JPNIC・JPCERT/CCセキュリティセミナー2005
サーバアプリケーションセキュリティ

RADIUSで実現する認証サーバ ～概要および脆弱性と対策～

株式会社アクセンス・テクノロジー
取締役・研究開発部部長
納村 康司

目次

◆RADIUSとは・・・

RADIUS ＝ Remote Access Dial In User Service

- 元来はダイアルアップによるリモート接続でのユーザ認証プロトコル
- 近年は、ダイアルアップだけでなく、ブロードバンド接続や、VPN、VLAN、無線LANなどへの接続ユーザ認証にも利用されている
- RFC 2865、2866、2867、2868、2869、3162、3579、3580など

RADIUSの歴史

1992年　米Livingston社（当時）の独自ユーザ認証プロトコルとして開発される

1997年　RFCの策定が行われる。細かな修正に伴い、数ヶ月で改訂（RFC 2138および2139）

2000年　RFC改訂。それに伴い、VPN装置への対応や、拡張機能についてのRFCも策定される（RFC 2865から2869）

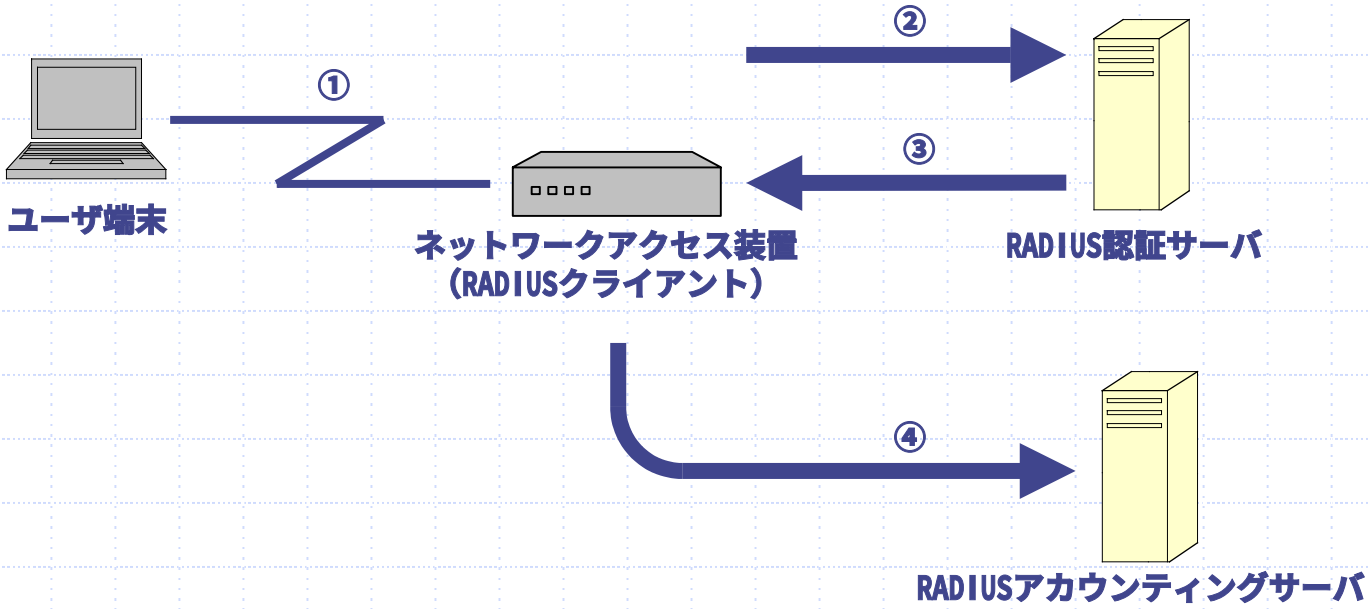
2001年　IPv6対応規格が策定される(RFC 3162)

2003年　EAP認証への対応規格(RFC 3579)、およびIEEE 802.1X対応へのガイドライン(RFC 3580)が策定される

RADIUSの特徴

- AAAモデルをサポートしている
- UDPベースのプロトコルで、コネクションレスで情報のやり取りを行う
- hop-to-hopのセキュリティモデルを採用している
- 共有鍵とMD5アルゴリズムを利用して、パスワードの隠蔽やパケットの整合性確認を行う
- AVP(Attribute Value Pair)による、柔軟な拡張性
- 基本機能としてPPPのPAP認証とCHAP認証を、拡張機能としてEAP認証をサポートしている

構成図



基本シーケンス

① ユーザ端末からネットワークアクセス装置に接続を開始する

② ネットワークアクセス装置は、RADIUSクライアントとして、接続要求をRADIUS認証サーバに送る。 RADIUS認証サーバでは、送られた情報をもとに認証を行う

③ 認証の結果を、RADIUS認証サーバからネットワークアクセス装置に送信する。その際、ユーザに対する承認情報も合わせて送る

④ ネットワークアクセス装置では、通知された結果をもとに、ユーザ端末の接続／切断を行う。接続を行った場合は、その事実をRADIUSアカウンティングサーバに送信し、記録を行う

AAAモデル

- Authentication（認証）
  ユーザが示した身分を確認する処理。身分を示す情報として、ユーザIDとパスワードや、電子証明書などがある

- Authorization（承認）
  ユーザにどのような振る舞いを許可するのか決定する処理

- Accounting（アカウンティング）
  ユーザがサービスを利用した事実とその内容を計測し、記録する処理

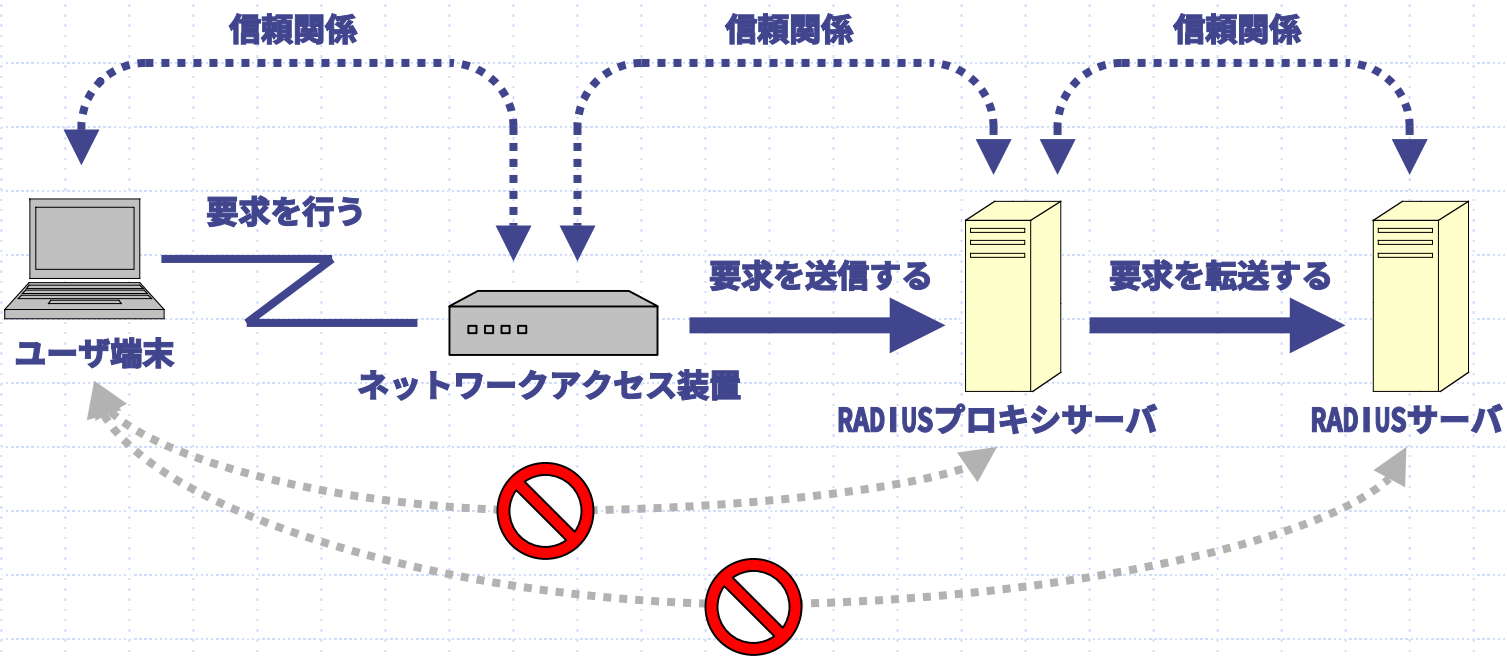
RADIUSはUDPベース

- 独自のタイミングによる再送処理が必要
- コネクションレスな設計で、構成がシンプルにできる
- 多数の処理を同時に扱う同時並行処理が行いやすい

これらの理由によりUDPを利用している

- UDPであることによる、パケットの破損や喪失に対しては、プロトコルの機能として対応している
- 使用するポート番号は、RADIUS認証においては1812番、RADIUSアカウンティングにおいては1816番

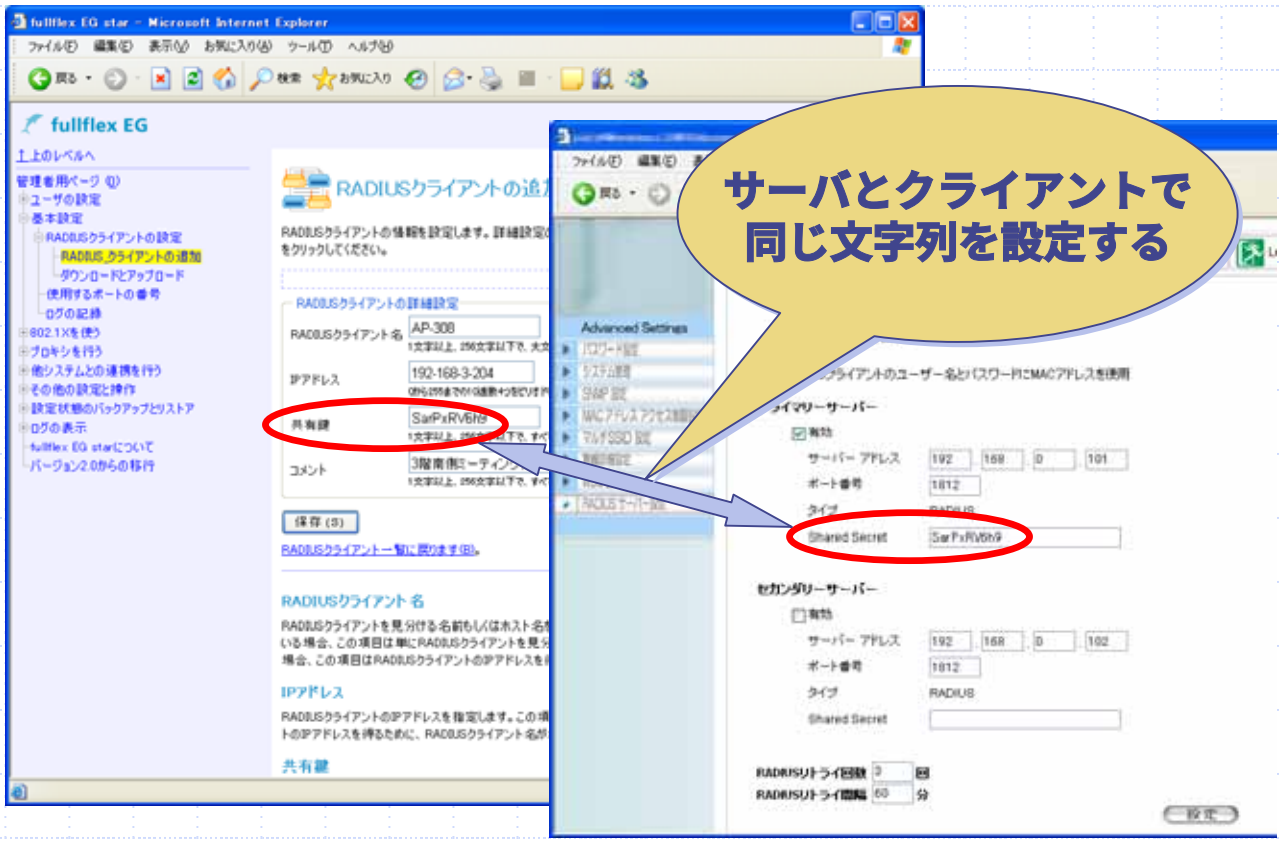
hop-to-hopのセキュリティ



hop-to-hopでは、ユーザ端末と中継するRADIUSサーバや最終的なRADIUSサーバの間に信頼関係はない。

共有鍵

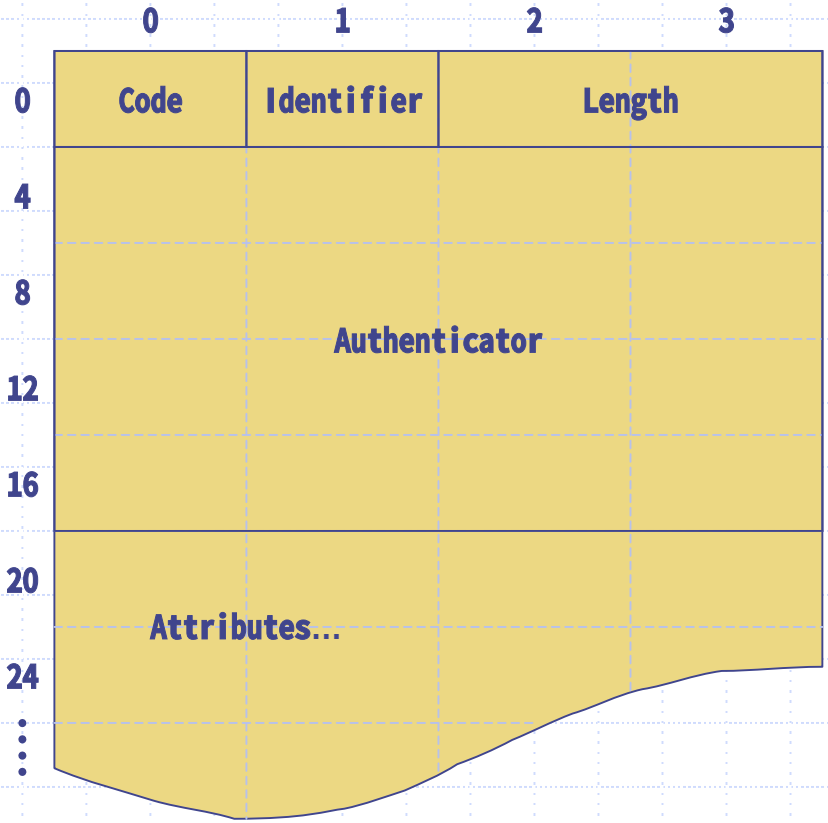
RADIUSサーバとクライアント間で共有するパスワード



共有鍵(2)

- RADIUSサーバとクライアントの対に設定する
- ネットワーク上に流れることはない
- どの共有鍵を使って処理をするかは、クライアントのソースIPアドレスを基に決定する
- RADIUSのセキュリティ機能の多くが、共有鍵を利用して実現されている

RADIUSパケットの構造



可変長のパケット
最大サイズは、4096オクテット

【Code】
RADIUSパケットの種別を示す
1 = Access-Request
2 = Access-Accept
3 = Access-Reject
4 = Accounting-Request
5 = Accounting-Response
11 = Access-Challenge
など

【Identifier】
特定のパケットを識別する連番

【Length】
パケットの長さ情報

Authenticator

【Authenticator】

16オクテットのバイナリデータ

- Request Authenticator ＝ Access-RequestのAuthenticator
- Response Authenticator ＝ 応答のAuthenticator

があり、用途が異なる

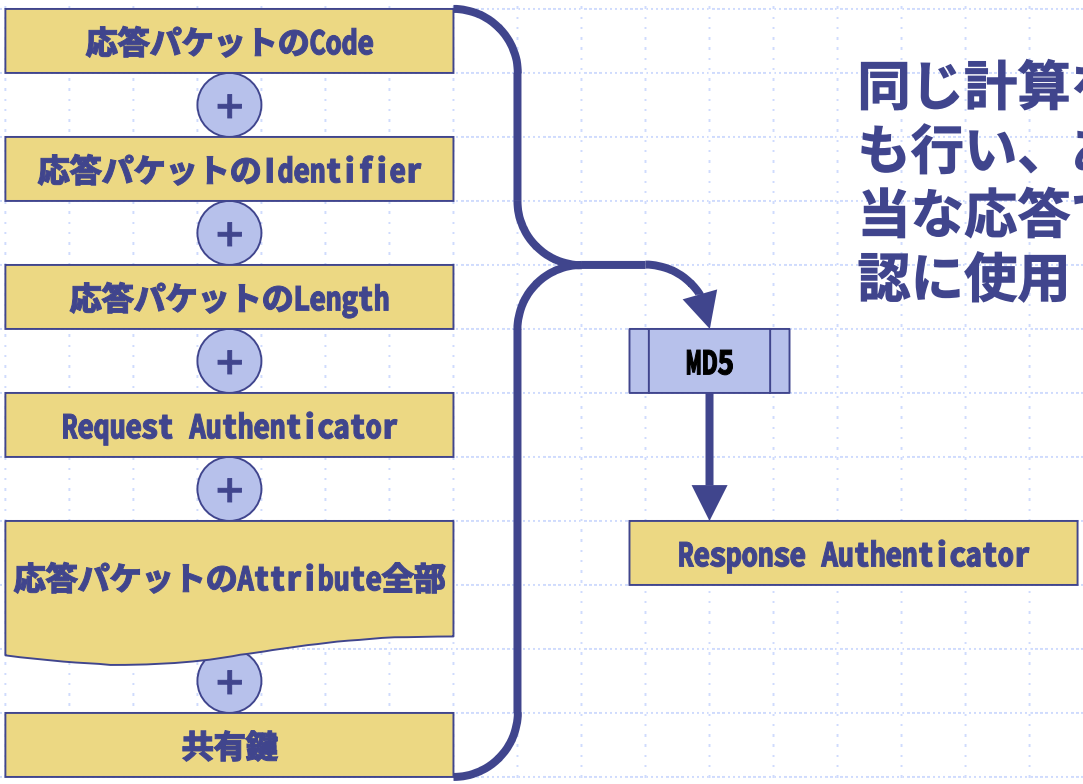
Request Authenticator

Request Authenticatorは、ランダムに生成した16オクテットのデータ

PAP認証時のユーザパスワードの隠蔽や、要求に対する応答の整合性の確認に、Nonceとして使用

Response Authenticator

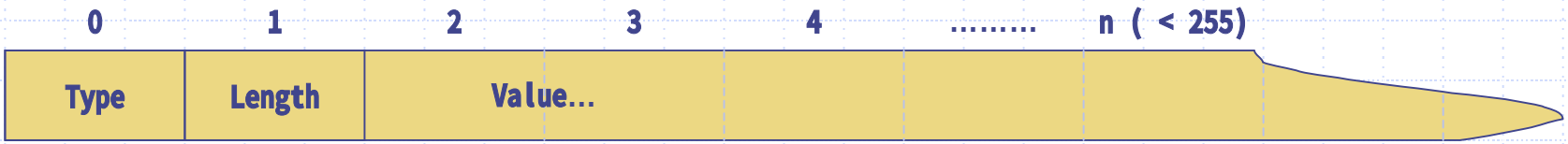
Response Authenticatorは、以下のように算出する、16オクテットの値



同じ計算をRADIUSクライアントでも行い、あるRequestに対する正当な応答であるかの、整合性の確認に使用

RADIUS属性

- RADIUSで扱う各種情報の表現単位
- 可変長のオクテット列



【Type】
属性のタイプを示す。1 = User-Name、2 = User-Password、など

【Length】
属性の長さを示す

【Value】
- 可変長のオクテット列。内容は属性タイプごとに異なる

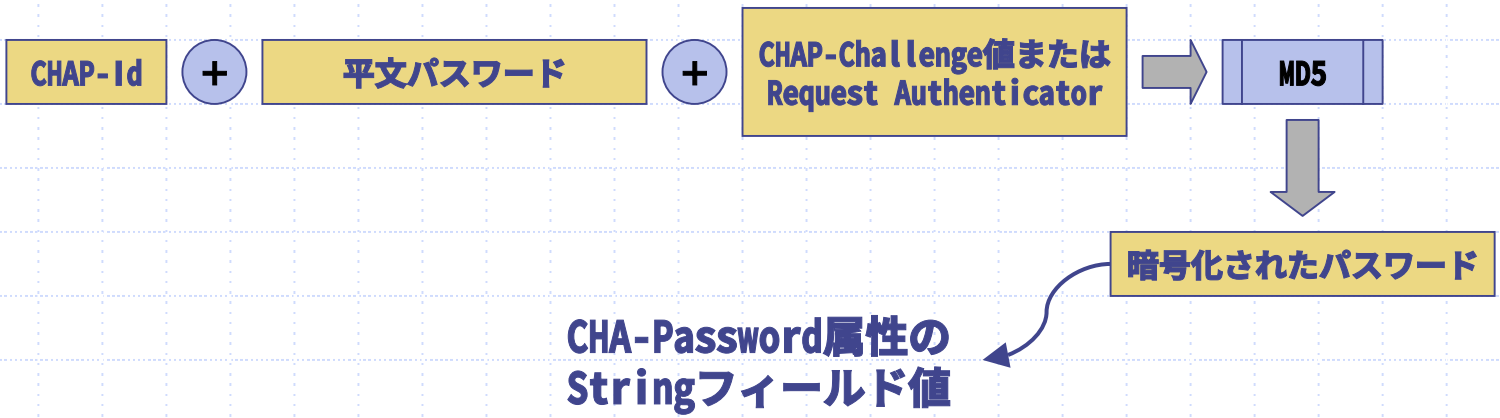
パスワード暗号化(PAP認証)

共有鍵とRequest Authenticatorを用いて、
平文のパスワードを暗号化

逆手順により、平文のパスワードに復号することができる

パスワード暗号化(CHAP認証)

CHAP-Challenge属性の値、またはRequest Authenticator
を用いて、平文のパスワードを暗号化



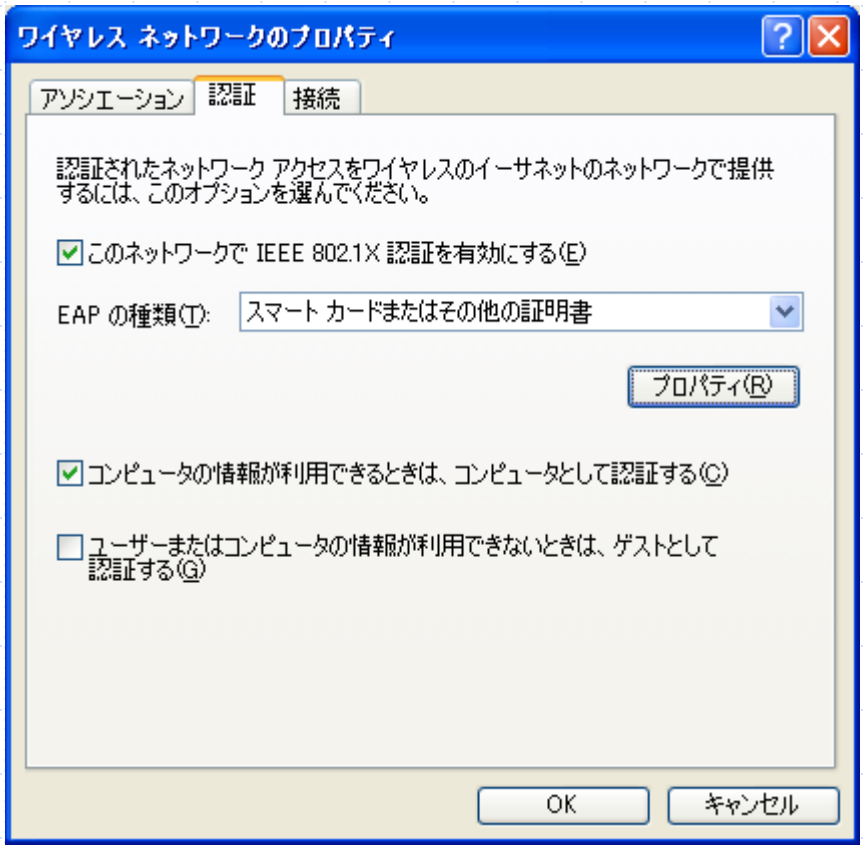
CHAP-Passwordの暗号化には、共有鍵を使用しない
また、 CHAP-Passwordは復号化することができない

EAP

EAP = Extensible Authentication Protocol

- PPPにおいて、PAP認証やCHAP認証以外の、様々な認証方法をサポートするための、拡張フレームワーク
  (RFC 3748)
- IEEE 802.1Xにおいて、Ethernetのような802ネットワークの接続認証用プロトコルとして、RADIUSとともに採用された

EAP(2)



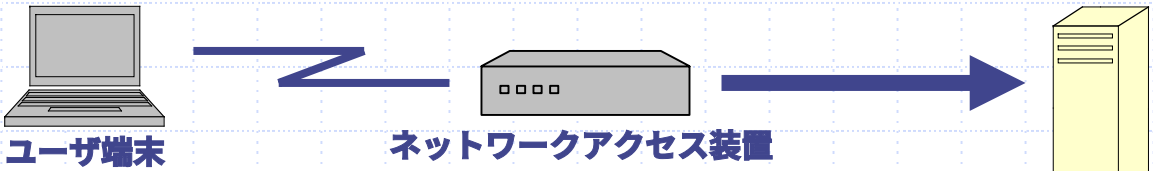
- 具体的な認証方法は別途定められる
- Windows XPのEAP認証方式の選択画面
  EAP-TLS(RFC 2716)や、PEAPなどの選択ができる

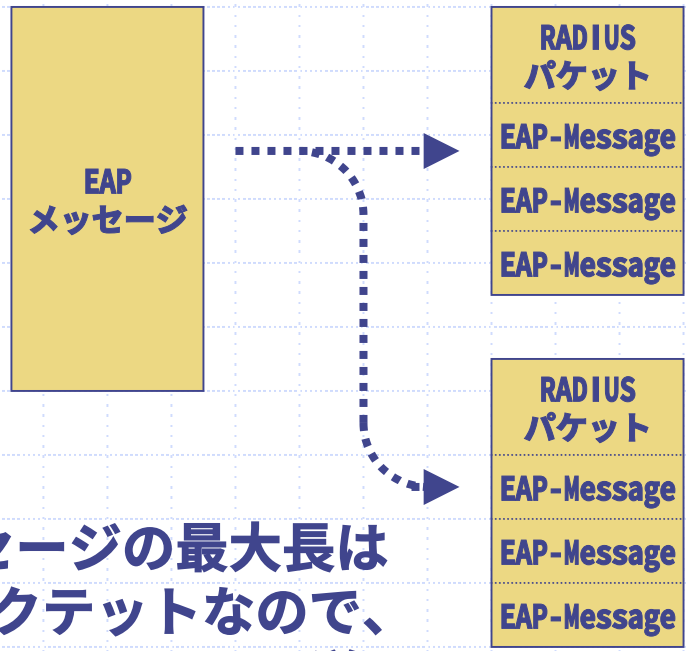
EAPの種類

EAPには多くの種類がある

EAP タイプ	オープン/ 商用	相互 認証	認証情報		規格
			ユーザ端末	認証サーバ	
MD5	オープン	しない	ユーザ名/パスワード	なし	RFC 3748
OTP	オープン	しない	ワンタイムパスワード	なし	RFC 3748
GTC	オープン	しない	トークンカード	なし	RFC 3748
TLS	オープン	する	証明書	証明書	RFC 2716
TTLS	オープン	する	ユーザ名/パスワード	証明書	Draft
PEAP	オープン	する	ユーザ名/パスワード	証明書	Draft
FAST	オープン	する	ユーザ名/パスワード	アクセス証明キー	Draft
LEAP	商用	する	ユーザ名/パスワード	なし	ー

RADIUSとEAP





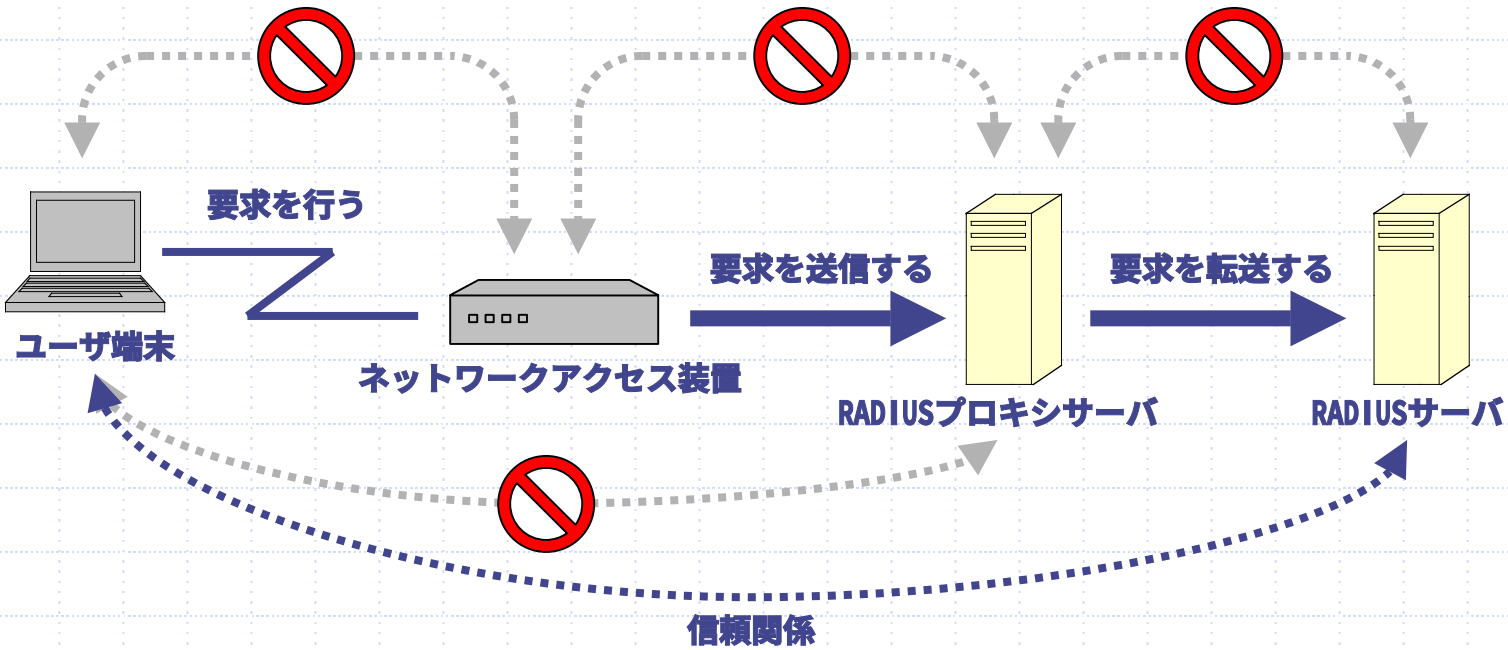
ユーザ端末から送られたEAPメッセージは、複数のEAP-Message属性を持つRADIUSパケットにカプセル化され、RADIUSサーバに送られる

EAPメッセージの最大長は65,536オクテットなので、そのままではRADIUSで送れない場合がある

EAP-Message属性を持つRADIUSパケットでは、Message-Authenticator属性が必須となる

end-to-endのセキュリティ

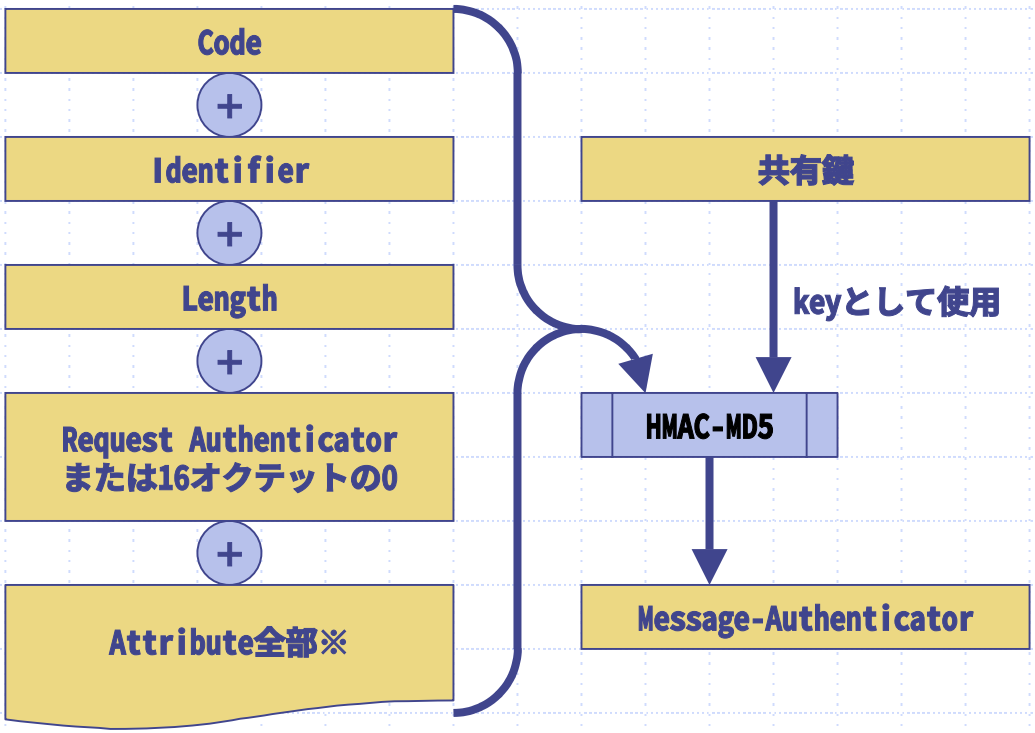
EAPのいくつかの種別では、end-to-endのセキュリティモデルを採用（PEAPやEAP-TTLS）



end-to-endでは、ユーザ端末と最終的なRADIUSサーバの間に信頼関係ができる

Message-Authenticator属性

- Response Authenticatorと同様の役割を果たす、拡張属性（RFC 2869）

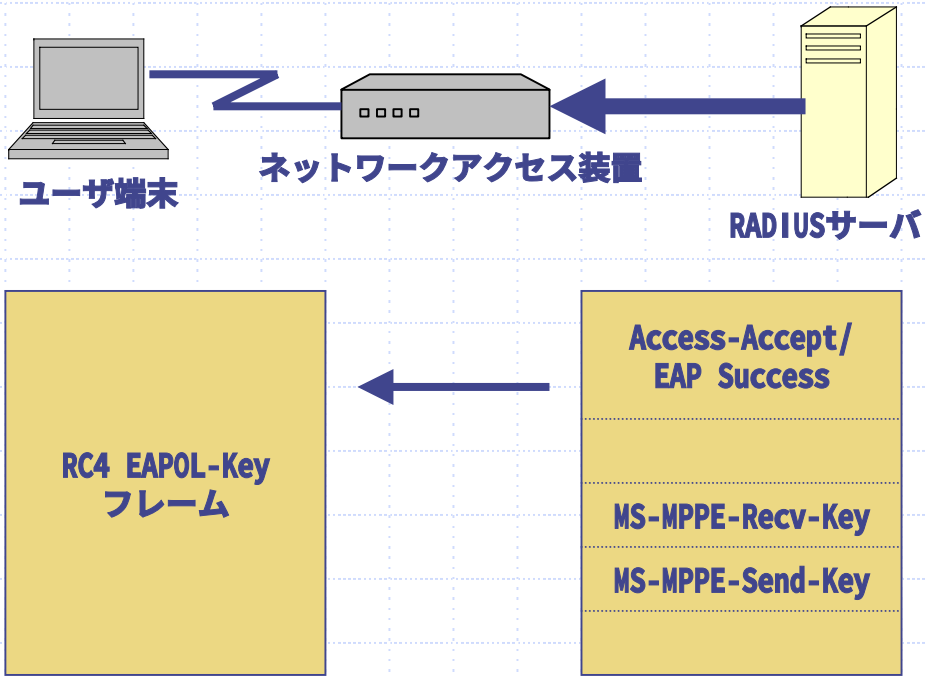


同じ計算をサーバとクライアント双方で行い、RADIUSパケットの整合性の確認に使用する

※Message-Authenticator属性の値部分には、16オクテットの0をパディング

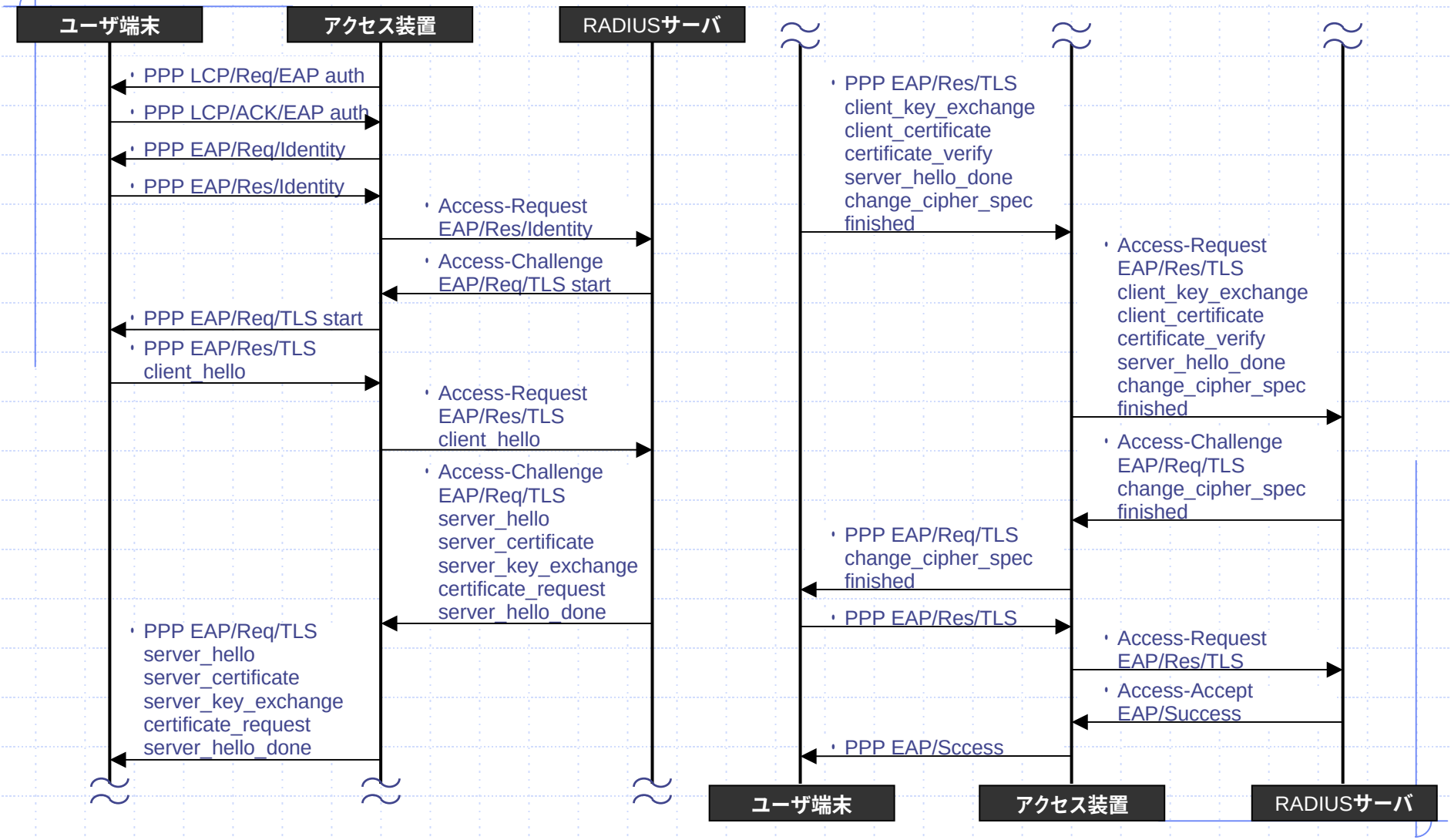
鍵配布

RADIUSを用いた、ユーザ端末とネットワークアクセス装置の間の動的鍵配布の方法は、以下の通り



- 動的鍵の配布は、RC4 EAPOL-Keyフレームによって、ネットワークアクセス装置からユーザ端末に向かって行われる
- しかしネットワークアクセス装置は、 RC4 EAPOL-Keyフレームの内容を暗号化するためのkeying materialを知らない
- RC4 EAPOL-Keyフレームの暗号化に使用するkeying materialは、RADIUSサーバから渡される
- keying materialの受け渡しには、 MS-MPPE-Recv-KeyおよびMS-MPPE-Recv-Keyの２つのベンダー拡張属性を使う
- MS-MPPE-Recv-KeyおよびMS-MPPE-Recv-Keyの内容の隠蔽には、User-Password属性と同じ方法を使う

EAPシーケンス(EAP-TLS)

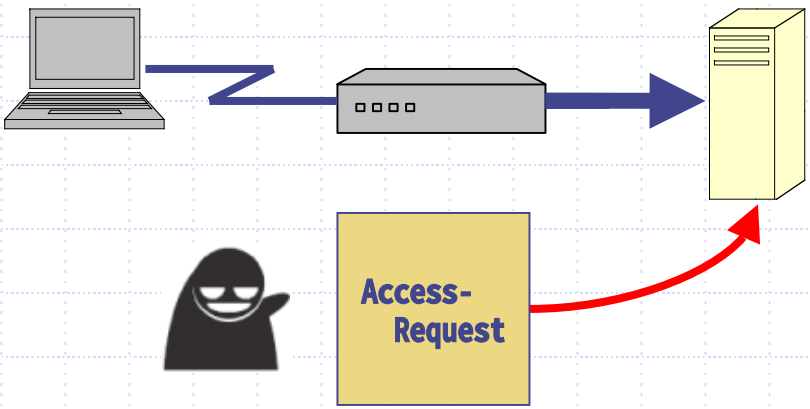


◆RADIUSの脆弱性

- クライアントなりすまし
- Access-Requestの改竄
- Offline Dictionary攻撃
- Known Plaintext攻撃
- より簡単な共有鍵の特定
- CHAP-Passwordの解読
- Request Authenticator重複
- Replay攻撃
- Man in the Middle攻撃
- Negotiation攻撃

クライアントなりすまし

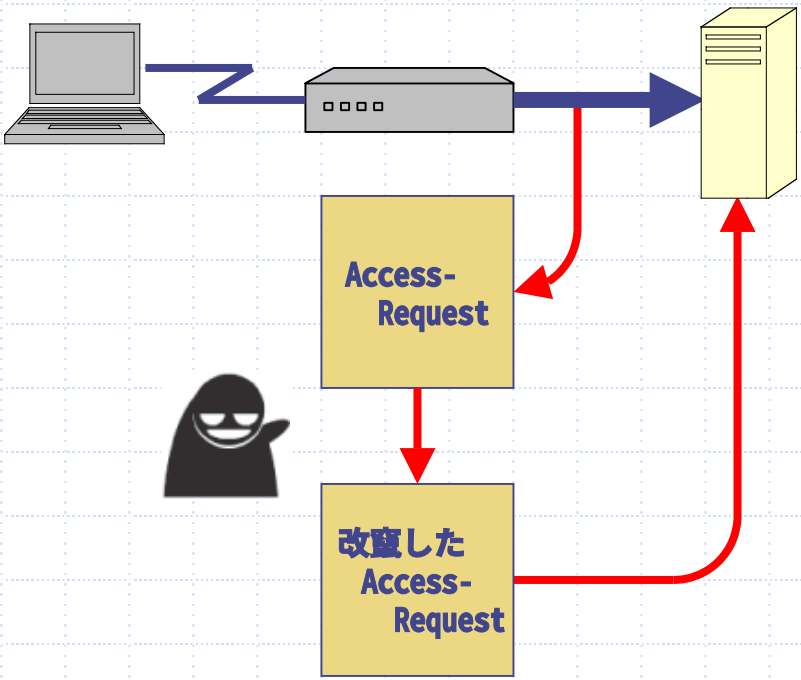
RADIUSサーバでは、クライアントのソースIPアドレスを基に、どの共有鍵を使用するかが決定するため、IPアドレスを偽装できる攻撃者は、容易にRADIUSクライアントのフリができる



IPアドレスを偽装できる攻撃者は、パスワードのクラックや、共有鍵の解読のために、RADIUSクライアントになりすまして、Access-Requestを送ることができる

Access-Requestの改竄

Access-Requestでは、Authenticatorを用いた整合性検査ができなため、 Access-Requestの改竄が調べられない



Access-Requestに含まれるRequest Authenticatorは、16オクテットの乱数値

↓

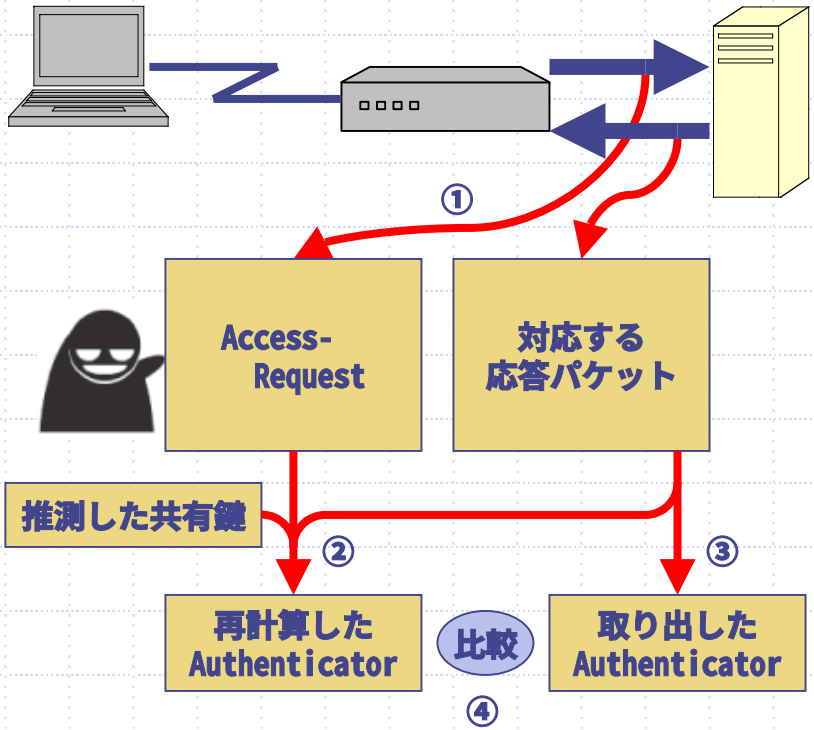
Access-Requestでは、Authenticatorを用いた整合性チェックができない

↓

Access-Requestの改竄が調べられない

Offline Dictionary攻撃

Access-Requestと、それに対応する応答パケットを一組傍受できる攻撃者は、傍受したパケットに対しオフラインでDictionary攻撃を試みることで、共有鍵を特定できる



① 攻撃者はAccess-Requestと、それに対する応答パケットを一組傍受する

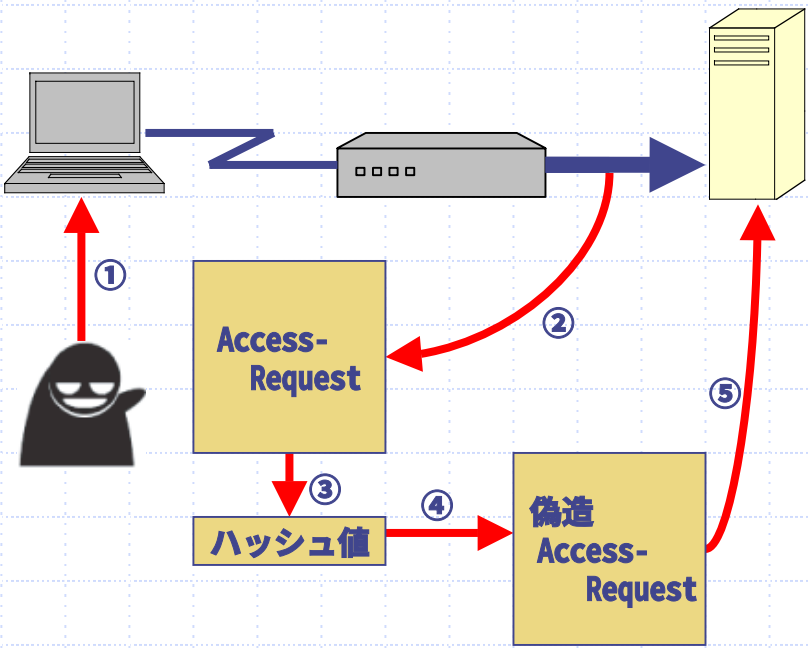
② 推測した共有鍵、Request-Authenticator、応答パケットの各情報を使って、Response Authenticatorを再計算する

③ 応答パケットから実際のResponse Authenticatorを取り出し比較する

④ ②～③を繰り返すことで、オフラインで共有鍵の特定が行える

Known Plaintext攻撃

ユーザとしてパスワードを試すことができ、かつネットワークに流れるパケットを傍受できる攻撃者は、共有鍵を知らなくても、自由にUser-Password属性を持つAccess-Requestを生成することができる



① 攻撃者は正規の方法で既知の平文(Known Plaintext)をパスワードとして入力する

② RADIUSサーバに送られるAccess-Requestを傍受する

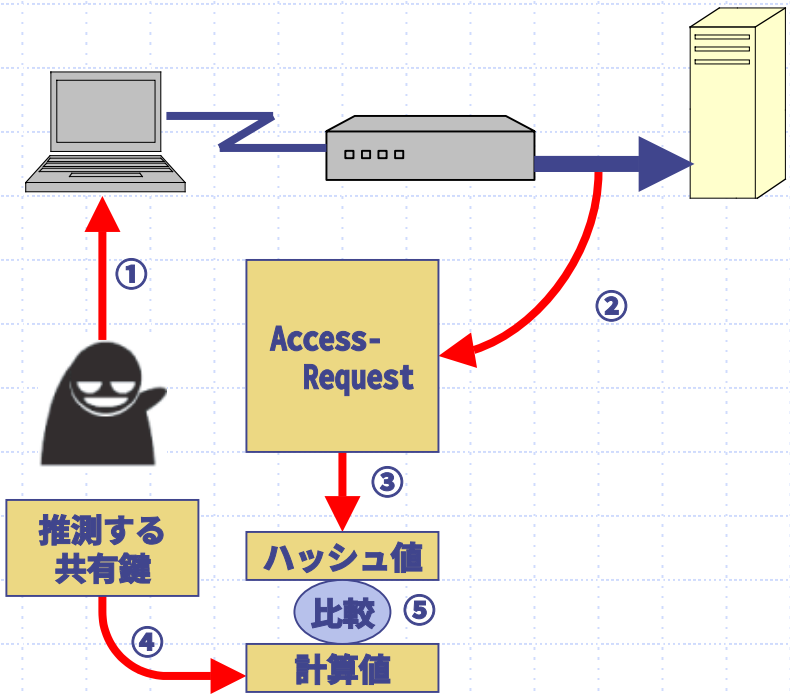
③ 傍受したAccess-Request中のUser-Password属性の値に既知の平文をxorすることで、ユーザパスワードの隠蔽の際に使用した、Request Authenticatorと共有鍵のMD5ハッシュ値を求める

④ 上で得たハッシュ値を使って任意のAccess-Requestを偽造する

⑤ 偽造したAccess-RequestをRADIUSサーバに送り、結果を調べることで、様々なパスワードをオンラインで試行できる（ただし16オクテット以内のパスワードに限る）

より簡単な共有鍵の特定

Offline Dictionary攻撃と、Known Plaintext攻撃を組み合わせることで、攻撃者はより簡単に共有鍵の特定を行える



① 攻撃者は正規の方法で既知の平文をパスワードとして入力する

② RADIUSサーバに送られるAccess-Requestを傍受する

③ 傍受したAccess-Request中のUser-Password属性の値に既知の平文をxorすることで、Request Authenticatorと共有鍵のMD5ハッシュ値を求める

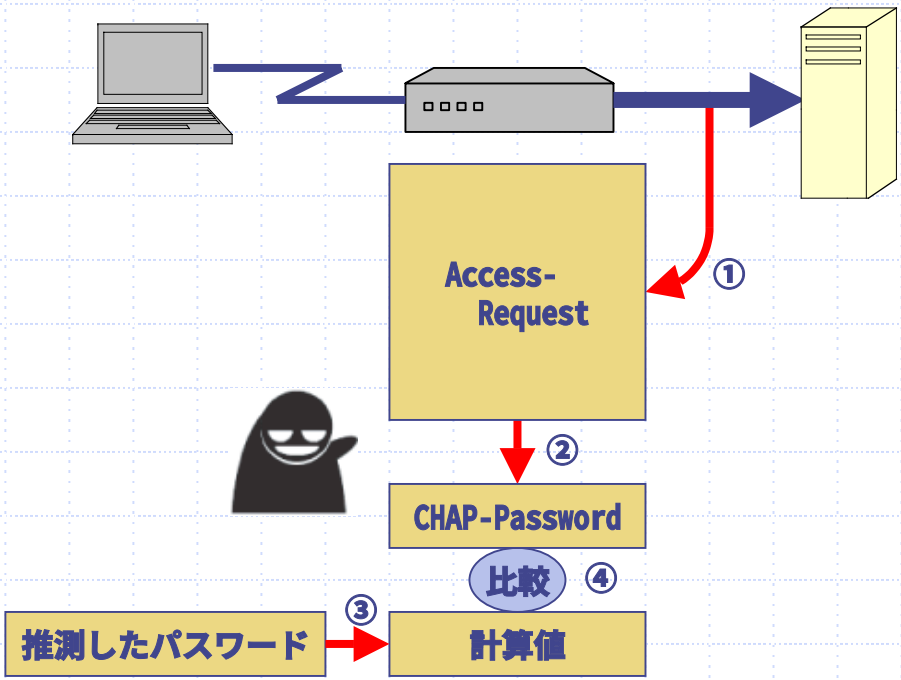
④ 推測する共有鍵とRequest Authenticatorからハッシュの計算値を求める

⑤ ③で得たハッシュ値と、④の計算値を比較する

④～⑤を繰り返すことで共有鍵の特定ができる
Offline Dictionary攻撃だけの場合より、計算する情報量が少ない分、多くの組み合わせを試すことができる

CHAP-Passwordの解読

CHAP認証を使用するAccess-Requestを傍受できる攻撃者は、Offline Dictionary攻撃を用いて、CHAP-Password属性の解読を行える



① CHAP-Password属性を含むAccess-Requestを傍受する

② CHAP-Password属性の値を取り出す

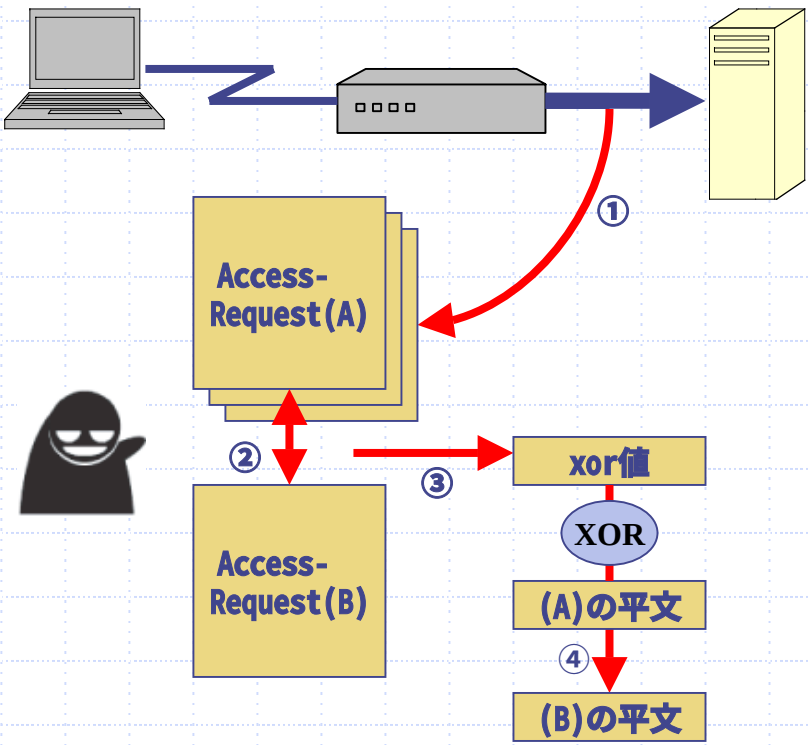
③ 攻撃者はCHAP-Passwordと同じ計算方法で、推測したパスワードを暗号化する

④ ②で得た値と、③の値を比較する

③～④を繰り返すことによりパスワードの解読が行われる
この場合の「解読し難さ」は、単にユーザのパスワードの複雑さのみに依存する

Request Authenticator重複

大量に傍受したAccess-Requestの中から、Request Authenticatorが重複しているものを探し出すことで、攻撃者はパスワードを推定するヒントを得たり、パスワードそのものを特定できる



① 攻撃者は、Access-Requestを大量に記録しておく

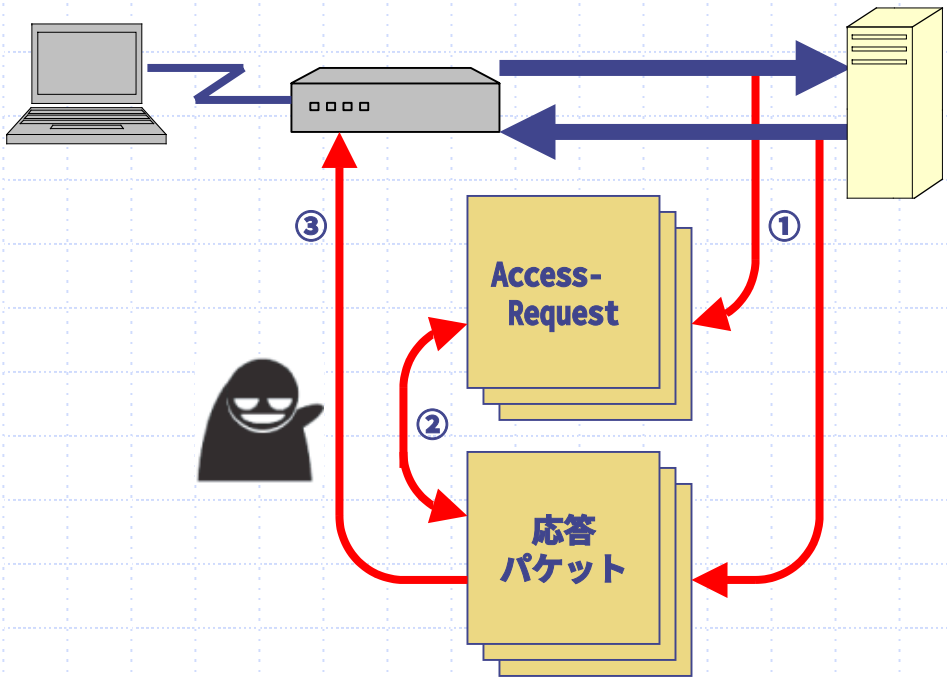
② Request Authenticatorが重複している組を見つけたら、双方のUser-Password属性の値をxorする

③ 傍受したAccess-Request中のUser-Password属性の値に既知の平文をxorすることで、ユーザパスワードの隠蔽の際に使用した、Request Authenticatorと共有鍵のMD5ハッシュ値を求める
得られる値は、2つのAccess-Requestで入力された、平文のパスワード同士のxor値になる

④ さらに2つのAccess-Requestのうち、どちらかが攻撃者が既知の平文パスワードで作成したものの場合、その平文を③の結果とxorすることで、もう片方のパスワードも平文に変換できる

Replay攻撃

攻撃者が、大量にRADIUSパケットを傍受して記録している場合、同じIdentifierとRequest Authenticatorを持つAccess-Requestに対して、サーバのフリをして応答できる



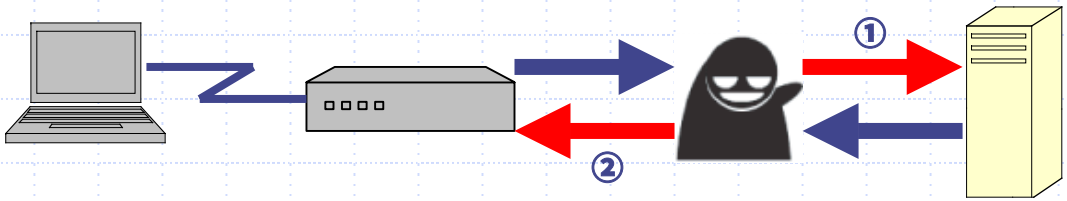
① 攻撃者はAccess-Requestと、それに対する応答パケットを大量に傍受、記録する

② 記録したAccess-Requestの中から、現在処理中のAccess-Requestと、同じIdentifierおよびRequest Authenticatorを持つものを探す

③ ②で見つかったAccess-Requestに対応する応答パケットを、ネットワークアクセス装置に送信する

Man in the Middle攻撃

RADIUSや一部のEAP認証などは、hop-to-hopモデルを採用しているので、Man in the Middle攻撃に耐性がない

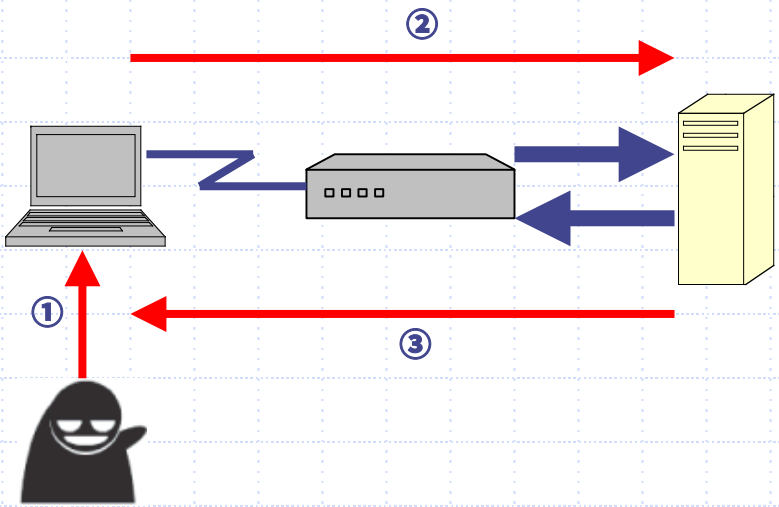


① プロキシとして振舞える攻撃者は、ネットワークアクセス装置からRADIUSサーバへ送られるパケットの、改竄、偽造、盗聴ができる

② RADIUSサーバからネットワークアクセス装置へのパケットも、同じように改竄、偽造、盗聴が行える

Negotiation攻撃

EAP認証を利用する場合、RADIUSサーバが適切に設定されていないと、攻撃者はより弱い認証方式を要求することで、強力な認証方式による安全性を無効にできる



① 正規のアカウントを保持する攻撃者は、PAP認証やCHAP認証、あるいはEAP認証でも安全性の低いEAP-MD5などを、接続時に要求する

② 要求した認証方式は、プロトコルの定め通りにネゴシエーションが行われる。しかし、適切に設定されているRADIUSサーバでない場合、より弱い認証方式の要求を受け入れてしまう

③ その結果、攻撃者は攻撃を行い易い状況（例えば、共有鍵のクラックなど）を作りだすことができる

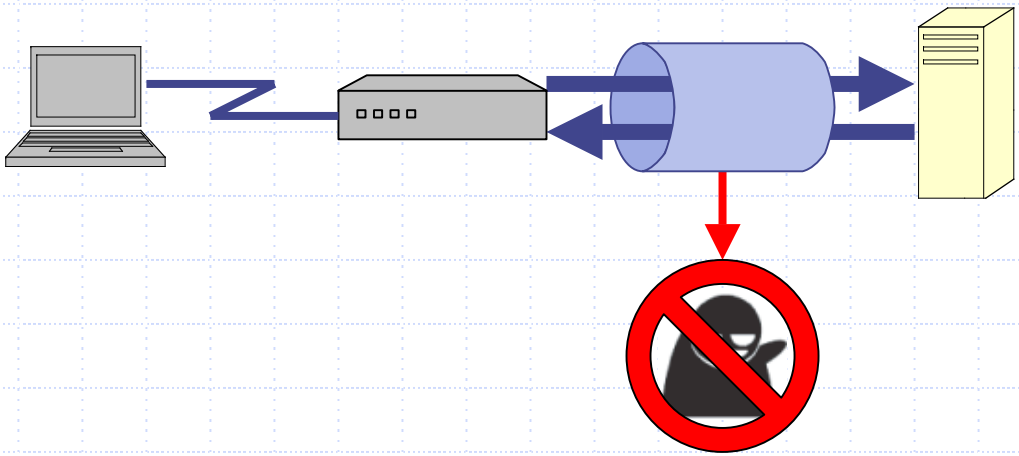
◆脆弱性への対策

- RADIUSのセキュリティの要は、共有鍵とAuthenticator
- EAPを正しく使う
- 伝送路レベルの対応も求められる

- IPSecの利用
- 共有鍵への対策
- 高品質なRequest Authenticator
- 強力なパスワードの使用
- Message-Authenticatorの利用
- EAPの利用

IPSecの利用

他のレガシーなプロトコルと同様RADIUSでも、伝送路にIPSecのような暗号化されたプロトコルを利用することが勧められている



伝送路が暗号化されることにより、攻撃者はパケットを傍受しても、内容を見ることができなくなる

共有鍵への対策

- 人間が覚えるものではないので、実装が許す限り、長く複雑な共有鍵を設定する
- 現在のプロセッサーの性能を考慮して、共有鍵の長さは、印字可能な文字ならば22文字以上にする
- 全ての文字の組み合わせを使用する
  可能なら、何らかのランダム文字列生成ツールを使って生成させるのが望ましい
- RADIUSサーバとクライアントの組ごとに、異なる共有鍵を使用する。定期的に変更する・・・などなど

高品質なRequest Authenticator

- Request Authenticatorは、16オクテットの乱数値で、Nonceとして使用する
- RADIUSの安全性は、Request Authenticatorの品質にも依存する
- 同じRequest Authenticatorが繰り返し使われないように、Request Authenticatorの生成には、質の良い乱数を使用する

強力なパスワードの使用

- CHAP認証においては、Access-Requestが傍受された場合の、ユーザパスワードの解読され難さは、単にユーザパスワードの複雑さに依存する
- 各ユーザに強力なパスワードの使用を求めることが重要

Message-Authenticatorの利用

- EAP-Message属性を含むRADIUSパケットだけでなく、全てのRADIUSパケットでMessage-Authenticator属性を利用する
- 特にAccess-Requestの整合性チェックに有効
- EAP-Message属性を含むRADIUSパケットでは必須だが、それ以外のパケットまで対応している実装は少ない

EAPの利用

EAPを利用することで、以下の安全対策の機能が利用できるようになる

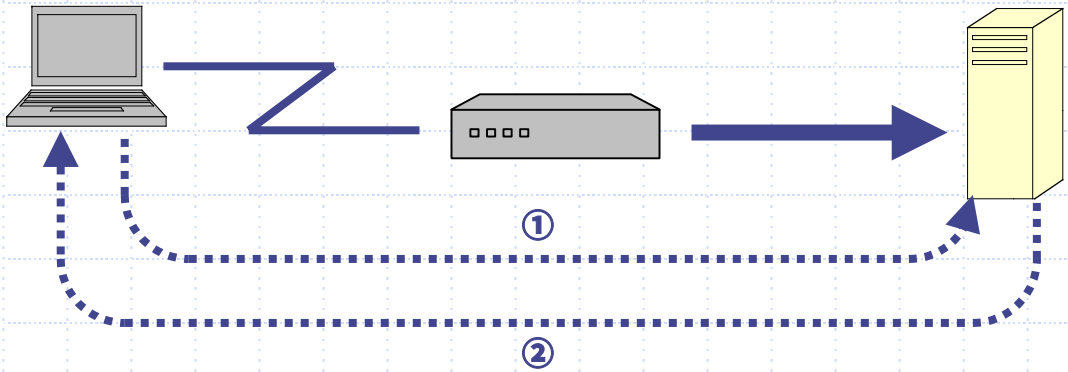
- 相互認証
- 認証情報の暗号化

しかし、以下の点に留意すべきである

- 認証方式の厳密な設定

相互認証

RADIUSサーバによるユーザ端末の認証だけではなく、ユーザ端末によるRADIUSサーバの認証も行う

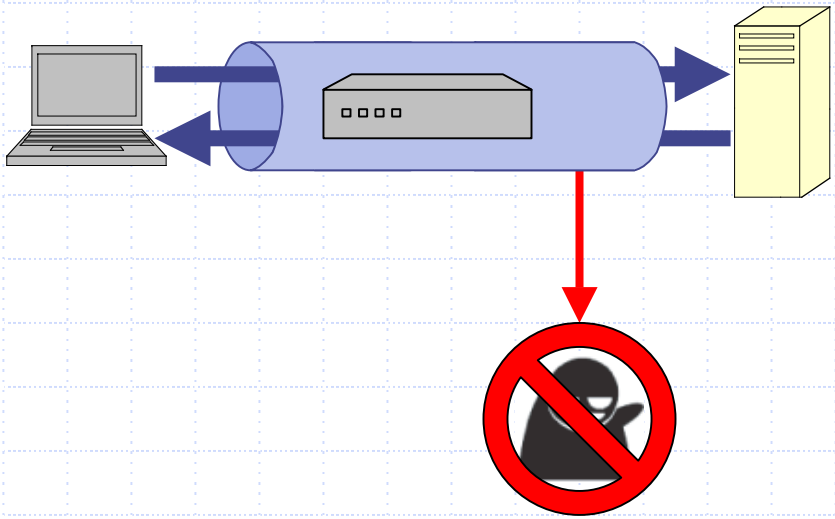


① RADIUSサーバによるユーザ端末の認証を行う。認証情報としては、クライアント証明書（EAP-TLSの場合）や、ユーザIDとパスワード（PEAPやEAP-TTLS）を使用する

② ユーザ端末によるRADIUSサーバの認証を行う。認証情報としてはサーバ証明書が主に使用される

認証情報の暗号化

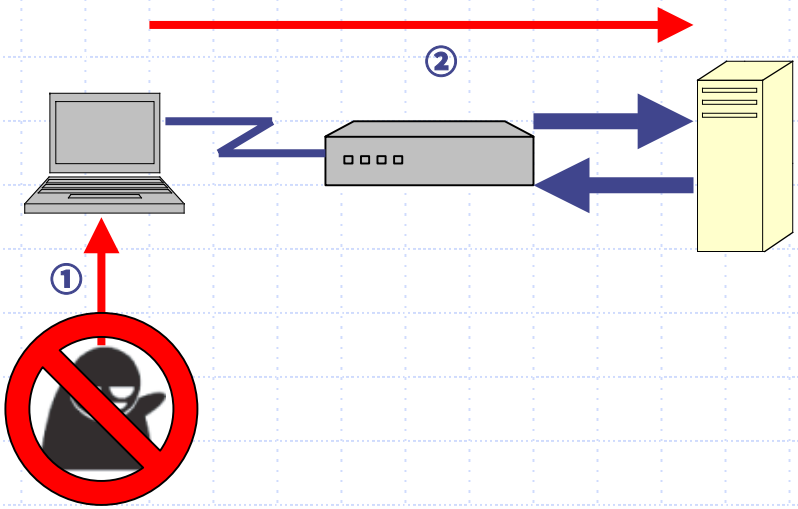
PEAPやEAP-TTLSでは、EAP-Message属性に含まれる認証情報は、暗号化される



end-to-endの認証情報の暗号化により、攻撃者はパケットを傍受しても、内容を見ることができなくなる

認証方式の厳密な設定

安全性の低い認証方式へのネゴシエーションが失敗するように、RADIUSサーバが適切に設定されていれば、Negotiation攻撃に対処できる



① 攻撃者が、Negotiation攻撃を試みる

② RADIUSサーバはより弱い認証方式を要求された場合、Access-Rejectで応答する

③ 結果、攻撃者の締め出しが行える

◆最後に・・・

- 柔軟な拡張性を持ち、様々な認証手順に対応できるRADIUSは、今後ますます利用シーンが増えるだろう
- だが、古いプロトコルであるが故の、数々の脆弱性も指摘されている
- DIAMETERのような代替プロトコルの提案もなされているものの、普及にはまだまだ時間が必要
- RADIUSの脆弱性を回避もしくは軽減するために、プロトコルの特徴を知り、より安全に使うことが重要である