RADIUS EXTensions Working Group   Sanal Kumar Kariyezhath Sivaraman
INTERNET-DRAFT   Aravind Prasad Sridharan
Intended Status: Standards Track   DELL
Expires: August 21, 2017   February 17, 2017

RADIUS Attribute Security
draft-aravind-radext-attribute-security-00

Abstract

This document specifies a simple method to provide security to RADIUS message attribute values.

Status of this Memo

Copyright and License Notice

Table of Contents

1 Introduction

The RADIUS protocol [RFC2865] is a widely deployed authentication and authorization protocol. The supplementary RADIUS Accounting specification [RFC2866] provides accounting mechanisms, thus delivering a complete authentication, Authorization, and Accounting (AAA) solution. However, the major drawback is the lack of security for the message contents such as sensitive attributes.

Although RADIUS over TLS addresses this issue, it involves significant cost and PKI deployment hassles. This draft proposal provides a mechanism to secure RADIUS message without any major change in the existing RADIUS server deployments.

Here the proposal is to encrypt the attribute value with a key using a symmetric cipher. To have less change in the existing deployment and to have a simplified key management, this proposal leverages the shared secret as one of the factor in making the key that is required for encryption.

1.1 Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2 Attribute Security in RADIUS messages

Here the proposal is to encrypt the attribute value with a key using a symmetric cipher at the sender and decrypt with the same key at the receiver. Key is generated dynamically for each message. Generate key by hashing the shared secret, RADIUS identifier and Authenticator using the hashing algorithm. Authenticator represents the Request Authenticator in the Request message and the Response Authenticator in the Response Message.

Length of key depends on the hashing algorithm.

Key k = Hash(shared secret, RADIUS Identifier, Authenticator)

A new attribute (SEC-Message) is introduced to indicate that the attribute values are secured and to propose the hashing algorithm and cipher for the secure communication. NIST supported hashing algorithms such as SHA and ciphers such as AES, are recommended. Selection of hashing and ciphers for the encryption at the sender, can be based on the configuration, which is implementation specific. Enabling of the attribute security at the sender also based on configuration, which is implementation specific.

Decryption is done based on the algorithms that are part of the SEC-Message attribute in the received RADIUS message. If the recipient doesn't support attribute security feature, then that would result in a failure indirectly as the encrypted attribute value cannot be recognized by the recipient. This attribute is to provide the flexibility in selecting the algorithms based on capability.

Encrypted attribute value V = Cipher(v, k) where v is the attribute value in plain text and k is the dynamically generated key using the proposed hash algorithm.

In a roaming scenario, each proxy needs to decrypt the attributes on receiving the message and encrypt the same again while sending the message. Recipient does the decryption only if the SEC-Message attribute is present in the message.

It is possible for a proxy to use different hashing algorithm or cipher while receiving and sending. It is possible for a proxy to receive a message with SEC-Message attribute and forward the decrypted message without encryption.

2.1 SEC-Message Attribute

This attribute indicates to the receiver that the message is encrypted. This Attribute consists of 2 sub-attributes to represent hashing algorithm and cipher. This attribute is applicable in all the RADIUS messages.

```
SEC-Message Attribute
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |  Length       |   Sub-Attribute(s)...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type

TBD

Length

1

Sub-Attributes

This includes the TLVs indicating the Hashing algorithm and cipher.

```
Sub-Attribute 1 - Hashing Algorithm
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |  Length       |   String ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type

TBD

Length

1

String

This indicates the hashing algorithm to be used. NIST supported algorithms are recommended. For example, sha-256.

Sub-Attribute 2 - Cipher

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |  Length       |   String ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type

TBD

Length

1

String

This indicates the cipher to be used. NIST supported algorithms are recommended. For example, aes256-cbc.

3 Example Message Flow with Sample data

3.1 Client

```
Shared secret          - "godofsmallthings"
Request Authenticator - "ecfe3d2fe4473ec6299095ee46aedf77"
RADIUS Identifier     - 70
User-Password         - "edada28173cb372896832ac78522b5c6"
Hashing Algorithm     - sha256         (config)
Cipher                - aes256-cbc     (config)
Attribute_Sec_Enabled - TRUE           (config)
```

Client does the following to send the RADIUS request message if Attribute_Sec_Enabled is TRUE.

1. Add SEC-Message attribute in the RADIUS message with sub-attributes as sha-256 and aes256-cbc

2. Generate key for encryption

```
      Key k  = Hash(shared secret, RADIUS Identifier,
                    Authenticator)

             = sha256("godofsmallthings", 70,
                      "ecfe3d2fe4473ec6299095ee46aedf77")

             = "88dd551af0fd16d33463cb7392d125edfea0683517e7ece2
                682afd629a048b20"
```

3. Encrypt the attribute (say, User-Password attribute)

```
      Encrypted attribute value V = Cipher(User-Password
                                           attribute value, key)

                                  = aes256-cbc("edada28173cb3728
                                               96832ac78522b5c6",
                                              "88dd551af0fd16d334
                                               63cb7392d125edfea0
                                               683517e7ece2682afd
                                               629a048b20")
                                 = "a6638bbae25cc5e627e9aa9c47e651
                                    d023251443381a5d77"
```

4. Add attribute (say, User-Password attribute) in the RADIUS message with the encrypted value.

3.2 Server

Shared secret	-	"godofsmallthings"	
Request Authenticator	-	"ecfe3d2fe4473ec6299095ee46aedf77"	
Response Authenticator	-	"f050649184625d36f14c9075b7a48b83"	
RADIUS Identifier	-	70	
Hashing Algorithm	-	sha128	(config)
Cipher	-	3des-cbc	(config)
Attribute_Sec_Enabled	-	TRUE	(config)

Server does the following upon receiving the RADIUS request Message if Attribute_Sec_Enabled is TRUE.

1. Check whether SEC-Message attribute exists to see whether the attribute values are encrypted. Get the hashing algorithm and cipher.

   ```
   Hashing Algorithm in the attribute - sha256
   Cipher in the attribute            - aes256-cbc
   ```

2. Generate key for decryption

   ```
   Key k  = Hash(shared secret, RADIUS Identifier,
                 Authenticator)

          = sha256("godofsmallthings", 70,
                     "ecfe3d2fe4473ec6299095ee46aedf77")

         = "88dd551af0fd16d33463cb7392d125edfea0683517
            e7ece2682afd629a048b20"
   ```

3. Decrypt the attribute (say, User-Password attribute)

   ```
   Decrypted attribute value v = Cipher(User-Password attribute
                                        encrypted value, key)

                               = aes256-cbc("a6638bbae25cc5e
                                             627e9aa9c47e651
                                             d023251443381a5
                                             d77",
                                            "88dd551af0fd16d
                                             33463cb7392d125
                                             edfea0683517e7e
                                             ce2682afd629a04
                                             8b20")


                              = "edada28173cb372896832ac7852
                                 2b5c6"
   ```

Server does the encryption procedure while sending the RADIUS response message if Attribute_Sec_Enabled is TRUE.

4 Recommendations

1. Keep the shared secret lengthy and complex as this is one of the main factor to decide the key. This can potentially save from brute force attacks.

2. Use the NIST recommended hashing algorithms and ciphers.

5 Advantages

1. Existing deployments can easily adapt with minimal configuration to ensure security.

2. Compared to TLS, this proposal ensures hop to hop security with less cost and maintenance overhead.

6 Security Considerations

This document does not introduce any new security concerns to RADIUS or any other specifications referenced in this document

7 IANA Considerations

This document requests IANA to allocate the new type code value to the proposed Security Attribute and add it to the list of existing RADIUS Attributes.

8 References

8.1 Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

8.2 Informative References

[RFC2865] Rigney, C., Willens, S., Rubens, A., and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, June 2000.

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

Authors' Addresses

Sanal Kumar Kariyezhath Sivaraman
DELL
Olympia Technology Park
Guindy, Chennai 600032
India
Phone: +91 9600081365
Email: Sanal_Kumar_Sivarama@dell.com

Aravind Prasad Sridharan
DELL
Olympia Technology Park
Guindy, Chennai 600032
India
Phone: +91 9884612715
Email: aravind_sridharan@dell.com