RADIUS 잘못된 인증자 및 메시지 인증자 문제 해결 가이드

목차

소개

이 문서에서는 두 가지 RADIUS 보안 메커니즘에 대해 설명합니다.

- 인증자 헤더
- Message-Authenticator 특성

이 문서에서는 이러한 보안 메커니즘의 정의, 사용 방법 및 유효성 검사 실패가 예상되는 시기를 다룹니다.

인증자 헤더

RFC 2865에 따라 Authenticator Header의 길이는 16바이트입니다.Access-Request에서 사용할 경우 요청 인증자라고 합니다.어떤 종류의 응답에서도 사용되는 경우 응답 인증자라고 합니다.사용 용도:

- 응답 인증
- 암호 숨기기

응답 인증

서버가 올바른 응답 인증자로 응답하면 클라이언트가 해당 응답이 유효한 요청과 관련되었는지 확인할 수 있습니다.

클라이언트가 임의 인증자 헤더로 요청을 전송합니다.그런 다음 응답을 전송하는 서버는 공유 암호와 함께 요청 패킷을 사용하여 응답 인증자를 계산합니다.

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

응답을 수신하는 클라이언트는 동일한 작업을 수행합니다.결과가 동일한 경우 패킷이 정확합니다.

참고:비밀 값을 아는 공격자는 요청을 스푸핑할 수 없는 한 응답을 스푸핑할 수 없습니다.

검증 실패 시기는 언제입니까?

스위치가 더 이상 요청을 캐시하지 않는 경우(예: 시간 초과) 검증 오류가 발생합니다. 공유 암호가 잘못되었을 때(예 - Access-Reject에도 이 헤더가 포함되어 있음) 이 문제가 발생할 수도 있습니다. 이렇게 하면 NAD(Network Access Device)가 공유 암호 불일치를 탐지할 수 있습니다.일반적으로 AAA(Authentication, Authorization, and Accounting) 클라이언트/서버에서 공유 키 불일치로 보고되지만 세부 정보는 표시되지 않습니다.

비밀번호 숨기기

Authenticator Header는 일반 텍스트로 User-Password 특성을 보내지 않도록 하는 데에도 사용됩니다.먼저 메시지 다이제스트 5(MD5 - 비밀, 인증자)가 계산됩니다.그런 다음 비밀번호의 청크와 함께 여러 XOR 작업이 실행됩니다.MD5가 더 이상 강력한 단방향 알고리즘으로 인식되지 않으므로 이 방법은 오프라인 공격에 취약합니다(레인보우 테이블).

다음은 사용자-비밀번호를 계산하는 Python 스크립트입니다.

```
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]))
```

재전송

RADIUS 액세스 요청의 특성이 변경된 경우(예: RADIUS ID, 사용자 이름 등) 새 인증자 필드가 생성되어야 하며 여기에 종속된 다른 모든 필드는 다시 계산되어야 합니다.재전송이라면 아무것도 변하지 않을 것이다.

회계

Authenticator Header의 의미는 Access-Request 및 Accounting-Request와 다릅니다.

Access-Request의 경우 인증자가 임의로 생성되며, 응답이 해당 특정 요청과 관련되었음을 나타내는 ResponseAuthenticator가 올바르게 계산된 응답을 수신해야 합니다.

어카운팅 요청의 경우 인증자는 무작위가 아니지만 RFC 2866에 따라 계산됩니다.

```
RequestAuth = MD5(Code + ID + Length + 16 zero octets + Attributes + Secret)
```
이렇게 하면 서버는 계정 메시지를 즉시 확인하고 재계산된 값이 인증자 값과 일치하지 않으면 패킷을 삭제할 수 있습니다.ISE(Identity Services Engine)는 다음을 반환합니다.

```
11038 RADIUS Accounting-Request header contains invalid Authenticator field
```
일반적인 이유는 잘못된 공유 비밀 키입니다.

메시지 인증자 특성

Message-Authenticator 특성은 RFC 3579에 정의된 RADIUS 특성입니다.유사한 용도로 사용됩니다.서명하고 검증합니다.그러나 이번에는 응답을 검증하기 위한 것이 아니라 요청을 위해 사용됩니다.

액세스 요청을 보내는 클라이언트(액세스 챌린지로 응답하는 서버일 수도 있음)는 자체 패킷에서 HMAC(Hash-Based Message Authentication Code)-MD5를 계산한 다음 Message-Authenticator 특성을 서명으로 추가합니다.그런 다음 서버에서 동일한 작업을 수행하는지 확인할 수 있습니다.

수식은 인증자 헤더와 유사합니다.

```
Message-Authenticator = HMAC-MD5 (Type, Identifier, Length, Request Authenticator,
Attributes)
```
HMAC-MD5 함수는 두 개의 인수를 사용합니다.

- 16바이트 Message-Authenticator 필드가 0으로 채워진 패킷의 페이로드
- 공유 암호

메시지 인증자는 언제 사용해야 합니까?

EAP(Extensible Authentication Protocol) 메시지(RFC 3579)를 포함하는 모든 패킷에 메시지 인증자를 사용해야 합니다. 여기에는 Access-Request 를 전송하는 클라이언트와 Access-Challenge에 응답하는 서버가 모두 포함됩니다.검증이 실패하면 다른 쪽에서 패킷을 자동으로 삭제해야 합니다.

검증 실패 시기는 언제입니까?

공유 암호가 유효하지 않으면 유효성 검사 오류가 발생합니다.그러면 AAA 서버가 요청을 검증할 수 없습니다.

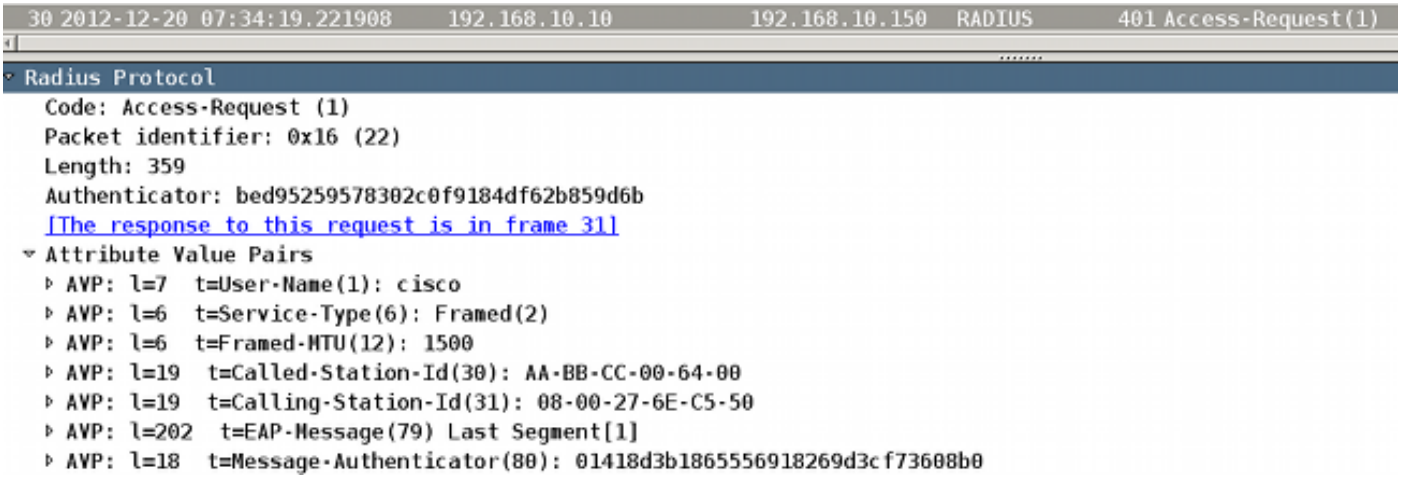
ISE는 다음을 보고합니다.

```
11036 The Message-Authenticator Radius Attribute is invalid.
```
일반적으로 EAP 메시지가 연결된 이후 단계에서 발생합니다.802.1x 세션의 첫 번째 RADIUS 패킷에는 EAP 메시지가 포함되지 않습니다.Message-Authenticator 필드가 없고 요청을 확인할 수 없지만, 이 단계에서 클라이언트는 Authenticator 필드를 사용하여 응답을 검증할 수 있습니다.

메시지 인증자 특성 확인

이 예는 값이 올바르게 계산되도록 수동으로 계산하는 방법을 보여 줍니다.

패킷 번호 30(Access-Request)이 선택되었습니다.EAP 세션의 중간에 있으며 패킷에 Message-Authenticator 필드가 포함됩니다.목표는 메시지 인증자가 올바른지 확인하는 것입니다.

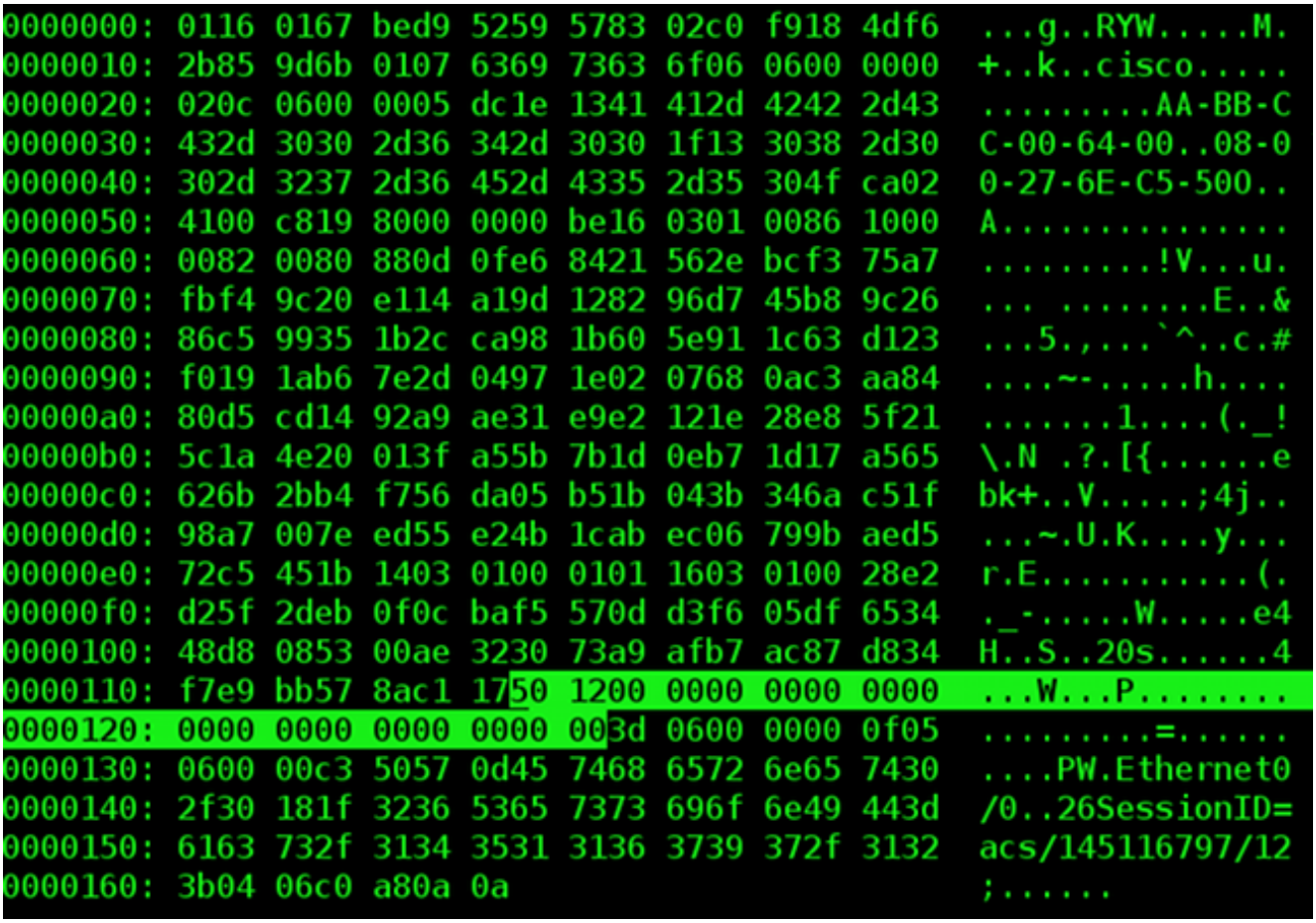


1. Radius **Protocol**을 마우스 오른쪽 버튼으로 클릭하고 **Export selected packet bytes**를 선택합니다.

2. 해당 RADIUS 페이로드를 파일(이진 데이터)에 기록합니다.

3. Message-Authenticator 필드를 계산하려면 0을 여기에 입력하고 HMAC-MD5를 계산해야 합니다.

   예를 들어 vim과 같은 16진수/이진 편집기를 사용할 경우 ":%!xxd"를 입력한 후 16진수 모드로 전환하고 "5012" 이후에 시작하는 16바이트를 0으로 전환합니다(50hex는 12의 80이고 12는 AVP(Attribute Value Pairs) 헤더를 포함하여 18입니다.).



그런 다음 페이로드가 준비됩니다.16진수/이진 모드로 다시 돌아가야 합니다(유형:":%!xxd -r") 및 파일(":wq")을 저장합니다.

4. HMAC-MD5를 계산하려면 OpenSSL을 사용합니다.

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

HMAD-MD5 함수는 두 개의 인수를 사용합니다.표준 입력(stdin)의 첫 번째 항목은 메시지 자체이고 두 번째 항목은 공유 비밀(이 예에서는 Cisco)입니다. 결과는 RADIUS Access-Request 패킷에 연결된 메시지 인증자와 정확히 동일한 값입니다.

Python 스크립트를 사용하여 동일한 값을 계산할 수 있습니다.

```
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
```

위의 예에서는 Access-Request에서 Message-Authenticator 필드를 계산하는 방법을 보여 줍니다 .Access-Challenge, Access-Accept 및 Access-Reject의 경우 논리는 정확히 동일하지만, 이전 Access-Request 패킷에 제공된 요청 인증자를 사용해야 한다는 점을 기억해야 합니다.

관련 정보

- RFC 2865
- RFC 2866
- RFC 3579
- 기술 지원 및 문서 – Cisco Systems