Guida alla risoluzione dei problemi relativi all'autenticatore e all'autenticatore dei messaggi RADIUS non validi

Sommario

Introduzione

In questo documento vengono descritti due meccanismi di sicurezza RADIUS:

- Intestazione autenticatore
- Attributo Message-Authenticator

In questo documento vengono descritti i meccanismi di protezione, le relative modalità di utilizzo e i casi in cui è possibile che la convalida non venga eseguita correttamente.

Intestazione autenticatore

In base alla RFC 2865, l'intestazione Authenticator è lunga 16 byte. Quando viene utilizzato in una richiesta di accesso, viene denominato autenticatore richiesta. Quando viene utilizzato in qualsiasi tipo di risposta, viene definito autenticatore di risposta. Viene utilizzato per:

- Autenticazione della risposta
- Nascondere password

Autenticazione della risposta

Se il server risponde con l'autenticatore di risposta corretto, il client può eseguire il calcolo se la risposta è correlata a una richiesta valida.

Il client invia la richiesta con l'intestazione dell'autenticatore casuale. Il server che invia la risposta calcola quindi l'autenticatore della risposta con l'utilizzo del pacchetto di richiesta insieme al segreto condiviso:

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

Il client che riceve la risposta esegue la stessa operazione. Se il risultato è lo stesso, il pacchetto è corretto.

> **Nota:** L'autore dell'attacco che conosce il valore segreto non è in grado di falsificare la risposta a meno che non sia in grado di identificare la richiesta.

Quando è previsto un errore di convalida?

Se lo switch non memorizza più la richiesta nella cache, ad esempio a causa del timeout, si verificherà un errore di convalida. È inoltre possibile che si verifichi quando il segreto condiviso non è valido (sì - l'intestazione include anche Access-Reject). In questo modo, il dispositivo di accesso alla rete può rilevare la mancata corrispondenza del segreto condiviso. In genere viene segnalata dai client/server di autenticazione, autorizzazione e accounting (AAA) come una mancata corrispondenza della chiave condivisa, ma non ne vengono visualizzati i dettagli.

Nascondere password

L'intestazione Authenticator viene utilizzata anche per evitare l'invio dell'attributo User-Password in testo normale. Viene innanzitutto calcolato il Message Digest 5 (MD5 - segreto, autenticatore). Vengono quindi eseguite diverse operazioni XOR con i blocchi della password. Questo metodo è suscettibile agli attacchi offline (tabelle arcobaleno) poiché MD5 non viene più percepito come un algoritmo a senso unico.

Di seguito è riportato lo script Python che calcola la password dell'utente:

```
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]))
```

Ritrasmissioni

Se uno degli attributi nella richiesta di accesso RADIUS è stato modificato (come ID RADIUS, Nome utente e così via), è necessario generare il nuovo campo Autenticatore e ricalcolare tutti gli altri campi che dipendono da esso. Se questa è una ritrasmissione, nulla dovrebbe cambiare.

Contabilità

Il significato dell'intestazione Authenticator è diverso per Access-Request e Accounting-Request.

Per una richiesta di accesso, l'autenticatore viene generato in modo casuale e si prevede di ricevere una risposta con ResponseAuthenticator calcolato correttamente, che dimostra che la risposta è correlata a quella richiesta specifica.

Per una richiesta di accounting, l'autenticatore non è casuale, ma viene calcolato (in base alla RFC 2866):

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

In questo modo, il server può controllare immediatamente il messaggio di accounting e rilasciare il pacchetto se il valore ricalcolato non corrisponde al valore di Authenticator. Identity Services Engine (ISE) restituisce:

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

La causa tipica è la chiave segreta condivisa non corretta.

Attributo Message-Authenticator

L'attributo Message-Authenticator è l'attributo RADIUS definito nella RFC 3579. È utilizzato per uno scopo simile: per firmare e convalidare. Questa volta, tuttavia, non viene utilizzato per convalidare una risposta ma una richiesta.

Il client che invia una richiesta di accesso (può anche essere un server che risponde con una richiesta di accesso) calcola il codice HMAC (Hash-Based Message Authentication Code)-MD5 dal proprio pacchetto, quindi aggiunge l'attributo Message-Authenticator come firma. Il server è quindi in grado di verificare che esegua la stessa operazione.

La formula è simile all'intestazione Authenticator:

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

La funzione HMAC-MD5 accetta due argomenti:

- Il payload del pacchetto, che include il campo Message-Authenticator da 16 byte con zeri
- Il segreto condiviso

Quando utilizzare Message-Authenticator?

L'autenticatore del messaggio DEVE essere utilizzato per ogni pacchetto, incluso il messaggio EAP (Extensible Authentication Protocol) (RFC 3579). Sono inclusi sia il client che invia la richiesta di accesso sia il server che risponde con la richiesta di accesso. Se la convalida ha esito negativo, l'altro lato del pacchetto deve ignorare il pacchetto.

Quando è previsto un errore di convalida?

Se il segreto condiviso non è valido, si verificherà un errore di convalida. Quindi, il server AAA non è in grado di convalidare la richiesta.

The ISE dichiara:

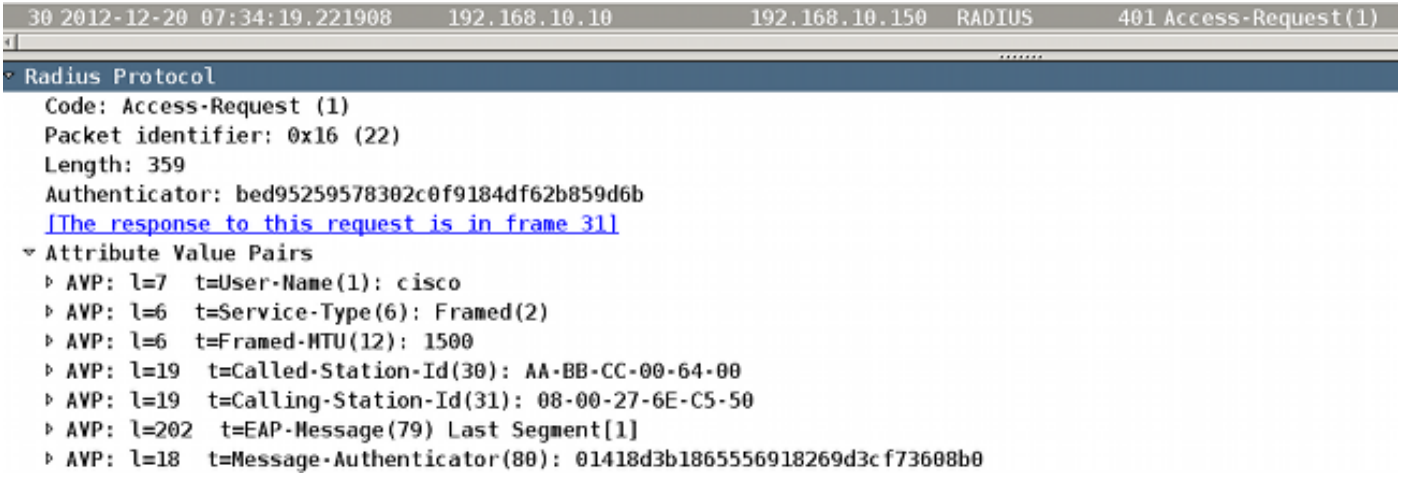
```
11036 The Message-Authenticator Radius Attribute is invalid.
```

Ciò si verifica in genere in una fase successiva quando il messaggio EAP viene allegato. Il primo pacchetto RADIUS della sessione 802.1x non include il messaggio EAP; non esiste un campo Message-Authenticator e non è possibile verificare la richiesta, ma in questa fase il client è in grado di convalidare la risposta utilizzando il campo Authenticator.

Convalida attributo Message-Authenticator

Di seguito è riportato un esempio per illustrare come contare manualmente il valore per accertarsi che venga calcolato correttamente.

Èstato scelto il pacchetto numero 30 (Access-Request). Si trova al centro della sessione EAP e il pacchetto include il campo Message-Authenticator. Lo scopo è verificare che l'autenticatore del messaggio sia corretto:

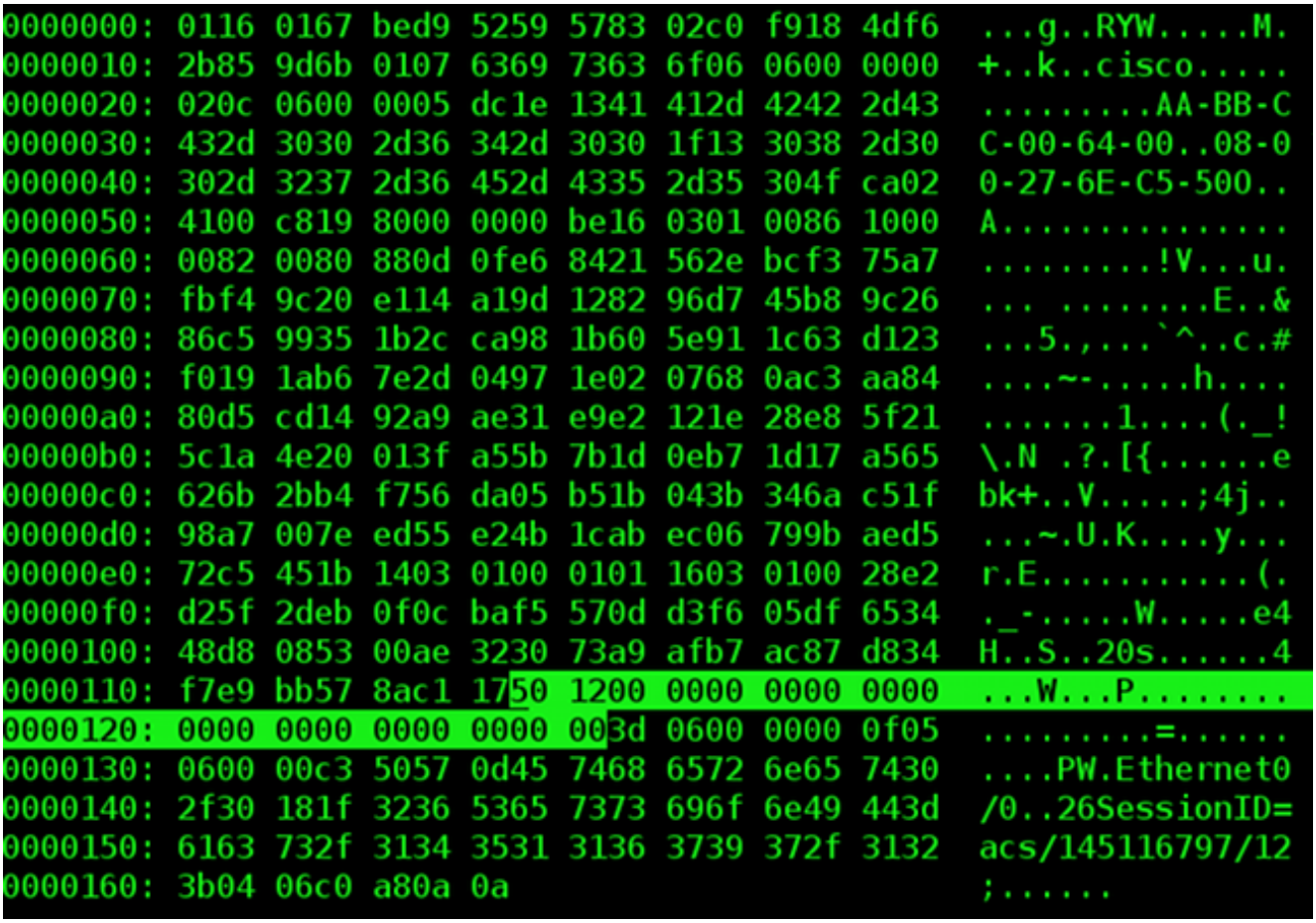


1. Fare clic con il pulsante destro del mouse su **Radius Protocol** e scegliere **Esporta byte pacchetto selezionati**.

2. Scrivere il payload RADIUS in un file (dati binari).

3. Per calcolare il campo Message-Authenticator, è necessario inserire degli zeri e calcolare l'HMAC-MD5.

   Ad esempio, quando si utilizza un editor esadecimale/binario, ad esempio vim, dopo aver digitato ":%!xxd", che passa alla modalità esadecimale e azzera 16 byte a partire da "5012" (50hex è 80 in decc, che è il tipo Message-Authenticator, e 12 è la dimensione 18, inclusa l'intestazione AVP (Attribute Value Pairs):



Dopo tale modifica, il payload è pronto. È necessario tornare alla modalità esadecimale/binaria (digitare: ":%!xxd -r") e salvare il file (":wq").

4. Utilizzare OpenSSL per calcolare HMAC-MD5:

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

La funzione HMAD-MD5 accetta due argomenti: il primo da standard input (stdin) è il messaggio stesso e il secondo è il segreto condiviso (Cisco in questo esempio). Il risultato è esattamente lo stesso valore di Message-Authenticator associato al pacchetto RADIUS Access-Request.

Lo stesso può essere calcolato usando lo script 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
```

Nell'esempio precedente viene illustrato come calcolare il campo Message-Authenticator da Access-Request. La logica di Access-Challenge, Access-Accept e Access-Reject è esattamente la stessa, ma è importante ricordare che è necessario utilizzare Request Authenticator, fornito nel pacchetto Access-Request precedente.

Informazioni correlate

- RFC 2865
- RFC 2866
- RFC 3579
- Documentazione e supporto tecnico – Cisco Systems