Guia de solução de problemas do autenticador e do autenticador de mensagem inválidos do RADIUS

Contents

Introduction

Este documento descreve dois mecanismos de segurança RADIUS:

- Cabeçalho do autenticador
- atributo Message-Authenticator

Este documento aborda o que são esses mecanismos de segurança, como eles são usados e quando você deve esperar uma falha de validação.

Cabeçalho do autenticador

Por RFC 2865, o Cabeçalho do Autenticador tem 16 bytes de comprimento. Quando usado em uma solicitação de acesso, é chamado de Autenticador de solicitação. Quando usado em qualquer tipo de resposta, é chamado de Autenticador de resposta. É utilizado para:

- Autenticação da resposta
- Senha oculta

Autenticação da resposta

Se o servidor responder com o Autenticador de resposta correto, o cliente poderá calcular se essa resposta foi relacionada a uma solicitação válida.

O cliente envia a solicitação com o cabeçalho aleatório do autenticador. Em seguida, o servidor que envia a resposta calcula o Autenticador de Resposta com o uso do pacote de solicitação junto com o segredo compartilhado:

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

O cliente que recebe a resposta executa a mesma operação. Se o resultado for o mesmo, o pacote está correto.

> **Note**: O invasor que sabe o valor secreto não pode falsificar a resposta, a menos que possa farejar a solicitação.

Quando você deve esperar uma falha de validação?

A falha de validação ocorre se o switch não armazena a solicitação em cache mais (por exemplo, devido ao tempo limite). Você também pode experimentá-lo quando o segredo compartilhado for inválido (sim - Access-Reject também inclui este cabeçalho). Dessa forma, o NAD (Network Access Device, dispositivo de acesso à rede) pode detectar a incompatibilidade de segredo compartilhado. Geralmente, ele é relatado por clientes/servidores AAA (Authentication, Authorization, and Accounting) como uma incompatibilidade de chave compartilhada, mas não revela os detalhes.

Ocultar senha

O cabeçalho do autenticador também é usado para evitar o envio do atributo User-Password em texto simples. Primeiro, o Message Digest 5 (MD5 - secret, autenticador) é computado. Em seguida, várias operações XOR com os pedaços da senha são executadas. Esse método é susceptível a ataques offline (tabelas do arco-íris) porque MD5 não é mais considerado um algoritmo unidirecional forte.

Aqui está o script Python que calcula a senha do usuário:

```
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]))
```

Retransmissões

Se algum dos atributos na Solicitação de Acesso RADIUS tiver sido alterado (como ID RADIUS, Nome de Usuário, etc.), o novo campo Authenticator deverá ser gerado e todos os outros campos que dependem dele deverão ser recompostos. Se isto é uma retransmissão, nada deve mudar.

Relatório

O significado do Cabeçalho do Autenticador é diferente para uma Solicitação de Acesso e uma Solicitação de Contabilidade.

Para uma solicitação de acesso, o Autenticador é gerado aleatoriamente e espera-se que receba uma resposta com o ResponseAuthenticator calculado corretamente, o que prova que a resposta foi relacionada a essa solicitação específica.

Para uma Solicitação de Contabilidade, o Autenticador não é aleatório, mas é calculado (conforme RFC 2866):

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

Dessa forma, o servidor pode verificar a mensagem contábil imediatamente e descartar o pacote se o valor recalculado não corresponder ao valor do Autenticador. O Identity Services Engine (ISE) retorna:

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

O motivo típico para isso é a chave secreta compartilhada incorreta.

Atributo de Autenticador de Mensagens

O atributo Message-Authenticator é o atributo RADIUS definido em RFC 3579. É utilizado para um fim semelhante: para assinar e validar. Mas desta vez, ele não é usado para validar uma resposta, mas uma solicitação.

O cliente que envia uma Solicitação de Acesso (também pode ser um servidor que responde com um Desafio de Acesso) calcula o Hash-Based Message Authentication Code (HMAC)-MD5 de seu próprio pacote e adiciona o atributo Message-Authenticator como uma assinatura. Em seguida, o servidor pode verificar se executa a mesma operação.

A fórmula é semelhante ao Cabeçalho do Autenticador:

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

A função HMAC-MD5 tem dois argumentos:

- O payload do pacote, que inclui o campo Message-Authenticator de 16 bytes preenchido com zeros
- O segredo compartilhado

Quando o Message-Authenticator deve ser usado?

O Message-Authenticator DEVE ser usado para cada pacote, que inclui a mensagem Extensible Authentication Protocol (EAP) (RFC 3579). Isso inclui o cliente que envia a solicitação de acesso e o servidor que responde com o desafio de acesso. O outro lado deve descartar silenciosamente o pacote se a validação falhar.

Quando você deve esperar uma falha de validação?

A validação falhará quando o segredo compartilhado for inválido. Em seguida, o servidor AAA não pode validar a solicitação.

O ISE relata:

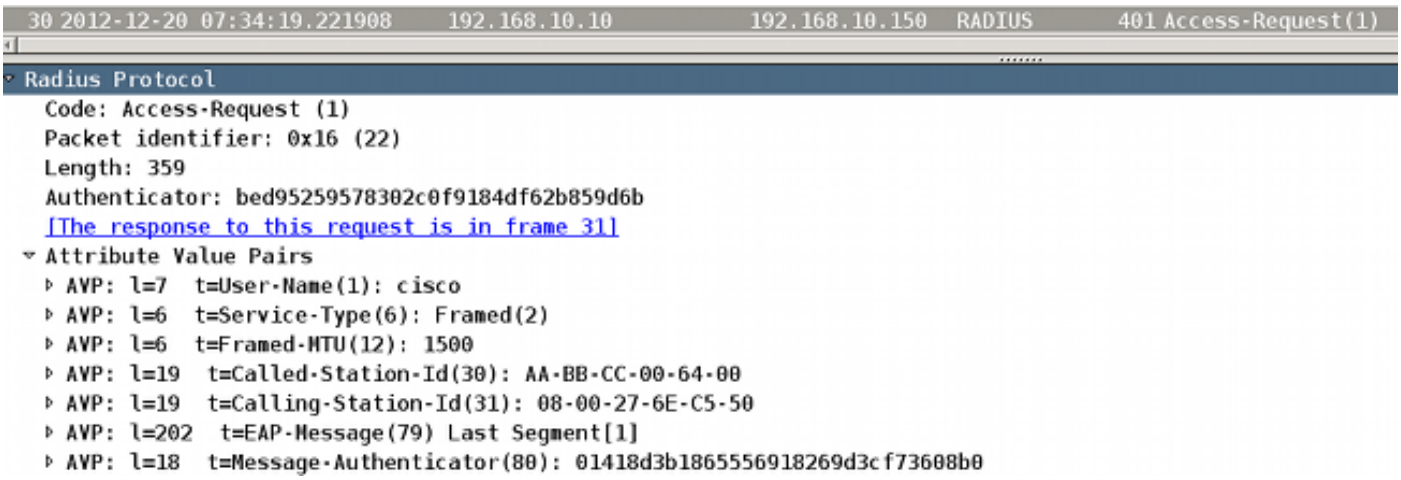
```
11036 The Message-Authenticator Radius Attribute is invalid.
```

Isso geralmente ocorre no estágio posterior em que a mensagem EAP é anexada. O primeiro pacote RADIUS da sessão 802.1x não inclui a mensagem EAP; não há campo Message-Authenticator e não é possível verificar a solicitação, mas nesse estágio, o cliente pode validar a resposta com o uso do campo Authenticator.

Validar o atributo do autenticador de mensagem

Aqui está um exemplo para ilustrar como você conta manualmente o valor para garantir que ele seja computado corretamente.

O pacote número 30 (Solicitação de acesso) foi escolhido. Ele está no meio da sessão EAP e o pacote inclui o campo Message-Authenticator. O objetivo é verificar se o Message-Authenticator está correto:

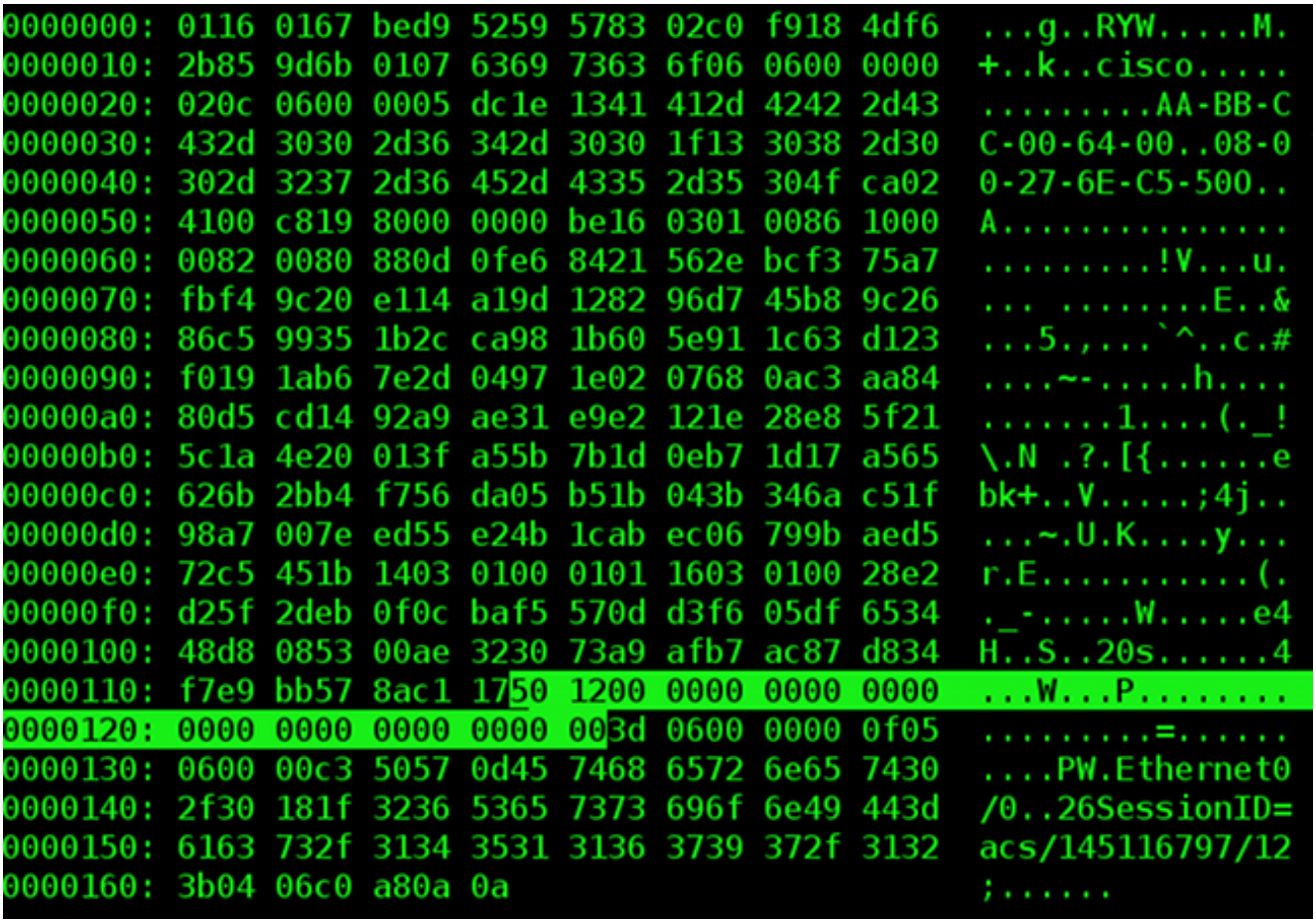


1. Clique com o botão direito do mouse em **RADIUS Protocol** e escolha **Exportar bytes de pacote selecionados**.

2. Escreva esse payload RADIUS em um arquivo (dados binários).

3. Para computar o campo Message-Authenticator, você deve colocar zeros lá e calcular o HMAC-MD5.

   Por exemplo, quando você usa o editor hex/binário, como vim, depois de digitar ":%!xxd", que muda para o modo hex e zera 16 bytes a partir de "5012" (50hex é 80 em dec, que é o tipo Message-Authenticator, e 12 é o tamanho que é 18 incluindo os Pares de Valor do Atributo (AVP)):



Depois dessa modificação, o payload está pronto. É necessário voltar ao modo hexadecimal/binário (digite: ":%!xxd -r") e salve o arquivo (":wq").

4. Use o OpenSSL para calcular o HMAC-MD5:

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

A função HMAD-MD5 tem dois argumentos: a primeira de entrada padrão (stdin) é a própria mensagem e a segunda é o segredo compartilhado (a Cisco neste exemplo). O resultado é exatamente o mesmo valor que o Message-Authenticator anexado ao pacote RADIUS Access-Request.

O mesmo pode ser computado com o uso do 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
```

O exemplo anterior mostra como calcular o campo Message-Authenticator a partir da Solicitação de acesso. Para Access-Challenge, Access-Accept e Access-Reject, a lógica é exatamente a mesma, mas é importante lembrar que o Request Authenticator deve ser usado, que é fornecido no pacote Access-Request anterior.

Informações Relacionadas

- RFC 2865
- RFC 2866
- RFC 3579
- Suporte Técnico e Documentação - Cisco Systems