L'authentificateur non valide et le Message-authentificateur de RAYON dépannent le guide

Contenu

Introduction

Ce document décrit deux mécanismes de sécurité de RAYON :

- En-tête d'authentificateur
- Attribut de Message-authentificateur

Ce document couvre ce que sont ces mécanismes de sécurité, comment ils sont utilisés, et quand vous devriez s'attendre à la panne de validation.

En-tête d'authentificateur

Par RFC 2865, l'en-tête d'authentificateur est de 16 octets de long. Quand il est utilisé dans une Access-demande, ce s'appelle un authentificateur de demande. Quand il est utilisé dans n'importe quel genre de réponse, ce s'appelle un authentificateur de réponse. Il est utilisé pour :

- Authentification de réponse
- Masquer de mot de passe

Authentification de réponse

Si le serveur répond avec l'authentificateur correct de réponse, le client peut calculer si cette réponse était liée à une demande valide.

Le client envoie la demande avec l'en-tête aléatoire d'authentificateur. Puis, le serveur qui envoie la réponse calcule l'authentificateur de réponse avec l'utilisation du paquet de demandes avec le secret partagé :

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

Le client qui reçoit la réponse exécute la même exécution. Si le résultat est identique, le paquet est correct.

> Remarque: L'attaquant qui connaît la valeur secrète ne peut pas charrier la réponse à moins qu'il puisse renifler la demande.

Quand devriez-vous attendre la panne de validation ?

La panne de validation se produit si le commutateur ne cache plus la demande (par exemple, en raison du délai d'attente). Vous pourriez également l'éprouver quand le secret partagé est non valide (oui - l'Access-anomalie inclut également cette en-tête). De cette façon, le périphérique d'accès au réseau (NAD) peut détecter la non-concordance secrète partagée. Habituellement il est signalé par des clients/serveurs d'Authentification, autorisation et comptabilité (AAA) car une non-concordance principale partagée, mais lui n'indique pas les détails.

Masquer de mot de passe

L'en-tête d'authentificateur est également utilisée afin d'éviter d'envoyer l'attribut de mot de passe utilisateur en texte brut. D'abord le Message Digest 5 (MD5 - secret, authentificateur) est calculé. Alors plusieurs exécutions XOR avec les blocs du mot de passe sont exécutées. Cette méthode est susceptible pour des attaques hors ligne (tables d'arc-en-ciel) parce que le MD5 n'est plus perçu comme algorithme à sens unique fort.

Voici le script de python qui calcule le mot de passe utilisateur :

```
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]))
```

Retransmissions

Si les attributs l'uns des dans l'Access-demande de RAYON ont changé (comme l'ID de RAYON, username, et ainsi de suite), le nouveau champ d'authentificateur devrait être généré et tous autres champs qui dépendent de lui devraient recomputed. Si c'est une retransmission, rien ne devrait changer.

Comptabilité

La signification de l'en-tête d'authentificateur est différente pour une Access-demande et une Comptabilité-demande.

Pour une Access-demande, l'authentificateur est généré aléatoirement et on s'attend à ce qu'il reçoive une réponse avec le ResponseAuthenticator calculé correctement, qui montre que la réponse a été liée à cette demande spécifique.

Pour une Comptabilité-demande, l'authentificateur n'est pas aléatoire, mais on le calcule (selon RFC 2866) :

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

De cette façon, le serveur peut vérifier le message de comptabilité immédiatement et relâcher le paquet si la valeur recalculée n'apparie pas la valeur d'authentificateur. Les retours du Cisco Identity Services Engine (ISE) :

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

La raison typique pour ceci est la clé secrète partagée incorrecte.

Attribut de Message-authentificateur

L'attribut de Message-authentificateur est l'attribut RADIUS défini dans RFC 3579. Il est utilisé pour un but semblable : pour signer et valider. Mais cette fois, il n'est pas utilisé afin de valider une réponse mais une demande.

Le client qui envoie une Access-demande (il peut également être un serveur qui répond avec un Access-défi) calcule le code d'authentification de message d'information Information (HMAC)-MD5 de son propre paquet, et ajoute alors l'attribut de Message-authentificateur comme signature. Puis, le serveur peut le vérifier exécute la même exécution.

La formule semble semblable à l'en-tête d'authentificateur :

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

La fonction HMAC-MD5 rentre deux arguments :

- La charge utile du paquet, qui inclut le champ de Message-authentificateur de 16 octets a rempli de zéros
- Le secret partagé

Quand le Message-authentificateur devrait-il être utilisé ?

Le Message-authentificateur DOIT être utilisé pour chaque paquet, qui inclut le message de Protocole EAP (Extensible Authentication Protocol) (RFC 3579). Ceci inclut le client qui envoie l'Access-demande et le serveur qui répond avec l'Access-défi. L'autre côté devrait silencieusement relâcher le paquet si la validation échoue.

Quand devriez-vous attendre la panne de validation ?

La panne de validation se produira quand le secret partagé est non valide. Puis, le serveur d'AAA ne peut pas valider la demande.

Les états ISE :

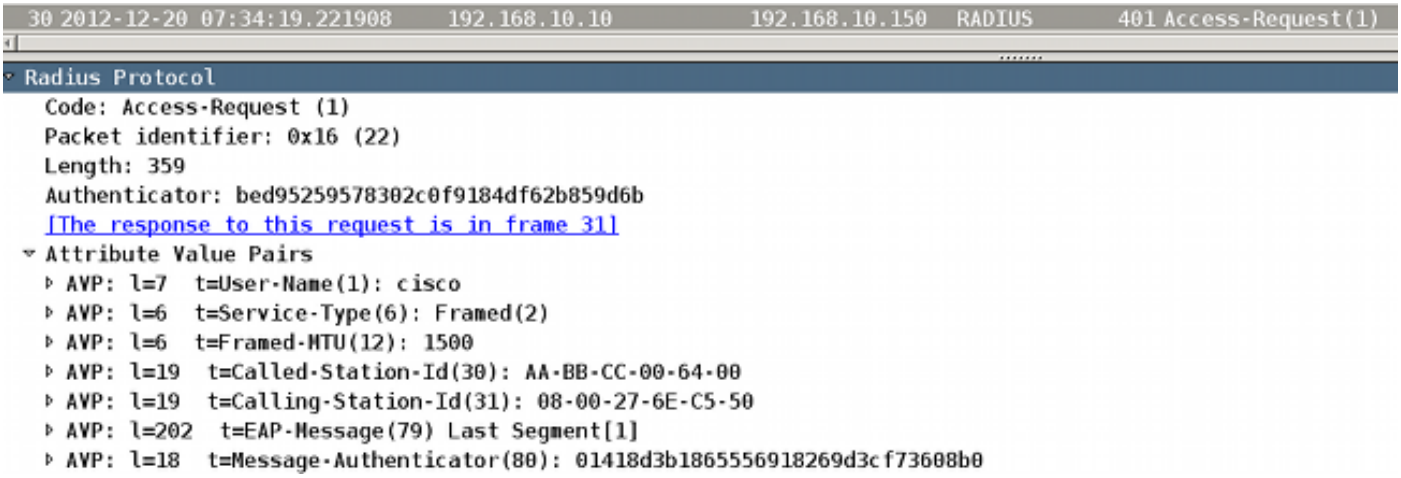
`11036 The Message-Authenticator Radius Attribute is invalid.`

Ceci se produit habituellement au stade avancé quand le message d'EAP est relié. Le premier paquet RADIUS de la session de 802.1x n'inclut pas le message d'EAP ; il n'y a aucun champ de Message-authentificateur et il n'est pas possible de vérifier la demande, mais à cette étape, le client peut valider la réponse avec l'utilisation du champ d'authentificateur.

Validez l'attribut de Message-authentificateur

Voici un exemple pour illustrer comment vous comptez manuellement la valeur afin de s'assurer qu'elle est calculée correctement.

Le paquet le numéro 30 (Access-demande) a été choisi. Il est au milieu de la session d'EAP, et le paquet inclut le champ de Message-authentificateur. Le but est de vérifier que le Message-authentificateur est correct :

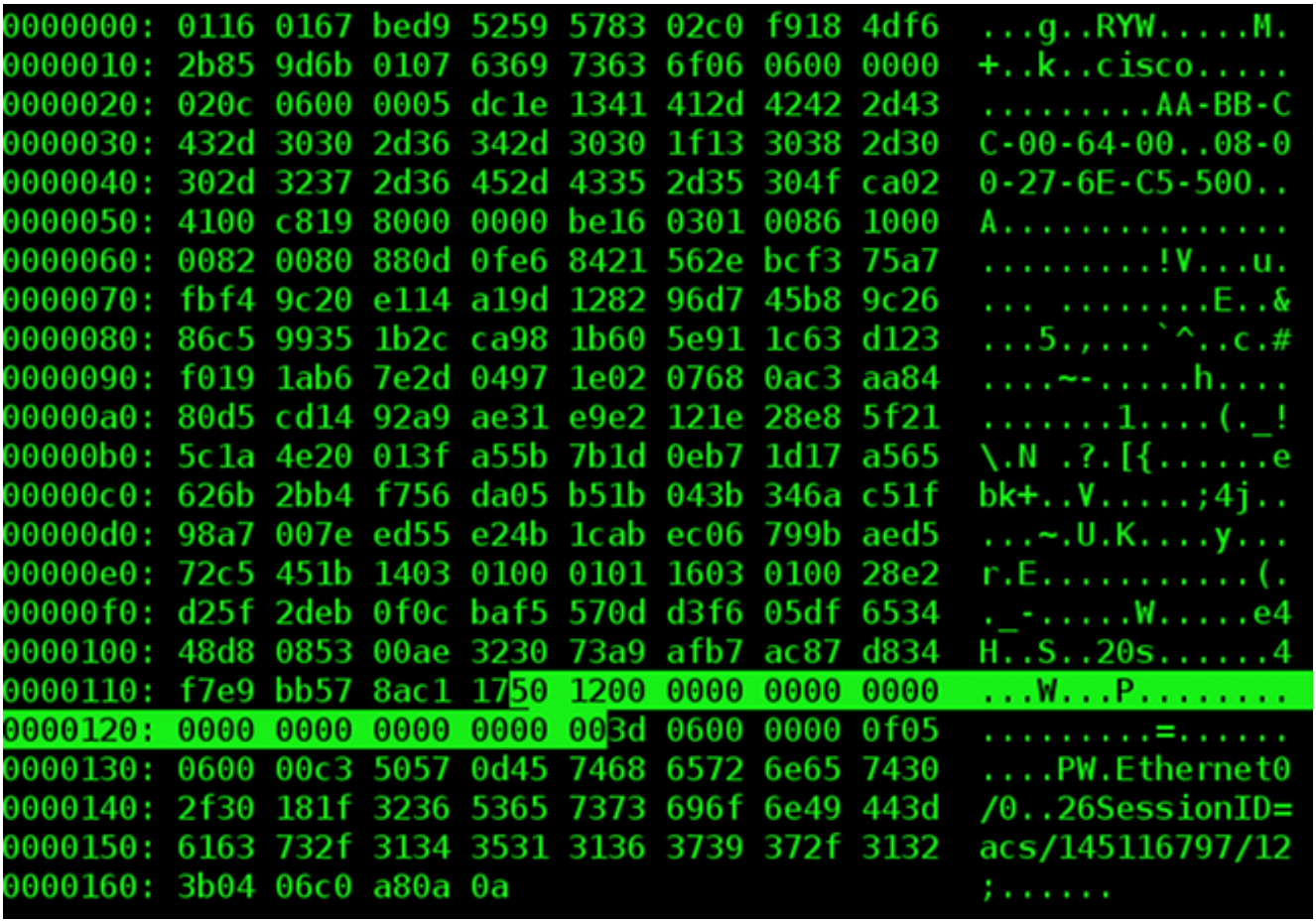


1. Cliquez avec le bouton droit le **protocole RADIUS** et choisissez les **octets de paquet sélectionnés par exportation**.

2. Écrivez cette charge utile de RAYON à un fichier (données binaires).

3. Afin de calculer le champ de Message-authentificateur, vous devez mettre des zéros là et calculer le HMAC-MD5.

   Par exemple, quand vous utilisez éditeur hexadécimal/binaire, tel que le score, après que vous tapiez « : % ! le xxd », qui commute pour ensorceller le mode et les zéros 16 octets commençant après que "5012" (50hex est 80 en décembre qui est type de Message-authentificateur, et 12 est la taille qui est 18 comprenant l'en-tête des paires de valeurs d'attribut (AVP)) :



Ensuite cette modification, la charge utile est prête. Il est nécessaire de retourner de nouveau à mode hexadécimal/binaire (type : « : % ! le xxd - r ») et sauvegardent le fichier (": wq »).

4. Utilisation OpenSSL afin de calculer HMAC-MD5 :

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

La fonction HMAD-MD5 prend deux arguments : le premier de l'entrée standard (stdin) est le message lui-même et le second est le secret partagé (Cisco dans cet exemple). Le résultat est exactement la même valeur que le Message-authentificateur relié dans le paquet de demande d'accès de RAYON.

Les mêmes peuvent être calculés avec l'utilisation du 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
```

L'exemple précédent présente comment calculer le champ de Message-authentificateur de l'Access-demande. Pour l'Access-défi, Access-recevez, et l'Access-anomalie, la logique est exactement identique, mais il est important de se souvenir que l'authentificateur de demande devrait être utilisé, qui est fourni dans le paquet de demande d'accès précédent.

Informations connexes

- RFC 2865
- RFC 2866
- RFC 3579
- Support et documentation techniques - Cisco Systems