

# Authenticator RADIUS y guía inválidos del Troubleshooting del Mensaje-authenticator

## Contenido

[Introducción](#)

[Encabezado del authenticator](#)

[Autenticación de la respuesta](#)

[¿Cuándo debe usted contar con el error de la validación?](#)

[Ocultación de contraseñas](#)

[Retransmisiones](#)

[Contabilidad](#)

[Atributo del Mensaje-authenticator](#)

[¿Cuándo debe el Mensaje-authenticator ser utilizado?](#)

[¿Cuándo debe usted contar con el error de la validación?](#)

[Valide el atributo del Mensaje-authenticator](#)

[Información Relacionada](#)

## Introducción

Este documento describe dos mecanismos de seguridad RADIUS:

- Encabezado del authenticator
- Atributo del Mensaje-authenticator

Este documentos abarca cuáles estos mecanismos de seguridad son, cómo se utilizan, y cuando usted debe contar con el error de la validación.

## Encabezado del authenticator

Por el RFC 2865, la encabezado del authenticator es 16 bytes de largo. Cuando se utiliza en un pedido de acceso, se llama un authenticator de la petición. Cuando se utiliza en cualquier clase de respuesta, se llama un authenticator de la respuesta. Se utiliza para:

- Autenticación de la respuesta
- Ocultación de contraseñas

## Autenticación de la respuesta

Si el servidor responde con el authenticator correcto de la respuesta, el cliente puede computar si esa respuesta fue relacionada con una petición válida.

El cliente envía la petición con la encabezado al azar del authenticator. Entonces, el servidor que envía la respuesta calcula el authenticator de la respuesta con el uso del paquete de pedidos junto con el secreto compartido:

```
ResponseAuth = MD5(Code + ID + Length + RequestAuth + Attributes + Secret)
```

El cliente que recibe la respuesta realiza la misma operación. Si el resultado es lo mismo, el paquete está correcto.

Nota: El atacante que conoce el valor secreto no puede spoof la respuesta a menos que pueda oler la petición.

## ¿Cuándo debe usted contar con el error de la validación?

El error de la validación ocurre si el Switch no oculta la petición más (por ejemplo, debido al descanso). Usted puede ser que también lo experimente cuando el secreto compartido es inválido (sí - el Access-Reject también incluye esta encabezado). Esta manera, el dispositivo de acceso a la red (NAD) puede detectar la discordancia del secreto compartido. Es señalada generalmente por los clientes/los servidores del Authentication, Authorization, and Accounting (AAA) como discordancia de clave compartida, pero no revela los detalles.

## Ocultación de contraseñas

La encabezado del authenticator también se utiliza para evitar enviar el atributo de la Contraseña del Usuario en el sólo texto. Primero la publicación de mensaje 5 (MD5 - secreto, authenticator) se computa. Entonces varias operaciones XOR con los pedazos de la contraseña se ejecutan. Este método es susceptible para los ataques offline (tablas del arco iris) porque el MD5 no se percibe como algoritmo unidireccional fuerte más.

Aquí está el script de Python que computa la Contraseña del Usuario:

```
def Encrypt_Pass(password, authenticator, secret):
    m = md5()
    m.update(secret+authenticator)
    return "".join(chr(ord(x) ^ ord(y)) for x, y in zip(password.ljust(
16, '\0')[:16], m.digest()[:16]))
```

## Retransmisiones

Si los atributos uces de los en el pedido de acceso RADIUS han cambiado (como el RADIUS ID, username, y así sucesivamente), el nuevo campo del authenticator debe ser generado y el resto de los campos que dependen de él deben ser recalculados. Si esto es una retransmisión, nada debe cambiar.

## Contabilidad

El significado de la encabezado del authenticator es diferente para un pedido de acceso y una Estadística-petición.

Para un pedido de acceso, el authenticator se genera aleatoriamente y se espera que reciba una respuesta con el ResponseAuthenticator calculado correctamente, que prueba que la respuesta fue relacionada con esa petición específica.

Para una Estadística-petición, el authenticator no es al azar, sino que se calcula (según el RFC 2866):

```
RequestAuth = MD5(Code + ID + Length + 16 zero octets + Attributes + Secret)
```

Esta manera, el servidor puede marcar el mensaje de las estadísticas inmediatamente y caer el paquete si el valor recalculado no hace juego el valor del authenticator. Las devoluciones del Identity Services Engine (ISE):

```
11038 RADIUS Accounting-Request header contains invalid Authenticator field
```

La razón típica para esto es la clave secreta compartida incorrecta.

## Atributo del Mensaje-authenticator

El atributo del Mensaje-authenticator es el atributo de RADIUS definido en el RFC 3579. Se utiliza para un propósito similar: para firmar y validar. Pero este vez, no se utiliza para validar una respuesta sino una petición.

El cliente que envía un pedido de acceso (él puede también ser un servidor que responde con un Acceso-desafío) computa el Hash-Based Message Authentication Code (HMAC)-MD5 de su propio paquete, y entonces agrega el atributo del Mensaje-authenticator como firma. Entonces, el servidor puede verificarla realiza la misma operación.

La fórmula parece similar a la encabezado del authenticator:

```
Message-Authenticator = HMAC-MD5 (Type, Identifier, Length, Request Authenticator, Attributes)
```

La función HMAC-MD5 admite dos argumentos:

- El payload del paquete, que incluye el campo del Mensaje-authenticator de 16 bytes llenó de los ceros
- El secreto compartido

## ¿Cuándo debe el Mensaje-authenticator ser utilizado?

El Mensaje-authenticator SE DEBE utilizar para cada paquete, que incluye el mensaje del Protocolo de Autenticación Extensible (EAP) (RFC 3579). Esto incluye al cliente que envía el pedido de acceso y el servidor que responde con el Acceso-desafío. El otro lado debe caer silenciosamente el paquete si la validación falla.

## ¿Cuándo debe usted contar con el error de la validación?

El error de la validación ocurrirá cuando el secreto compartido es inválido. Entonces, el servidor de AAA no puede validar la petición.

Los informes ISE:

11036 The Message-Authenticator Radius Attribute is invalid.

Esto ocurre generalmente en la etapa posterior cuando se asocia el mensaje EAP. El primer paquete RADIUS de la sesión del 802.1x no incluye el mensaje EAP; no hay campo del Mensaje-authenticator y no es posible verificar la petición, pero en esa etapa, el cliente puede validar la respuesta con el uso del campo del authenticator.

## Valide el atributo del Mensaje-authenticator

Aquí está un ejemplo para ilustrar cómo usted cuenta manualmente el valor para asegurarse lo se computa correctamente.

Se ha elegido el paquete número 30 (pedido de acceso). Está en el medio de la sesión EAP, y el paquete incluye el campo del Mensaje-authenticator. El objetivo es verificar que el Mensaje-authenticator está correcto:

```
30 2012-12-20 07:34:19.221908 192.168.10.10 192.168.10.150 RADIUS 401 Access-Request(1)
|
+ Radius Protocol
  Code: Access-Request (1)
  Packet identifier: 0x16 (22)
  Length: 359
  Authenticator: bed95259578302c0f9184df62b859d6b
  [The response to this request is in frame 31]
+ Attribute Value Pairs
  + AVP: l=7 t=User-Name(1): cisco
  + AVP: l=6 t=Service-Type(6): Framed(2)
  + AVP: l=6 t=Framed-MTU(12): 1500
  + AVP: l=19 t=Called-Station-Id(30): AA-BB-CC-00-64-00
  + AVP: l=19 t=Calling-Station-Id(31): 08-00-27-6E-C5-50
  + AVP: l=202 t=EAP-Message(79) Last Segment[1]
  + AVP: l=18 t=Message-Authenticator(80): 01418d3b1865556918269d3c f73608b0
```

1. Haga clic con el botón derecho del ratón el **protocolo RADIUS** y elija los **bytes del paquete seleccionados exportación**.
2. Escriba ese payload RADIUS a un archivo (datos binarios).
3. Para computar el campo del Mensaje-authenticator, usted debe poner los ceros allí y computar el HMAC-MD5.

Por ejemplo, cuando usted utiliza el maleficio/el Editor binario, tal como vim, después de que usted teclee “: ¡%! el xxd”, que conmuta para embrujar el modo y los ceros 16 bytes que comienzan después de que el "5012" (50hex es 80 en diciembre que es tipo del Mensaje-authenticator, y 12 es el tamaño que es 18 incluyendo la encabezado de los pares de valores de atributos (AVP)):

```

0000000: 0116 0167 bed9 5259 5783 02c0 f918 4df6 ...g..RYW.....M.
0000010: 2b85 9d6b 0107 6369 7363 6f06 0600 0000 +..k..cisco.....
0000020: 020c 0600 0005 dc1e 1341 412d 4242 2d43 .....AA-BB-C
0000030: 432d 3030 2d36 342d 3030 1f13 3038 2d30 C-00-64-00..08-0
0000040: 302d 3237 2d36 452d 4335 2d35 304f ca02 0-27-6E-C5-500..
0000050: 4100 c819 8000 0000 be16 0301 0086 1000 A.....
0000060: 0082 0080 880d 0fe6 8421 562e bcf3 75a7 .....!V...u.
0000070: fbf4 9c20 e114 a19d 1282 96d7 45b8 9c26 ... ..E..&
0000080: 86c5 9935 1b2c ca98 1b60 5e91 1c63 d123 ...5.,...^..c.#
0000090: f019 1ab6 7e2d 0497 1e02 0768 0ac3 aa84 ....~.....h...
00000a0: 80d5 cd14 92a9 ae31 e9e2 121e 28e8 5f21 .....1....(._!
00000b0: 5c1a 4e20 013f a55b 7b1d 0eb7 1d17 a565 \.N .?.[f.....e
00000c0: 626b 2bb4 f756 da05 b51b 043b 346a c51f bk+..V.....;4j..
00000d0: 98a7 007e ed55 e24b 1cab ec06 799b aed5 ...~.U.K....y...
00000e0: 72c5 451b 1403 0100 0101 1603 0100 28e2 r.E.....(
00000f0: d25f 2deb 0f0c baf5 570d d3f6 05df 6534 ._-.....W.....e4
0000100: 48d8 0853 00ae 3230 73a9 afb7 ac87 d834 H..S..20s.....4
0000110: f7e9 bb57 8ac1 1750 1200 0000 0000 0000 ...W...P.....
0000120: 0000 0000 0000 0000 003d 0600 0000 0f05 .....=.....
0000130: 0600 00c3 5057 0d45 7468 6572 6e65 7430 ...PW.Ethernet0
0000140: 2f30 181f 3236 5365 7373 696f 6e49 443d /0..26SessionID=
0000150: 6163 732f 3134 3531 3136 3739 372f 3132 acs/145116797/12
0000160: 3b04 06c0 a80a 0a ;.....

```

Después esa modificación, el payload está lista. Es necesario volver de nuevo al maleficio/al modo binario (tipo: “: ¡%! el xxd - r”) y salva el archivo (“: wq”).

#### 4. Utilice el OpenSSL para computar el HMAC-MD5:

```

pluton # cat packet30-clear-msgauth.bin | openssl dgst -md5 -hmac 'cisco'
(stdin)= 01418d3b1865556918269d3cf73608b0

```

La función HMAD-MD5 toma dos argumentos: primer de la entrada estándar (stdin) es el mensaje sí mismo y segundo es el secreto compartido (Cisco en este ejemplo). El resultado es exactamente el mismo valor que el Mensaje-authenticator asociado al paquete access-request RADIUS.

Lo mismo se pueden computar con el uso del script de Python:

```

pluton # cat hmac.py
#!/usr/bin/env python

import base64
import hmac
import hashlib

f = open('packet30-clear-msgauth.bin', 'rb')
try:
    body = f.read()
finally:
    f.close()

digest = hmac.new('cisco', body, hashlib.md5)
d=digest.hexdigest()
print d

pluton # python hmac.py

```

01418d3b1865556918269d3cf73608b0

El ejemplo anterior presenta cómo calcular el campo del Mensaje-authenticator del pedido de acceso. Para el Acceso-desafío, el access-accept, y el Access-Reject, la lógica es exactamente lo mismo, pero es importante recordar que el authenticator de la petición debe ser utilizado, que se proporciona en el paquete access-request anterior.

## Información Relacionada

- [RFC 2865](#)
- [RFC 2866](#)
- [RFC 3579](#)
- [Soporte Técnico y Documentación - Cisco Systems](#)