hobbyist to perform brute-force attacks to crack even seemingly complex shared secrets.

We address each of these issues in detail below.

3.1. Information is sent in Clear Text

Other than a few attributes such as User-Password, all RADIUS traffic is sent "in the clear". The resulting traffic has a large number of privacy issues. We refer to [RFC6973], and specifically to Section 5 of that document for detailed discussion. RADIUS is vulnerable to all of the issues raised by [RFC6973].

There are clear privacy and security information with sending user identifiers, and user locations [RFC5580] in clear-text across the Internet. As such, the use of clear-text protocols across insecure networks is no longer acceptable.

3.2. MD5 has been broken

Attacks on MD5 are summarized in part in [RFC6151]. While there have not been many new attacks in the decade since [RFC6151] was published, that does not mean that further attacks do not exist. It is more likely that no one is looking for new attacks.

It is reasonable to expect that new research can further break MD5, but also that such research may not be publicly available.

3.3. Complexity of cracking RADIUS shared secrets

The cost of cracking a a shared secret can only go down over time as computation becomes cheaper. The issue is made worse because of the way MD5 is used in RADIUS. The attacker does not have to calculate the hash over the entire packet, as that can be precalculated, and cached. The attacker can simply begin with that precalculated portion, and brute-force only the shared secret portion.

At the time of writing this document, an "off the shelf" commodity computer can calculate at least 100M MD5 hashes per second. If we limit shared secrets to upper/lowercase letters, numbers, and a few "special" characters, we have 64 possible characters for shared secrets. Which means that for 8-character passwords, there are 2^48 possible password combinations.

The result is that using one readily available machine, it takes approximately 32 days to brute-force the entire 8 octet / 64 character password space. The problem is even worse when graphical processing units (GPUs) are used. A high-end GPU is capable of performing more than 64 billion hashes per second. (https://gist.github.com/Chick3nman/e4fcee00cb6d82874dace72106d73f). At that rate, the entire 8 character space described above can be searched in approximately 90 minutes.

This is an attack which is feasible today for a hobbyist. Increasing the size of the character set raises the cost of cracking, but not enough to be secure. Increasing the character set to 93 characters means that the hobbyist using a GPU could search the entire 8 character space in about a day.

Increasing the length of the shared secret a bit helps. For secrets ten characters long, a GPU can search a 64-character space in about six months, and a 93 character space would take approximately 24 years.

The brute-force attack is also trivially parallelizable. Nation-states have sufficient resources to deploy hundreds to thousands of systems dedicated to these attacks. That realization means that a "time to crack" of 24 years is simply expensive, but is not particularly difficult.

Whether the above numbers exactly correct, or only approximate is immaterial. These attacks will only get better over time. The cost to crack shared secrets will only go down.

Even worse, administrators do not always derive shared secrets from secure sources of random numbers. The "time to crack" numbers given above are the absolute best case, assuming administrators follow best practices for creating secure shared secrets. For shared secrets created manually by a person, the search space is orders of magnitude smaller than the best case outlined above.

It should be assumed that an average attacker with modest resource can crack most human-derived shared secrets in minutes, if not seconds.

Despite the ease of attacking MD5, it is still a common practice for some "cloud" and other RADIUS providers to send RADIUS/UDP packets over the Internet "in the clear". It is also common practice for administrators to use "short" shared secrets, and to use shared secrets created by a person, or derived from a limited character set. Theses practice are followed for ease of use of administrators, but they are also highly insecure.

3.4. Tunnel-Password and CoA-Request packets

There are similar security issues for the Tunnel-Password attribute, at least in CoA-Request and Disconnect-Request packets.

[RFC5176] Section 2.3 says:

Request Authenticator

In Request packets, the Authenticator value is a 16-octet MD5 [RFC1321] checksum, called the Request Authenticator. The Request Authenticator is calculated the same way as for an Accounting-Request, specified in [RFC2866].

Where [RFC2866] Section 3 says:

The NAS and RADIUS accounting server share a secret. The Request Authenticator field in Accounting-Request packets contains a one-way MD5 hash calculated over a stream of octets consisting of the Code + Identifier + Length + 16 zero octets + request attributes + shared secret (where + indicates concatenation). The 16 octet MD5 hash value is stored in the Authenticator field of the Accounting-Request packet.

Taken together, these defintions means that for CoA-Request packets, all attribute obfuscation is calculated with the Reply Authenticator being all zeroes.

[RFC5176] Section 3.6 allows for Tunnel-Password in CoA-Request packets:

...
Change-of-Authorization Messages

Request	ACK	NAK	#	Attribute
...				
0+	0	0	69	Tunnel-Password (Note 5)
...				

(Note 5) When included within a CoA-Request, these attributes represent an authorization change request. Where tunnel attributes are included within a successful CoA-Request, all existing tunnel attributes are removed and replaced by the new attribute(s).

However, [RFC2868] Section 3.5 says that Tunnel-Password is encrypted with the Request Authenticator:

Call the shared secret S, the pseudo-random 128-bit Request Authenticator (from the corresponding Access-Request packet) R,

The assumption that the Request Authenticator is random data is true for Access-Request packets. It is not true for CoA-Request packets

That is, when the Tunnel-Password attribute is used in CoA-Request packets, the only source of randomness in the obfuscation is the salt, as defined in [RFC2868] Section 3.5;

Salt
The Salt field is two octets in length and is used to ensure the uniqueness of the encryption key used to encrypt each instance of the Tunnel-Password attribute occurring in a given Access-Accept packet. The most significant bit (leftmost) of the Salt field MUST be set (1). The contents of each Salt field in a given Access-Accept packet MUST be unique.

Which means that there is only 15 bits of entropy in the Tunnel-Password obfuscation (plus the secret). It is not known if this limitation makes it easier to determine the contents of the Tunnel-Password. However, it cannot be a good thing, and it is one more reason to deprecate RADIUS/UDP.

4. All short Shared Secrets have been compromised

Unless RADIUS packets are sent over a secure network (IPSec, TLS, etc.), administrators should assume that any shared secret of 8 characters or less has been immediately compromised. Administrators should assume that any shared secret of 10 characters or less has been compromised by an attacker with significant resources. Administrators should also assume that any private information (such as User-Password) which depends on such shared secrets has also been compromised.

Further, if a User-Password has been sent over the Internet via RADIUS/UDP or RADIUS/TCP in the last decade, you should assume that password has been compromised by an attacker with sufficient resources.

5. Deprecating RADIUS, and mandating TLS.

The solution to an insecure protocol using thirty year-old cryptography is to deprecate the insecure cryptography, and to mandate modern cryptographic transport.

5.1. Deprecating UDP and TCP as transports

RADIUS/UDP and RADIUS/TCP MUST NOT be used outside of secure networks. A secure network is one which is known to be safe from eavesdroppers, attackers, etc.

For example, if IPSec is used between two systems, then those systems may use RADIUS/UDP or RADIUS/TCP over the IPSec connection.

Similarly, RADIUS/UDP and RADIUS/TCP may be used in secure management networks. However, administrators should not assume that such uses are secure.

Using RADIUS/UDP and RADIUS/TCP in any environment is still NOT RECOMMENDED. A network misconfiguration could result in the RADIUS traffic being sent over an insecure network. Neither the RADIUS client nor the RADIUS server would be aware of this misconfiguration.

In contrast, when TLS is used, the RADIUS endpoints are aware of all security issues, and can enforce security for themselves.

5.2. Mandating TLS transport

All RADIUS systems MUST support RADIUS/TLS and/or RADIUS/DTLS.

5.3. Crypto-Agility

The crypto-agility requirements of [RFC6421] are addressed in [RFC6614] Appendix C, and in Section 10.1 of [RFC7360]. For clarity, we repeat the text of [RFC7360] here, with some minor modifications to update references, but not content.

Section 4.2 of [RFC6421] makes a number of recommendations about security properties of new RADIUS proposals. All of those recommendations are satisfied by using TLS or DTLS as the transport layer.

Section 4.3 of [RFC6421] makes a number of recommendations about backwards compatibility with RADIUS. [RFC7360] Section 3 addresses these concerns in detail.

Section 4.4 of [RFC6421] recommends that change control be ceded to the IETF, and that interoperability is possible. Both requirements are satisfied.

Section 4.5 of [RFC6421] requires that the new security methods apply to all packet types. This requirement is satisfied by allowing TLS and DTLS to be used for all RADIUS traffic. In addition, [RFC7360] Section 3, addresses concerns about documenting the transition from legacy RADIUS to crypto-agile RADIUS.

Section 4.6 of [RFC6421] requires automated key management. This requirement is satisfied by using TLS or DTLS key management.

We can now finalize the work began in [RFC6421]. We now state that new RADIUS specifications MUST NOT create any new cryptographic primitives to sign individual packets, or to obfscate the contents of any attributes. All security and privacy MUST instead be provided by a secure transport layer such as TLS. Simply using IPSec is not enough, as the use (or not) of IPSec is unknown to the RADIUS application.

It is NOT RECOMMENDED to define new attributes which use the content obfuscation methods defined for User-Password or Tunnel-Password. We would like to forbid such constructs entirely. We recognize that RADIUS/UDP will still be in use for many years, and that new standards may require some modicrum of privacy. As a result, it is difficult to forbid the use of these constructs.

hat being said, there has been no need since [RFC2868] in 2000 for new attribute which use these obfuscation methods. We believe therefore that there will be no demand for this kind of new attribute.

6. Migration Path and Recommendations

We recognize that it is difficult to upgrade legacy devices with new cryptographic protocols and user interfaces. The problem is made worse because the volume of RADIUS devices which are in use. The exact number is unknown, and can only be approximated. Our best guesses would be in the order of hundreds of thousands, if not millions of RADIUS/UDP devices are in daily use.

We therefore need to define a migration path to using secure transports. We give a few migration steps by making stronger recommendations for shared secrets. Where [RFC6614] Section 2.3 makes support for TLS-PSK optional, we now require support for TLS-PSK.

6.1. Shared Secrets

[RFC2865] Section 3 says:

> It is preferred that the secret be at least 16 octets. This is to ensure a sufficiently large range for the secret to provide protection against exhaustive search attacks. The secret MUST NOT be empty (length 0) since this would allow packets to be trivially forged.

This recommendation is no longer adequate, so we strengthen it here.

RADIUS implementations MUST support shared secrets of at least 32 octets, and SHOULD support shared secrets of 64 octets. Implementations MUST warn administrators that the shared secret is insecure if it is 10 octets or less in length.

Administrators SHOULD use shared secrets of at least 24 octets, generated using a source of secure random numbers. Any other

practice is likely to lead to compromise of the shared secret, user information, and possibly of the entire network.

Creating secure shared secrets is not difficult. One solution is to use a simple script given below.

```
#!/usr/bin/env perl use MIME::Base32; use Crypt::URandom(); print
join('-', unpack("(A4)*", lc
encode_base32(Crypt::URandom::urandom(12)))), "\n";
```

This script reads 96 bits of random data from a secure source, encodes it in Base32, and then makes it easier for people to work with. The generated secrets are of the form "2nw2-4cfi-nicw-3g2i-5vxq". This form of secret will be accepted by any known implementation which supports at least 24 octets for shared secrets.

Given the simplicity of creating strong secrets, there is no excuse for using weak shared secrets with RADIUS.

RADIUS implementors SHOULD provide tools for administrators which automatically create secure shared secrets.

Given the insecurity of RADIUS, the absolute minimum acceptable security is to use strong shared secrets. However, TLS-PSK is not substantially more complex, and offers significantly increased security and privacy.

6.2. TLS-PSK

TLS uses certificates in most common uses. However, we recognize that it may be difficult to fully upgrade client implementations to allow for certificates to be used with RADIUS/TLS and RADIUS/DTLS. It is therefore RECOMMENDED that client implementations allow the use of a pre-shared key (TLS-PSK). The client implementation can then expose a flag "TLS yes / no", and then a shared secret (now PSK) entry field.

Any shared secret used for RADIUS/UDP MUST NOT be used for TLS-PSK.

Implementations MUST support PSKs of at least 32 octets, and SHOULD support PSKs of 64 octets. Implementations MUST require that PSKs be at least 16 octets in length. That is, short PSKs MUST NOT be permitted to be used.

Administrators SHOULD use PSKs of at least 24 octets, generated using a source of secure random numbers. The script given above can again be used.

We also incorporate by reference the requirements of Section 10.2 of [RFC7360] when using PSKs.

6.2.1. PSK Identities

[RFC6614] is silent on the subject of PSK identities, which is an issue that we correct here.

The need to manage identities associated with PSK is a new requirement for NAS management interfaces, and is a new requirement

for RADIUS servers. As such, implementors and operators require guidance.

RADIUS systems implementing TLS-PSK MUST support identities as per [RFC4279] Section 5.3, and MUST enable configuring TLS-PSK identities in management interfaces as per [RFC4279] Section 5.4.

A RADIUS client implementing TLS-PSK MUST update their management interfaces and application programming interfaces (APIs) to label the PSK field as "PSK" or "TLS-PKS, and MUST NOT label the PSK field as "shared secret".

Where dynamic server lookups [RFC7585] are not used, RADIUS clients MUST still permit the configuration of a RADIUS server IP address.

When a RADIUS server implements TLS-PSK, the PSK identity replaces the IP address as the logical identifier for a RADIUS client. RADIUS servers MUST be able to look up PSK identity in a subsystem which then returns the actual PSK.

We note that due to NAT, etc. there may be multiple RADIUS clients using one public IP address. RADIUS servers MUST support such a configuration, and MUST support unique PSK identities for each client which is deployed in such a scenario.

RADIUS servers SHOULD tie PSK identities to a particular permitted IP address or permitted network, as doing so will lower the risk if a PSK is leaked. RADIUS servers MUST permit multiple clients to share one permitted IP address or network.

A RADIUS server which accepts TLS-PSK MUST support a unique PSK identifier per RADIUS client. There is no reason to use the same identifier for multiple clients. A RADIUS server which accepts TLS-PSK MUST have a unique PSK per RADIUS client.

It is RECOMMENDED that RADIUS clients and server track all used shared secrets and PSKs, and then verify that the following requirements all hold true:

*no shared secret is used for more than one RADIUS client

*no PSK is used for more than one RADIUS cleint

*no shared secret is used as a PSK

*no PSK is used as a shared secret

7. Privacy Considerations

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

Deprecating insecure transport for RADIUS, and requiring secure transport means that personally identifying information is no longer sent "in the clear". As noted earlier in this document, such information can include MAC addresses, user identifiers, and user locations.

8. Security Considerations

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

Deprecating insecure transport for RADIUS, and requiring secure transport means that historical and/or future security issues with the RADIUS protocol no longer apply.

Experience has shown that it can be difficult to configure and update certificates in a RADIUS environment. Public Certification Authorities (CAs) will not issue certificates specifically for use by RADIUS servers. As for updates, certificates may be valid for many years. By the time a certificate is up for renewal, the people and processes responsible for it may have changed. Which means that updating certificates can be a complex and error-prone task.

Therefore while [RFC6614] suggested that TLS-PSK was optional, we mandate support for TLS-PSK here. We also correct the omission of any discussion related to TLS-PSK identies in that specification, and explain how those identities are used on clients and servers.

9. IANA Considerations

There are no IANA considerations in this document.

RFC Editor: This section may be removed before final publication.

10. Acknowledgements

TBD.

11. Changelog

12. References

12.1. 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>.

[RFC4279] Eronen, P., Ed. and H. Tschofenig, Ed., "Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)", RFC

4279, DOI 10.17487/RFC4279, December 2005, <https://www.rfc-editor.org/info/rfc4279>.

[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>.

[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>.

12.2. 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>.

[RFC2866] Rigney, C., "RADIUS Accounting", RFC 2866, DOI 10.17487/RFC2866, June 2000, <https://www.rfc-editor.org/info/rfc2866>.

[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>.

[RFC5580] Tschofenig, H., Ed., Adrangi, F., Jones, M., Lior, A., and B. Aboba, "Carrying Location Objects in RADIUS and Diameter", RFC 5580, DOI 10.17487/RFC5580, August 2009, <https://www.rfc-editor.org/info/rfc5580>.

[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>.

[RFC6973] Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., and R. Smith, "Privacy Considerations for Internet Protocols", RFC 6973, DOI 10.17487/RFC6973, July 2013, <https://www.rfc-editor.org/info/rfc6973>.

[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>.

[RFC7585] Winter, S. and M. McCauley, "Dynamic Peer Discovery for RADIUS/TLS and RADIUS/DTLS Based on the Network Access Identifier (NAI)", RFC 7585, DOI 10.17487/RFC7585, October 2015, <https://www.rfc-editor.org/info/rfc7585>.

Author's Address

Alan DeKok
FreeRADIUS

Email: aland@freeradius.org