

## ファイル拡張子

3文字のファイル拡張子は慣例であり、標準ではありません。また残念ながら、ファイル拡張子は証明書ファイル内で使用されるエンコーディングのタイプ、または証明書ファイルの内容にさえも常に正常にマッピングされるとは限りません。唯一の確実な確認方法は、テキストエディタで開くことになります。

- .DERと.PEMの両方ともファイル拡張子として使用できますが、後者はWindowsでは認識されません。.PEMは、LinuxのASCII形式ファイルの拡張子として最も広く使用されています。
- .CRTと.CERの拡張子も使用できますが、十分に標準化されていません。混乱のほとんどは、Windowsが証明書を処理する方法から発生します。Linuxでは.CRTがASCII証明書を表示する可能性が高いです。Windowsで最も一般的な拡張子は.CERですが、これでは、ファイル形式がバイナリなのかASCIIなのかが判断しかねます。

## 内容

証明書ファイルには、単一の証明書以外のものを含めることができます。

- **PKCS #12形式**により、秘密鍵と証明書のエクスポートが可能になります。これは、独自に鍵を生成できないホストに秘密鍵を転送したり、秘密鍵のバックアップ/アーカイブに使用できます。このタイプのファイル形式は通常パスワードで保護されており、常にバイナリになります。Windowsの場合、これには.PFX拡張子が使用され、MacOSとiOSでは.P12が使用されます。Linuxでは、証明書と鍵は通常個別のファイルに保存されます。
- **P7B形式**ではPKCS #7を実装します。これは、同じファイルで複数の証明書をバンドルする方法です。通常はASCII形式になります。これは、処理を行うホストによって信頼される必要がある証明書チェーンを提供するために最もよく使用されます。電子メールメッセージを暗号化するためのS/MIMEの使用に関連しています。P7Bファイルには秘密鍵は含まれません。Linuxの場合、証明書チェーンには.PEM拡張子が幅広く使用されています。

## OpenSSL

Windows環境の場合、証明書のインフラストラクチャはActive Directory証明書サービスとしてインストールされ、管理されます。コマンドライン管理用のcertutilツールがありますが、PowerShellも使用できます。

Linuxの場合、CAサービスは通常、OpenSSLサイト([openssl.org](https://openssl.org))を使用して実装されます。次は、opensslコマンドを使用して達成できる多くの操作の一部を示します。

### ルートCA

OpenSSLでルートCAを構成するには、ディレクトリ構成を設定して、OpenSSL構成ファイル(openssl.cnf)の内容を、サイトのローカル設定に適合させます。その後RSA鍵ペアを作成する必要があります。次のコマンドを使用します。

```
openssl genrsa -aes256 -out cakey.pem 4096
```

-aes256引数は鍵を暗号化し、それを使用するにはパスワードが必要になります。4096引数は鍵の長さを設定します。出力ファイルデータはデフォルトでPEM ASCII形式になります。サイトによっては.ca.keyのような命名規則が好まれる場合があります。

次の手順では、このRSA鍵ペアを使用して、自己署名済みルートX.509デジタル証明書を生成します。次のコマンドを使用します。

```
openssl req -config openssl.cnf -key cakey.pem -new -x509 -days 7300 -sha256 -out cacert.pem
```

 この例は簡易化しています。ルートCAを使用してリーフ証明書を直接発行するのは賢明ではありません。1つ以上の中間CAを作成するようにしてください。

## 証明書署名要求

ホストで証明書を構成するには、新しい鍵ペアで証明書署名要求(CSR)を作成します。このコマンドはWebサーバーで実行されます。

```
openssl req -nodes -new -newkey rsa:2048 -out
www.csr -keyout www.key
```

コマンドを実行したら、プロンプトに従い証明書に対象情報をすべて入力します。このとき共通名(CN)は、クライアントがサーバーにアクセスするFQDNと一致するようにしてください。ここで作成される鍵はパスワードなしで作成されます。パスワードを設定した場合はWebサーバーアプリケーションを再起動するときに入力する必要がある可能性があります。鍵の保護には、一般的なアクセス制御セキュリティ対策を使用できます。

このCSRファイルは、CAサーバーに送信される必要があります。CAで、次のコマンドを実行してCSRに署名をし、X.509証明書を出力します。

```
openssl ca -config openssl.cnf -extensions webserver
-infiles www.csr -out www.pem
```

`cakey.pem`秘密鍵の使用を確定するには、パスフレーズを入力する必要があります。`-extensions`引数で、特定の証明書の種類に対し構成ファイルのエリアを選択します。これにより、鍵使用法属性と、その他の必要な拡張属性が設定されます。

次の2つのコマンドを使用して新しい証明書を表示し、詳細を確認できます。

```
openssl x509 -noout -text -in www.pem
openssl verify -verbose -cafile cacert.pem www.pem
```

`www.pem`ファイルをWebサーバーに送信し、それと`www.key`秘密鍵を使用するようにサーバー構成を更新します。

## 鍵および認証管理

このサーバーから秘密鍵のコピーをエクスポートし、バックアップとしてエスクローに保管できます。この使用法では、鍵をパスワード保護する必要があります。

```
openssl rsa -aes256 -in www.key -out www.key.bak
```

Javaなどのアプリケーションサーバーと互換性を持たせるために、証明書のフォーマットを変換する必要があるかもしれません。次のコマンドでは、PEMでエンコードされた証明書を受け取り、DERバイナリでエンコードされた証明書を出力します。

```
openssl x509 -outform der -in www.pem -out www.der
```

他の使用例として、Windowsで使用するために鍵と証明書をエクスポートすることができます。

```
openssl pkcs12 -export -inkey www.key -in www.pem
-out www.pfx
```

## 証明書の問題

証明書を取り扱う際の最も一般的な問題は、クライアントでサーバー証明書を拒否されることです（もしくは少々一般的ではありませんが、認証サーバーでクライアントの証明書が拒否されることがあります）。

- 以前機能していた既存の証明書で問題が起きる場合は、証明書の有効期限が切れておらず、取り消されたり、一時停止されていないことを確認します。

- 新しい証明書に問題がある場合は、鍵使用法の設定がアプリケーションに適していることを確認します。VPNや電子メールクライアントなど、一部のクライアントには、鍵使用法の構成に非常に具体的な要件があります。また、対象名が正しく構成されており、クライアントで正しいアドレスが使用されていることを確認します。例えば、クライアントで、FQDNの代わりにIPアドレスでサーバーへの接続を試みる場合、FQDNで構成される証明書は拒否されます。
- 正しく構成されている新しい証明書のトラブルシューティングを行っている場合は、クライアントが適切な信頼の連鎖で構成されていることを確認します。リーフ証明書が信頼できるようになる前に、ルートCA証明書と中間CA証明書をクライアントにインストールする必要があります。一部のクライアントアプリケーションでは、OSのものとは異なる証明書ストアを維持している場合があるので注意してください。
- いずれの場合も、サーバーの日時設定とクライアントが同期されていることを確認します。間違った日時設定は、典型的な証明書の問題の原因です。

セキュリティの観点から、証明書のインフラストラクチャを監査して、有効な証明書のみが発行され、信頼されているようにします。発行済みの証明書のログは定期的に確認してください。証明書サービスの管理が割り当てられたユーザーの権限を確認してください。クライアントを確認して、有効なルートCA証明書のみが信頼されているようにします。クライアントが取り消された証明書や一時停止の証明書をチェックしていることを確認します。

# レビュー アク ティビティ： PKI管理

次の質問にお答えください。

1. 企業が秘密鍵の制御を失う場合に、何が起こると考えられますか。
2. あなたは、提供しているデータバックアップセキュリティとキーエスクローサービスの暗号化について顧客にアドバイスをしています。キーエスクローのリスクと、潜在的な緩和策について、どのように説明すべきですか。
3. クライアントに一時停止された鍵や取り消された鍵について通知するはどのメカニズムですか。
4. HPKPが実装するはどのメカニズムですか。
5. あるWindowsホストコンピューターから別のコンピューターに秘密鍵と証明書を転送する場合に使用できる証明書フォーマットの種類は何ですか。
6. 次のコマンドによって実行されるはどのような操作ですか。

```
openssl req -nodes -new -newkey rsa:2048 -out my.csr  
-keyout mykey.pem
```

# レッスン6

## 概要

さまざまな種類の証明書の発行と、PKIの操作の管理に使用されるツールや手順に精通してい  
る必要があります。

### 公開鍵インフラストラクチャ (PKI)を実装する際のガイドライン

プライベートネットワークに公開鍵インフラストラクチャ (PKI)を実装する際には、次のガイドライ  
ンに従います。

- 単一CAまたは中間CAを使った階層構造のどちらを使用するかを判断し、ルートCAのセキュ  
リティを確保するための措置を講じ、運用上可能な場合はオフラインのままにします。
- マシン、電子メール/ユーザー、コード署名証明書の種類などユーザーやビジネスのワー  
クフローの要件を満たす証明書ポリシーとテンプレートを決定します。証明書発行時に共通名  
属性が正しく構成されていることを確認します。
- 証明書を要求するユーザーとサーバー向けのポリシーと手順と、識別、認証、認可プロセ  
スを作成し、証明書が有効な対象にのみ発行されるようにします。
- 証明書をさまざまな形式に変換するオプションを用意し、ユーザーをサポートします。
- 鍵の取り消し、バックアップ/エスクローを含む、鍵と証明書の管理手順を設定します。
- 証明書のトラブルシューティングでユーザーをサポートできるようにします



# レッスン7

## 認証制御の実施

### レッスン概要

組織のアプリケーション、データ、サービスへのアクセスを制御するには、各ネットワークユーザーやホストデバイスは、アカウントを使って識別する必要があります。この要件をサポートするプロセスをアイデンティティとアクセス管理 (identity and access management、IAM) と呼びます。IAMの認証技術を使用することで、有効な主体（ユーザーまたはデバイス）だけがアカウントを操作できるようになります。認証の際、アカウント保持者は彼らだけが知り得るまたは保有し得る、アカウントにアクセスするための認証情報を提示する必要があります。認証技術は複数存在するため、これらのセキュリティ制御の違いを比較対照し、実装できなくてはなりません。

### レッスンの目的

このレッスンの内容は、以下のとおりです。

- 認証設計概念の要約。
- ナレッジベース認証の実装。
- 認証技術の実装。
- 生体認証概念の要約。

# トピック7A

## 認証設計概念を要約する



### 対象試験範囲

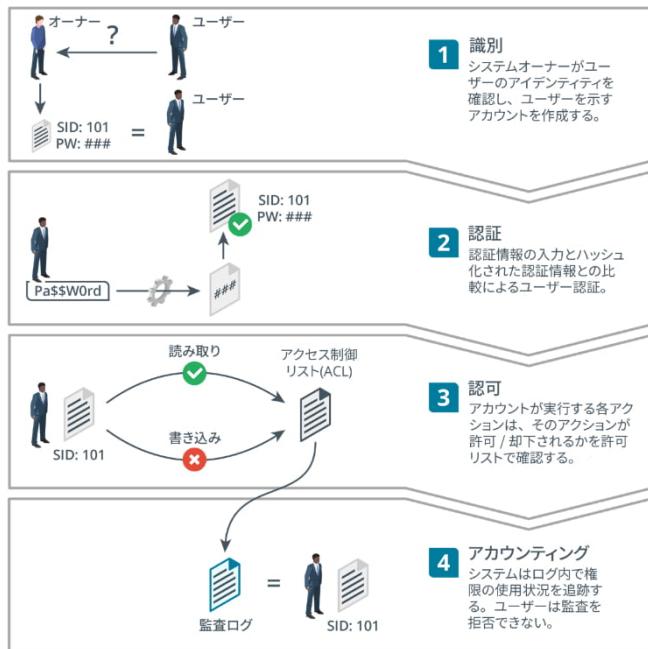
2.4認証と認可の設計コンセプトを要約することができる

ネットワークリソースを守るために戦いにおいて、強い認証は防御の第一線を担っています。しかし、認証プロセスは単一のプロセスではなく、さまざまな方法やメカニズムが存在します。そのうちのいくつかを組み合わせることで、さらに有効な成果を生むことができます。ネットワークセキュリティの専門家として、識別や認証技術に詳しくなると、環境に適したものを選択、実装、サポートする際にそれらの知識を役立たせることができます。

### アイデンティティとアクセス管理

アクセス制御システムとは、主体と対象の相互作用を管理する一連の技術的制御です。この場合の主体とは、リソースへのアクセスを要求したり許可されたりするユーザー、デバイス、ソフトウェアプロセスなどを指します。対象とは、ネットワークやサーバー、データベース、ファイルなどのリソースを指します。通常、**アイデンティティとアクセス管理(IAM)**は、次のように4つの主要プロセスの観点から説明されます。

- **識別** — ネットワーク上でユーザーや、デバイス、プロセスを一意に表すアカウントやIDを作成します。
- **認証** — ある主体がリソースにアクセスを試みるとき、その主体が誰であるか、または何であるかを証明することです。
- **認可** — 主体は各リソースに対してどのような権利をもっているべきかを判断し、それらの権利を実施します。
- **アカウンティング** — 主体による認可されたリソースや権利の使用を追跡し、認可されていない使用が検出または試みられた場合は警告します。



識別、認証、認可、アカウンティングの違い。(画像提供：© 123RF.com)

IAMでは、目的、機能、セキュリティ検査などのエンティティのアイデンティティを構成する属性を定義できます。これらの属性によって、アクセス管理システムは、エンティティによるアクセスを許可するか拒否するか、また許可する場合はそのエンティティが何をする権限があるかについて、情報に基づいた判断を行うことができます。例えば、個々の従業員はIAMシステムに自分自身のアイデンティティを持つ必要があります。社員がどの部署にいるか、マネージャーかどうかなど、会社での役割もアイデンティティの要因になります。例えば、制作中のイーコマースサイトにユーザーを登録する場合、各機能の実行に適した制御を選択する必要があります。

- **識別**—正規の顧客であることを確認します。例えば、請求先住所と配送先住所が一致しているかや、不正な支払方法を利用しようとしていないかなどを確認する必要があります。
- **認証**—顧客は一意のアカウントを所有し、注文や請求情報は本人だけが管理できるようにします。
- **認可**—有効な支払いメカニズムを設定した場合のみ注文できるようにするルールを設定します。特別な提案やコンテンツを閲覧する権限を特定の顧客に付与するロイヤルティーの仕組みやプロモーションを運営することもできます。
- **アカウンティング**—システムには、顧客がとった行動を記録しておく必要があります（例えば、注文したことを否定できないようにします）。

これらの機能を実装するサーバーやプロトコルを**認証、認可、アカウンティング(AAA)**と呼びます。識別フェーズの重要性に対する認識が高まっているため、IAMを用いて企業のプロセスやワークフローを説明するのがより一般的になってきています。

## 認証要素

アカウントが安全に作成されていると仮定して（アカウント保持者のアイデンティティは確認済みであることを意味します）、認証によって、アカウント保持者だけがそのアカウントを使用でき、システムはアカウント保持者だけが使用できることを検証します。アカウント保持者が適切な認証情報（または認証子やトークン）をシステムに提供すると、認証が実行されます。これらは、システムに保存されている認証情報と比較されます。2つが一致すると、アカウントが認証されます。

認証情報を定義する技術には様々な種類があり、要素で分類できます。

### Something You Know Authentication (知っていることによる認証)

一般的な知識要素は、ユーザー名とパスワードで構成されたログオンです。ユーザー名は通常は機密情報ではありませんが（公開するものではありません）、パスワードはアカウント保持者だけが知っているべきものです。パスフレーズとは、いくつかの単語で構成される長いパスワードです。これは安全性が高く、覚えやすいというメリットがあります。個人識別番号(PIN)も知っている情報に該当します。長いPINコードは覚えているのが困難ですが、短いと脆弱すぎて認証システムでは使用できません。多くの場合、タッチベースデバイスの認証には、スワイプパターンが使用されます。



Windowsのサインイン画面。（スクリーンショットはMicrosoftからの許可を得て使用。）

知識要素は、アカウントリセットメカニズムに使用されることもあります。例えば、アカウントのパスワードをリセットする場合に、ユーザーは「あなたのお気に入りの映画は？」といった秘密の質問に答えなくてはならないことがあります。

### Something You Have Authentication (持っているものによる認証)

所有要素とは、アカウント保持者が、固有の個人情報証明やアカウント番号がプログラムされたスマートカード、フォブ（キーホルダーサイズの器具）、リストバンドなど、他の誰も持っていないものを所有することを意味します。あるいは、一意のコードを生成するUSBフォブを所有している場合もあります。これらの所有者要素をハードトークンと呼びます。

スマートフォンなどのデバイスを使用して、ソフトトークンとして一意に生成されたアクセスコードを受け取ることもできます。これらのトークンは、パスワードと異なり、一般的に短い時間枠で一度だけ使用することができます。

## Something You Are/Do Authentication (身体的特徴/行動的特徴による認証)

生体認証要素には、指紋などの身体的識別子、または動き方（歩き方）といった行動識別子を使用します。識別子は、スキャンされ、テンプレートとして記録されます。ユーザーが認証を行うと、再度スキャンが実行され、テンプレートと比較されます。

## 認証設計

認証設計とは、機密性、完全性、可用性の要件に合った技術を選択することです。

- 機密性は、認証の観点から非常に重要です。アカウントの認証情報が漏れると、脅威アクターがアカウント保持者になりますし、持っている全ての権限を使ってシステム上で活動することができます。
- 完全性は、認証メカニズムが信頼でき、脅威アクターがバイパスしたり偽の認証情報で騙すことが容易でないことを意味します。
- 可用性は、認証に要する時間がワークフローを邪魔することではなく、ユーザーにとって操作が簡単であることを意味します。

認証はさまざまなコンテキストで使用されるため、要素が常にどのコンテキストにも適しているとは限りません。例えば、デバイスへのアクセス権を得るためにパスワードを入力してPCに対する認証を行う場合があります。これは、ネットワークに対する認証にも使用します。しかし認証は、物理的なセキュリティに対して用いられることもあります。出社した社員が、建物に入る際にパスワードを入力することを検討した場合、社員の人数が多いと、時間がかかるだけでなく大きな混乱が生じる可能性があります（可用性の欠如）。パスワードが知られてしまう可能性もあります（機密性の欠如）。挙句の果てに、ユーザーがお互いのためにドアを開けたままにしておく可能性もあります（完全性の欠如）。認証設計では、こういった問題を予測して、使用例に適した技術を実装するようにします。

## 多要素認証

認証技術は、知識、所有、生体認証要素から2種類以上を組み合わせると強力になると考えられており、これを**多要素認証(MFA)**と呼びます。単一要素認証システムは、パスワードが書き留められたり共有される、スマートカードを紛失したり盗まれる、生体認証システムのエラー率が高くなったりスプーフィング攻撃を受けるなど、非常に簡単に侵害されてしまう可能性があります。

二要素認証(2FA)では、所有権ベースのスマートカードまたは生体認証識別子を、パスワードやPINなどの「ユーザーが知っている情報」と組み合わせます。三要素認証では、3つの技術全てを組み合わせたり、位置情報などの追加属性を組み込んだりします。例としては、指紋リーダーを一体化したスマートカードが挙げられます。つまり認証を行う場合、ユーザーはスマートカードを所持している必要があり、ユーザーの指紋はスマートカードに保存されているテンプレートと一致している必要があります、ユーザーは正しいPINまたはパスワードを入力する必要があります。



多要素認証では、いくつかの異なる技術を組み合わせる必要があります。たとえば、PINとともに生年月日の入力を要求した場合、PINを単独で入力するより強力ですが、それだけでは多要素ではありません。

## 認証属性

3つの主な認証要素と比較する場合、認証属性とは、一意でない特性か単独では使用できない要素のことです。

### Somewhere You Are Authentication (場所に基づく認証)

位置情報に基づく認証では、ユーザーのいる場所に関する統計を測定します。これは、デバイスの位置サービスを使って測定するか、IPアドレスによって測定する地理的位置です。デバイスのIPアドレスを、論理ネットワークセグメントを参照するために使用したり、位置情報サービスを使って地理的位置とリンクさせることもできます。構内ネットワークでは、物理ポートの位置、仮想LAN (VLAN)、またはWi-Fiネットワークも位置情報に基づく認証の基礎となり得ます。

位置情報に基づく認証は、主要な認証要素としては使用しませんが、継続的認証メカニズムまたはアクセス制御機能として使用する可能性があります。たとえば、ユーザーがVPNゲートウェイに正しい認証情報を入力したときに、ユーザー IPアドレスが予想とは異なる国にユーザーがいると表示する場合、アクセス制御によって、付与された特権が制限されたり、アクセスが完全に拒否される場合があります。別の例として、移動が物理的に不可能な地理的場所からユーザーがログインしているように見える場合があります。

### Something You Can Do Authentication (行動的特徴による認証)

歩き方やスマートフォンの持ち方などの行動上の特徴によって、かなりの確率で一意に人物を特定することができます。この要素を主要な認証に使用するのは実用的ではありませんが、デバイスが所有者によって継続的に操作されていることを確認するためのコンテキスト認証や継続的認証で使用することができます。

### Something You Exhibit Authentication (提示するものによる認証)

提示するものとは、個人の特徴を特に強調することで行動に基づいて認証や認可を行うことです。例えば、スマートフォンアプリやウェブ検索エンジンの使い方が、統計テンプレートのような機械学習分析によって取得可能な行動パターンと一致する場合があります。もし他人がデバイスを使用した場合、彼らは所有者とは異なる行動をとるため、この変位パターンは、デバイスをロックしたり、再認証を要求したりするために使用できます。

### Someone You Know Authentication (知っている人による認証)

「信頼の網(Web of trust)」モデルを使用しており、新しいユーザーは既存のユーザーによって保証されます。ユーザーがネットワークに参加すると、ユーザーのアイデンティティがより良く確立されます。その一例として、Pretty Good Privacy (PGP)をPKIの代わりに使用する「信頼の網」の分散化モデルが挙げられます([weboftrust.info/index.html](http://weboftrust.info/index.html))。

# レビュー アク ティビティ：

## 認証設計概念

次の質問にお答えください。

1. 認可と認証の違いは何ですか。
2. 新しい従業員をドメインネットワークに登録する場合、どんなステップをとる必要がありますか？
3. 次の記述は正しいですか、誤りですか？パスワード、PIN、スマートカードが必要なアカウントは、三要素認証の例です。
4. 位置情報に基づく認証を実施する場合、どんな方法を用いますか？

# トピック7B

## ナレッジベース認証を実装する



### 対象試験範囲

- 1.2 与えられたシナリオに基づいて、可能性あるインジケーターを分析して攻撃のタイプを特定することができる
- 3.8 与えられたシナリオに基づいて、認証と認可のソリューションを導入することができる
- 4.1 与えられたシナリオに基づいて、適切なツールを利用して組織のセキュリティを評価することができる（パスワードクラッカーのみ）

ナレッジベース認証とは主に、パスワードベースのアカウントアクセスメカニズムをユーザーに発行することです。パスワードベースの認証プロトコルを設定し、認証に関する問題を抱えるユーザーをサポートすることは、情報セキュリティの役割の重要な部分です。このトピックでは、一般的な認証プロトコルがどのように機能するかや、パスワードクラッキング技術によってプロトコルがリスクにさらされる過程について学びます。

### ローカル認証、ネットワーク認証、リモート認証

オペレーティングシステムの最も重要な機能の1つは、認証プロバイダです。これは、シェルを起動する前にユーザーが認証される仕組みを支えるソフトウェア・アーキテクチャとコードのことです。これは通常、ログイン(Linux)、ログオン、サインイン(Microsoft)などと呼ばれます。パスワードや個人識別番号(PIN)を使用するナレッジベース認証は、多くのオペレーティングシステムのデフォルト認証プロバイダーです。

ナレッジベース認証は暗号学的ハッシュに依存します。攻撃を受けるリスクがあるため、認証情報データベースに対して平文でのパスワードの転送や保存は通常行いません。代わりに、暗号学的ハッシュとしてパスワードを保存します。ユーザーがログインのためにパスワードを入力すると、オーセンティケーター(authenticator)は入力された内容をハッシュに変換し、それをオーソリティ(authority)に送信します。オーソリティは、送信されたハッシュとデータベース内のハッシュを比較し、一致した場合のみ対象者を認証します。

### Windows認証

Windows認証には、複雑なアーキテクチャのコンポーネントが関係しますが([docs.microsoft.com/en-us/windows-server/security/windows-authentication/credentials-processes-in-windows-authentication](https://docs.microsoft.com/en-us/windows-server/security/windows-authentication/credentials-processes-in-windows-authentication))、次の3つのシナリオが一般的です。

- Windowsローカルサインイン — ローカルセキュリティオーソリティ(LSA, Local Security Authority)は、転送された認証情報と、レジストリの一部であるセキュリティアカウントマネージャー(SAM, Security Account Manager)データベースに保存されているハッシュを比較します。これは、インタラクティブログオンとも呼ばれます。
- Windowsネットワークサインイン — LSAは、認証情報をネットワークサービスに渡します。ネットワーク認証の優先システムはKerberosに基づきますが、レガシーネットワークアプリケーションでは**NT LAN Manager (NTLM)**認証を使用する場合があります。
- リモートサインイン — ユーザーのデバイスがローカルネットワークに接続されていない場合、認証は、仮想プライベートネットワーク(VPN)またはウェブポータルを介して行うことができます。

## Linux認証

Linuxでは、ローカルユーザーのアカウント名が /etc/passwd に保存されます。ユーザーがローカルインタラクティブシェルにログインすると、/etc/shadow に保存されたハッシュに対してパスワードを確認します。ネットワークを介したインタラクティブログインは通常、Secure Shell (SSH) を使って実施されます。SSH の場合、ユーザーは認証にパスワードではなく暗号化鍵を使用します。

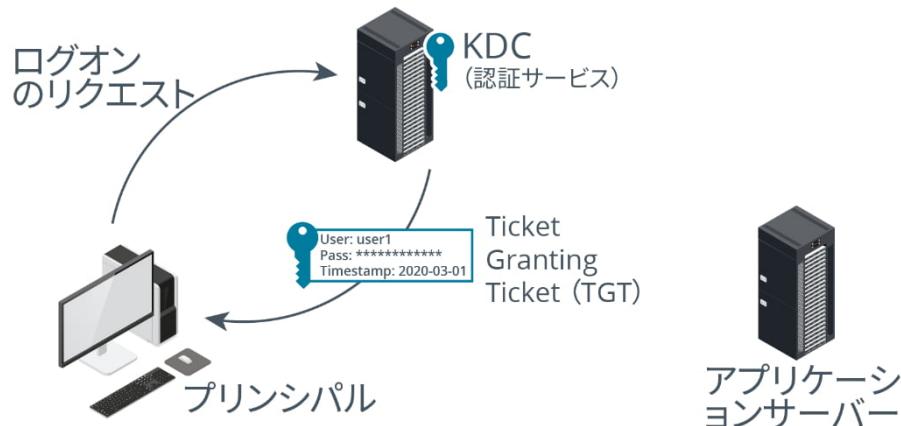
**プラグ可能な認証モジュール(PAM, Pluggable Authentication Module)**は、スマートカードログインのような、別の認証プロバイダーを有効にするパッケージです([tecmint.com/configure-pam-in-centos-ubuntu-linux](https://tecmint.com/configure-pam-in-centos-ubuntu-linux))。PAMフレームワークは、ネットワークサーバーに認証を実装する際に使用できます。

## シングルサインオン(SSO)

**シングルサインオン(SSO)**システムを使用した場合、ユーザーはローカルデバイスに対して一度だけ認証を行うだけで、認証情報を再度入力することなく互換性のあるアプリケーションサーバーに認証されます。Windowsでは、KerberosフレームワークによってSSOが提供されます。

## Kerberos認証

**Kerberos**は、MicrosoftのActive Directory (AD)サービスによる実装をはじめとする、多くのネットワークで使用されるシングルサインオンネットワーク認証および認可プロトコルです。Kerberosは、3つのパーツから構成されていることから、3つの頭を持つ番犬（ケルベロス）にちなんで命名されました。クライアントはアプリケーションサーバーにサービスを要求する際に、その両者が依存する**KDC (Key Distribution Center、鍵配布センター)**という仲介者によって、クライアントのアイデンティティが保証されます。KDCは、認証サービス(AS, Authentication Service)とチケット発行サービス(TGS, Ticket Granting Service)の2つのサービスで構成されます。KDCは、TCPまたはUDPを使ってポート88で動作します。



Kerberos認証サービス。(画像提供：© 123RF.com)

認証サービスは、ユーザーログオンリクエストの認証に対する責任を担います。より一般的には、より一般的には、ユーザーとサービスを認証することができます。これらを総称してプリンシパル(principal)と呼びます。例えば、Windowsドメインワークステーションに座ってrealm (realm、ドメインや領域) にログオンするとき、ログオンの最初のステップは、ドメインコントローラーとして実装されているKDCサーバーで認証することです。

1. クライアントは AS に **TGT (Ticket Granting Ticket)** の要求を送信します。これは、ユーザーのパスワードハッシュを鍵として、ローカルコンピュータの日付と時刻を暗号化することによって構成されます。



パスワードハッシュ自体はネットワークを介して転送されません。簡単にするために、パスワードに言及しましたが、システムはスマートカードによるログオンなど他の認証プロバイダーを使用することができます。



*Ticket Granting Ticket (TGT)*またはユーザー チケット) は、タイムスタンプ済みです (Windows では、最大10時間がデフォルトです)。これは、ネットワーク上のワークステーションやサーバーが同期されている必要があります (5分以内)、でなければチケットが拒否されることを意味しています。これはリプレイ攻撃を防ぐのに有益です。

2. ASは、ユーザー アカウントが存在しており、ユーザーのパスワードハッシュがActive Directoryデータベースのパスワードハッシュと一致することによってリクエストが復号されること、そのリクエストの有効期限が切れていないことを確認します。リクエストが有効な場合、ASは次のデータで応答します。
  - TGT — これにはクライアント情報 (名前とIPアドレス)、タイムスタンプ、有効期限が含まれます。また、KDCの秘密鍵を使用して暗号化されます。
  - TGSセッション鍵 — クライアントとTGS間の通信に使用されます。この鍵は、ユーザー パスワードのハッシュを使って暗号化されます。

TGTは、論理トークンの一例です。TGTが行うことは、ユーザーが誰なのかを特定し、認証済みかを確認することですが、これを行うことでドメインリソースへのアクセス権が提供されるわけではありません。

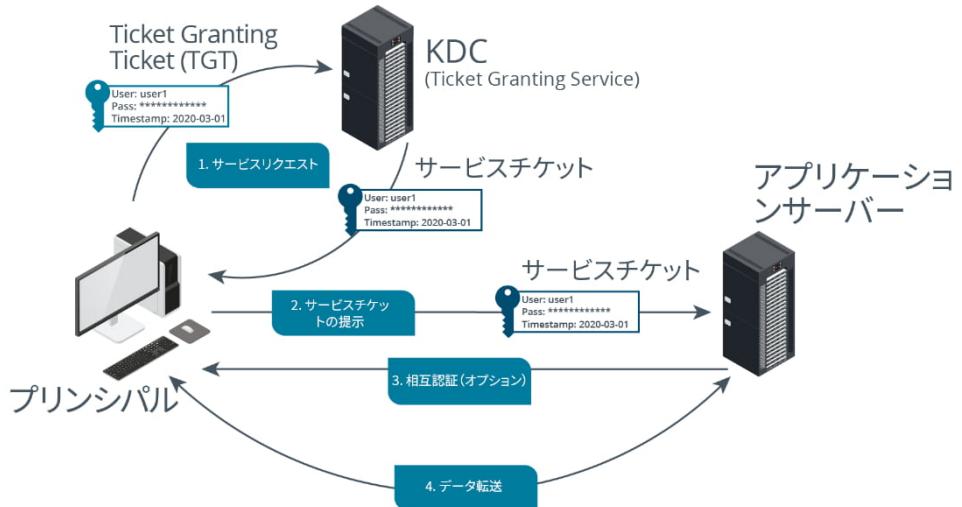
## Kerberos認可

ユーザーによって入力されたパスワードを正しいと仮定すると、クライアントはTGSセッション鍵を復号できますが、TGTについてはそうではありません。これにより、クライアントとKDCは同じ共有の秘密鍵を知ることになりますが、クライアントはTGTに干渉できません。

1. ドメイン内のリソースにアクセスする場合、クライアントはサービスチケット (ターゲット アプリケーションサーバーへのアクセス権を許可するトークン) をリクエストします。このサービスチケットを許可するプロセスは、TGSによって処理されます。
2. クライアントはTGSに、TGTのコピーと、アクセスしたいアプリケーションサーバーの名前、さらにTGSのセッション鍵で暗号化されたタイムスタンプ付きのクライアントIDからなる認証子(authenticator)を送信します。

TGSは、1番目のメッセージにはKDCの秘密鍵を、2番目のメッセージにはTGSのセッション鍵を使用して、両方のメッセージを復号できるはずです。これによってリクエストが本物であることを確認できます。また、チケットが有効期限切れでなく、以前に使用されていないことを確認します (リプレイ攻撃)。

3. TGSサービスは次のものを使って応答します。
  - サービスセッション鍵 — クライアントとアプリケーションサーバー間で使用されます。これは、TGSセッション鍵を使って暗号化されます。
  - サービスチケット — タイムスタンプ、システムIPアドレス、セキュリティ識別子 (SID)、ユーザーが所属するグループのSID、サービスセッション鍵など、ユーザーに関する情報が含まれます。これは、アプリケーションサーバーの秘密鍵を使用して暗号化されます。
4. クライアントは、サービスチケット (これは復号できない) をアプリケーションサーバーに転送し、サービスセッション鍵を使用して暗号化する別のタイムスタンプ済みの認証子 (authenticator)を追加します。



Kerberos Ticket Granting Service (画像提供: © 123RF.com)

5. アプリケーションサーバーはサービスチケットを復号し、その秘密鍵を使ってサービスセッションを入手し、クライアントが改ざんされていないメッセージを送ったことを確認します。次に、サービスセッション鍵を使って認証子を復号します。
6. オプションとしてアプリケーションサーバーは、サービスセッション鍵を使って暗号化された認証子の中で使用したタイムスタンプで、クライアントに応答します。クライアントはタイムスタンプを復号し、既に送った値と一致していることを確認し、アプリケーションサーバーが信頼できると結論付けます。

これで、サーバーはクライアントに対して認証されたことになります（相互認証と呼ばれます）。これによって、悪意のあるユーザーがクライアントとサーバー間の通信を傍受する中間者攻撃を防ぐことができます。

7. 次にサーバーは、（サーバーのアクセス制御リストに一致していると仮定して）クライアントのリクエストに応答します。



データ転送自体は暗号化されません（少なくともKerberosの一部としては。ある種の転送暗号化は使用できます。）

Kerberosの良く知られている欠点の一つは、KDCがネットワークの単一障害点を提示する点です。実際には、バックアップKDCサーバーを実装できます（例えば、Active Directoryは、KDCサービスを実行している複数のドメインコントローラーをサポートします）。

## PAP認証、CHAP認証、MS-CHAP認証

Kerberosは、信頼できるローカルネットワーク上で動作するよう設計されています。リモートアクセスプロトコルと共に機能する認証プロトコルがいくつも開発されており、シリアルリンクまたは仮想プライベートネットワーク(VPN)を介して接続されています。

### パスワード認証プロトコル(PAP)

**パスワード認証プロトコル(PAP)**は、Point-to-Pointプロトコルの一部として開発された洗練されていない認証方式であり、シリアル接続またはダイヤルアップ接続を介してTCP/IPデータを転送する際に使用します。また、HTTPの基本認証メカニズムとしても使用されます。しかし、平文パスワード交換に依存しているため、暗号化されたトンネルを介した場合を除いてほとんど使われていません。

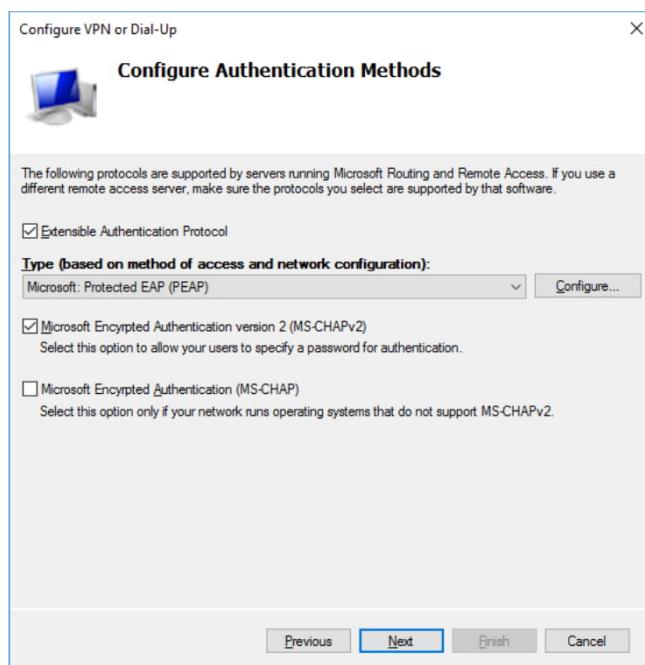
## チャレンジハンドシェイク認証プロトコル(Challenge Handshake Authentication Protocol、CHAP)

チャレンジハンドシェイク認証プロトコル(CHAP)もまた、リモートリンクを介してユーザーを認証する手段として、PPPの一部として開発されました。CHAPは、3方向ハンドシェイクと呼ばれるシステムの暗号化チャレンジに依存しています。

1. チャレンジサーバーは、ランダムに生成されたチャレンジメッセージを送ってクライアントにチャレンジします。
2. レスポンスークライアントは、サーバーチャレンジメッセージとクライアントパスワード（または他の共有の秘密鍵）から計算したハッシュを使って応答します。
3. 検証サーバーは、クライアントのために保存されているパスワードハッシュを使って自身でもハッシュを実行します。これが応答と一致するとアクセスが許可され、一致しないと接続を切断します。

接続中には繰り返しさまざまなチャレンジメッセージでハンドシェイクが実行されます（ユーザーには分かりません）。これは、以前のセッションをキャプチャして再利用することによってアクセスするリプレイ攻撃から保護するものです。

**MS-CHAPv2**は、Microsoftに実装されているCHAPです。脆弱なNTLMハッシュを使用する方法を採用しているため、渡される認証情報が暗号化されるように安全な接続トンネルが保護されていない場合は、MS-CHAPを展開すべきではありません。



Windows VPN上で可能な認証メカニズムを定義します。  
(スクリーンショットはMicrosoftからの許可を得て使用。)

## パスワード攻撃

ユーザーがパスワードを選択すると、このパスワードは、MD5またはSHAなどの暗号機能を使ってハッシュに変換されます。つまり理論上では、ハッシュから平文を復号できないため、ユーザー以外に（システム管理者でさえ）このパスワードを知る人はいないということです。

### 平文/非暗号化攻撃

平文/非暗号化攻撃は、暗号化を用いないパスワードストレージまたは認証プロトコルを悪用します。この例には、PAP、基本HTTP/FTP認証、Telnetなどが含まれます。これらのプロトコルは使用すべきではありません。パスワードは、管理されていないファイルに保存してはいけません。認証情報侵害の一般的な原因のひとつに、パスワードをアプリケーションコードに組み込んで、その後パブリックリポジトリにアップロードしてしまう場合があります。

### オンライン攻撃

オンラインパスワード攻撃では、脅威アクターが、例えばウェブログインフォームやVPNゲートウェイなどの認証サービスと直接やり取りします。脅威アクターは、既知のパスワードデータベース（とその変形）や、オフラインでクラックしたパスワードリストを使ってパスワードを送信します。



また、インターネットに保存された複数のアカウントに、ユーザー名とパスワード/パスワードハッシュの組み合わせのデータベースが存在することにご留意ください。これらの詳細情報は様々な企業システムのハッキング成功事例から抽出しています。これらのデータベースの検索には[haveibeenpwned.com](http://haveibeenpwned.com)のようなサイトを使用します。

繰り返しログオンに失敗した後でログオンに成功したり、通常とは異なる時刻または場所からログオンが試みられ成功する場合、オンラインパスワード攻撃が監査ログに表示される可能性があります。ユーザーが強いパスワードを使用するだけでなく、ログオンを試みる回数や速度を制限したり、既知の不正なIPアドレスから試みるログオン試行を回避することによって、オンラインパスワード攻撃を緩和することができます。



ログオンを制限すると、アカウントがサービス拒否攻撃にさらされて脆弱になる可能性があるのをご注意ください。脅威アクターは、認証を繰り返すことによって有効なユーザーをロックアウトします。

### パスワードスプレー攻撃

**パスワードスプレー攻撃**とは、水平方向のブルートフォースオンライン攻撃です。これは、脅威アクターが一つまたはそれ以上の共通パスワード（例えば、passwordまたは123456）を選択し、複数のユーザー名に関連付けて試すことを意味します。

### オフライン攻撃

オフライン攻撃では、脅威アクターが、%SystemRoot%\System32\config\SAM、%SystemRoot%\NTDS.DIT（Active Directory認証情報ストア）、または/etc/shadowのようなパスワードハッシュのデータベースの入手を試みます。パスワード・データベースを入手した後は、クラッカーは認証システムとやり取りすることはありません。この種の攻撃の唯一のインジケーターは、（攻撃が成功した際のアカウントの誤用以外に）悪意のあるアカウントがこれらのファイルにアクセスしたことを記録するファイルシステム監査ログです。脅威アクターもホストメモリから認証情報を読むことができますが、その場合の信頼できる唯一のインジケータはホスト上の攻撃ツールの存在でしょう。

脅威アクターがパスワードデータベースを入手できない場合、NTLMまたはCHAP/MS-CHAPなどのプロトコルのサーバーチャレンジに対するクライアントの応答を入手するために、パケットスニッファが使われる可能性があります。これらのプロトコルでは、パスワードのハッシュを直接送るのを回避し、何らかの方法でパスワードのハッシュからレスポンスを導き出します。パスワードクラッカーはプロトコルの脆弱性を悪用してハッシュを計算したり、辞書の単語と一致させたり、ブルートフォース攻撃を行います。

## ブルートフォース攻撃と辞書攻撃

パスワード攻撃には、ユーザーが選択した弱い認証情報を悪用するものがあります。その他のパスワード攻撃では、ストレージメカニズムの脆弱性を悪用するものもあります。たとえば、Windows SAM データベースは、古いバージョン（LM およびNTLMv1ハッシュ）との互換性のためのハッシュを保存するように構成することができます。これらのレガシーハッシュは暗号法的に脆弱で、パスワードクラッキングの攻撃を受ける可能性が非常に高いのです。

### ブルートフォース攻撃

**ブルートフォース攻撃**は、取得したハッシュと、それを生成した平文を推測するために、両者が一致する出力スペースでの可能な組み合わせを試行します。出力スペースは、アルゴリズムが使用するビット数によって決定されます（例えば、MD5は128ビット、SHA-256は256ビット）。出力スペースを大きくしたり、平文のパスワードに使用された文字数が増加すると、可能性のある各ハッシュ値を計算したりテストして一致を発見するのがさらに難しくなります。ブルートフォース攻撃は、時間やコンピューティングリソースによる制約が厳しいため、短いパスワードのクラックで大きな効果を発揮します。しかしブルートフォース攻撃は、ハイエンドグラフィックカードのクラスターのように、複数のハードウェアコンポーネントに分配すれば、長いパスワードのクラックにも成功する可能性があります。

### 辞書攻撃とレインボーテーブル攻撃

**辞書攻撃**は、複雑でないパスワードなど、もっともらしい値の平文を推測するチャンスがある場合に使用できます。ソフトウェアは平文の辞書からハッシュ値を生成し、取得したハッシュとのマッチングを試みます。**レインボーテーブル**攻撃は、辞書攻撃のアプローチを改良したもので、脅威アクターは、すべての可能なパスワードとそれに対応するハッシュの、あらかじめ計算されたルックアップテーブルを使用します。ただし、あまりにも多くのメモリが要求されるので、可能性のあるハッシュ値がすべて保存されるわけではありません。値は連鎖的に計算されますが、保存が必要なのは最初と最後の値だけです。保存しているパスワードのハッシュ値がテーブル内で見つかると、対応する平文が分かります。

ソルトを使って保存された平文にランダム値を追加すると、レインボーテーブル攻撃の速度を遅くすることができます。なぜならテーブルを事前に生成することができず、パスワードとソルト値の組み合わせごとに生成し直す必要があるからです。またレインボーテーブルは、長いパスワード（約14文字超）の発見を試みる際も実用的ではありません。UNIXとLinuxのパスワード保存メカニズムではソルト値を使用しますが、Windowsでは使用しません。したがってWindows環境では、強いパスワードポリシーを実施することがさらに重要です。

### ハイブリッド攻撃

**ハイブリッドパスワード攻撃**では、辞書攻撃とブルートフォース攻撃を組み合わせて使用します。この攻撃では主に、`james1`のような複雑さが不十分で単純なパスワードがターゲットになります。パスワードクラッキングのアルゴリズムは、辞書に載っている単語や名前を、数字の接頭辞や接尾辞を加えるなどして、テストするバリエーション数を制限するマスクと組み合わせてテストします。他のタイプのアルゴリズムは、複雑なパスワードの選択を強要されたときに、覚える苦労を避けたいユーザーがどのような行動をとるかについての、ハッカーの知識に基づいて適用します。他の例として、「s」を「5」、または「o」を「0」で代用することなどが含まれます。