◇◇◇◇◇◇
解　説
◇◇◇◇◇◇

情報システムにおける利用者認証方式の動向について

林　豊洋[1]

1　概要

ほとんどの情報システムでは，利用開始の最初の段階において「利用者認証」が行われます．これは，学内の情報システムである講義室の端末，e-Learning システム，無線 LAN アクセスポイントへの接続などに留まらず，商用の情報システムである EC サイトやネットバンキングでも同様です．大半のシステムは,「ユーザ ID とパスワードによる認証」を採用しています．この方法は古くから存在し，単純ではあるものの，セキュリティと利便性のバランスが取れた妥当な方法として用いられています．

世の中には，おおよそ 20〜30 年前では想像できないほどの多数の情報システムが存在します．即ち，利用者認証を求めるシステムで溢れており，同様に沢山の利用者認証方式が存在しているということです．利用者認証方式は，利用者側の要望や管理者側の要望に応えるため，多数の方式が提案され実用化されました．以前に比べて，現在の認証方式は多機能化・利便性向上・複雑化しています．利用者から見ると，単にユーザ ID とパスワードを入力する従前の方式に見えるものでも，その裏側は様々な新しい方式で動いています．本稿では,「利用者認証方式が時代と共にどのように変遷・進歩してきたのか」に注目し，解説を進めます．

2　利用者認証方式の変遷・進歩

本稿では，利用者認証方式について以下の流れで解説します．

1. システムごとに独自に認証する方式
2. 主に LAN 内のシステムで利用者情報を統合し，認証する方式 (NIS，NT ドメイン認証等)
3. 安全性，安定性，多様性等を考慮したより多機能な統合認証方式 (LDAP，ActiveDirectory 認証等)
4. 異なる認証方式，認証サーバの統合 (アイデンティティ管理，プロビジョニング)
5. ログイン処理の省力化，認証と認可の分離 (シングルサインオン，ソーシャル ID 連携方式 (OpenID, OAuth)，フェデレーション方式 (Shibboleth, OpenAM))
6. シングルサインオン + 情報システムに応じた認証強度，認証方式切り替え，多層化 (多要素認証)
7. 利用者の振舞の判定に基づく認証方式の自動切り替え (リスクベース認証)

以降にて，これらの認証方式がなぜ提案されたのか，またその利便性等について解説を行います．

[1]情報科学センター　助教　toyohiro@isc.kyutech.ac.jp

2.1 システムごとに独自に認証する方式

計算機が高価な時代は，一台の計算機を複数の利用者が共有して利用する形態が一般的でした．この当時は，対象とする計算機に利用者情報 (アカウントと対応するパスワード) を登録・管理していました．利用者がログイン処理を行う際に，対象とする計算機上で独自に認証および認可の処理が行われます．この方式では，複数の計算機が導入された場合，それぞれに利用者情報を登録・更新・削除等を行う必要が生じるため，計算機が増えるごとに管理者の負担が増大します．利用者から見た場合においても，「ある計算機のパスワードを変更しても，別の計算機では変更されない」等の状況が生まれるため，利便性が高いとは言えません[2]．

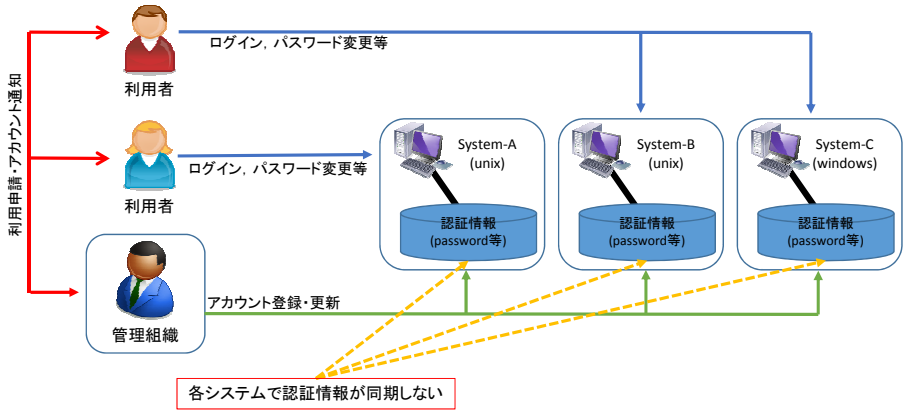

図 1: システムごとに独自に認証する方式

2.2 1980 年代〜1990 年代 : 主に LAN 内のシステムで利用者情報を統合し，認証する方式

学術機関においては，1980〜1990 年代に UNIX ワークステーション (Sun SPARCstation 等，EWS) の導入が進みました．本学においても，数十台のワークステーションと，数百台の利用者クライアント (X 端末) を LAN によって接続し，教育システム環境として活用されました．企業においては，PC の性能向上や PC 向け OS のネットワーク対応 (Microsoft Windows NT 等) により，PC ベースのサーバ，クライアントが積極的に用いられ，オフィス向けの計算機環境が実現されました．

このように，LAN 内に利用者がログイン可能な計算機が複数存在する状況において，以前のように利用者情報を計算機ごとに管理・認証を行った場合，前述の通り

- 計算機が増えるごとに，利用者情報を登録・管理する対象が増大する
- パスワード変更等を行うと利用者情報の不一致が生じ，利便性が低下する

等の問題が生じるため，合理的でありません．したがって，LAN 内のワークステーション (サーバ) とクライアントによる計算機環境に対して，「利用者情報の統合化・認証」を実現するシステムが考えられ，組み合わせて利用されました．

この当時よく用いられたシステムとして，UNIX 環境向けの NIS(Network Information System，Sun microsystems 社が開発)[1] と，Microsoft Windows 向けの NT ドメイン認証 (Microsoft 社が開発)[2] が挙げられます．いずれも LAN 内に利用者情報を一元的に管理する認証サーバを配置し，クライアントは

[2]一般家庭においては利用面・管理面の負担への意識は低く，現在でも「複数の PC を所有しているが，利用者情報はそれぞれ別々である」という場合が大半ではないかと思います．

ログイン処理時に認証サーバと通信を行い，利用の可否を判定します．利用者情報が認証サーバ上に配置されているため，利用者情報の登録・管理が集中化され，管理者・利用者双方にとって有用な方式といえます．

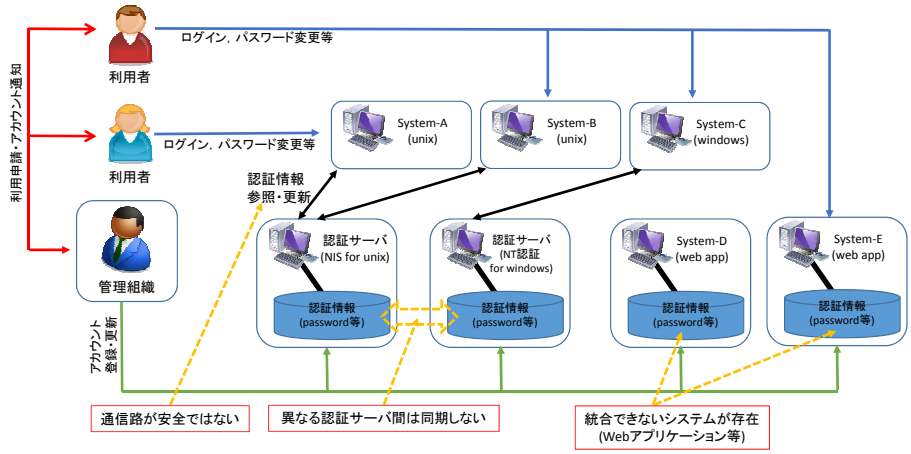

図 2: LAN 内での利用者情報統合

2.3 2000 年代初頭：安全性，安定性，多様性等を考慮したより多機能な統合認証方式

2000 年代に入ると，計算機の低価格化や情報システムへの依存がさらに進み，より多くのサーバやクライアント (PC) がネットワークに接続され，利用されるようになりました．また，これまでクライアント側のアプリケーションとして構築されていた情報システムの利用者機能が，サーバサイドで動作する形式に変わり，Web ブラウザ経由で利用するシステムが増えました[3]．加えて 2000 年代に入ると，これまで比較的希薄であった「情報セキュリティ」のリスクが高まり，不正アクセスを考慮したシステムが求められるようになりました[4]．

このような背景より，LAN での利用を前提とした認証システム (NIS や NT ドメイン認証) では，以下のような機能の不足が生じることとなりました．

LAN の範囲を超える規模での利用に適さない 大規模なネットワークでの利用を想定していないため，クライアントの台数が膨大な場合や，(認証サーバから見て) 外部ネットワークからの応答性が低い等の問題がありました．また，認証データの暗号化レベルが低く，重要な情報が漏えいする危険性がありました．

利用者の属性情報を拡張できない 何れの方式も，ベンダーの独自プロトコルであり，「認証情報を自由に拡張する」といったことは考慮されていませんでした．したがって，想定する計算機システム以外の認証には適さない仕組みでした．

[3] 例として，本学情報工学部で運用されていた教務情報システムは，第一世代システムのクライアントは UNIX 上のアプリケーションとして実装され，UNIX 専用システムでした．第二世代は Java アプリケーションとして再設計され，OS を問わずに利用可能となりました．現在の第三世代のシステムは Web ブラウザ経由で利用する形式となり，クライアント OS との依存性はさらに低くなりました．

[4] 1990 年代まではいわゆる「牧歌的」な時代であり，パスワードの平文送信や telnet による遠隔アクセス，ファイアウォール未設置などが散見される時代でした．

Web システム等との認証連携が煩雑 上述した問題と関連しますが，Web システム等の認証システムとして用いようとした場合，仕組みが複雑であり，対応できる OS が限られるといった問題がありました[5].

これらの問題に対応するためには，認証システムが「外部ネットワークからの認証要求に対応」「サーバ・クライアント間のセキュアな通信に対応」「認証に関わる基礎的情報は共通の構造を持ち，自由に拡張可能」「多様なシステムからの認証要求に対応」することが求められます．このような多くの要求に対応できるシステムとして，通信網上で階層構造を用いたデータ表現である「ディレクトリ」にアクセスする規格である，X.500(DAP) が注目されました [3]．X.500 自体にはインターネット上で使用されにくい欠点があり，これを改良した仕組みとして LDAP(最新は RFC4511，[4]) が規定されました (図 3).

LDAP は，階層構造を用いて利用者情報を管理します．LDAP サーバは，決まったルール (スキーマ) に基づきデータを管理するための階層構造 (ディレクトリ) を作成します．ある階層下に利用者ごとの認証に関わる情報 (ログイン ID やパスワード) を保存することにより，利用者の認証情報が一元管理できます．LDAP サーバ同士でデータの複製を行うことにより，可用性を高めることが可能です．サーバ・クライアント間の接続は SSL により保護可能であり，TCP/IP を用いたアクセスを前提としているため，外部ネットワークからの安全な認証が可能です．

また，LDAP は認証の仕組みがシンプルであるため，クライアントの構築が容易に行えます．多くのプログラミング言語が LDAP の認証ライブラリを有しており，Web アプリケーション等の利用者認証にも容易に適用できます．

加えて，LDAP はスキーマを作成することにより，柔軟に属性情報の拡張が可能です．Windows 環境の認証システムとして利用されている Active Directory(AD)[5] は，LDAP の属性を拡張した概念として構築されています．

NIS や NT ドメイン認証から LDAP/AD への置き換えが進み，利用者認証方式の利便性は大きく向上しました[6].

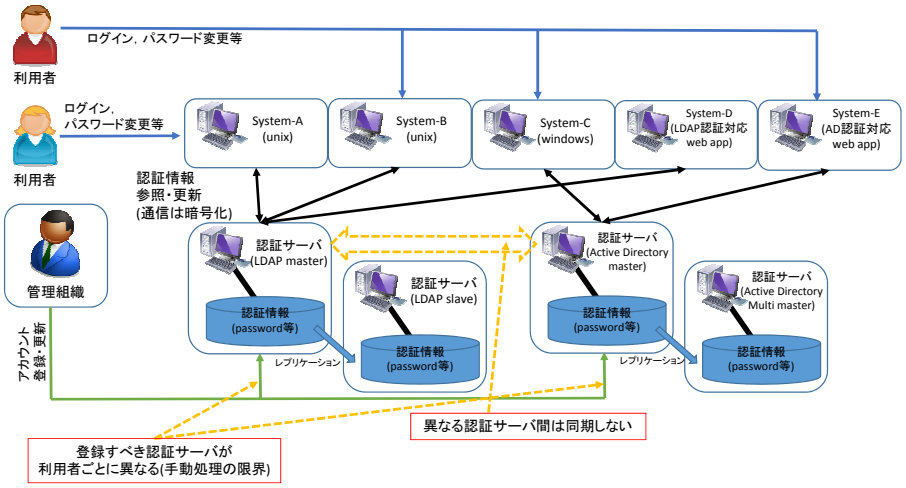

図 3: 安全性，安定性，多様性等を考慮した統合認証方式

[5]Web サーバ単体では認証連携できず，サーバを稼働させている UNIX サーバに認証を任せる仕組み (Apache+外部認証パッチ+PAM 認証) が必要でした．

[6]本センターの教育システムの利用者認証方式も，2000 年に導入されたシステムから LDAP に切り替りました

2.4 2000 年代後半：アイデンティティ管理による異なる認証方式，認証サーバの統合

LDAP や AD 等の認証システムにより，LAN 内外を問わず，多くの計算機を統合認証の対象とすることが可能になりました．本学においても，各学科システムや業務システム毎に LDAP や AD 向けの認証サーバが設置され，各システムの利用者認証を行っています．このように多くの認証サーバが組織内に存在する時代になると，新たな管理上の問題が顕在化しました．それは,「ライフサイクルに基づく利用者情報の管理が複雑化しつつある」というものです．

例として，以下の 3 種類の情報システム

1. LDAP，学生のみを登録，アカウント名は学籍番号，パスワードは英小文字・大文字 8 桁
2. AD, 教職員を登録，アカウント名はランダム英字，パスワードは英小文字・数字 8 桁
3. LDAP，学生・教職員を登録，アカウント名は氏名のローマ字表記，パスワードは上記 1 と共通 (連動)

が存在したと仮定します．

このような環境に対して新規に利用者を追加する場合，以下の手順を要します．

- 利用者が学生であるか，教職員であるかを判定
- 各システムに対応したアカウント名，パスワードを生成
- 対応する情報システムにアカウントを追加
- アカウント名，パスワード等を記したアカウント通知書を発行

また，利用者の異動・卒業・退職などのライフサイクルに応じて，定期的に以下の管理を要します．

- 属性の変更を要するアカウントの抽出
- 対象のアカウント情報の変更箇所を抽出し，更新すべき値を生成
- 対応する情報システムのアカウント情報を更新

利用者がパスワードを変更する場合，例示した環境ではパスワードが連動したシステムが存在します．したがって，各情報システムのパスワード情報をルールに基づき同期する手順を要します．加えて，特に企業においては，情報システムのセキュリティレベルを高く保つため，各システムが独自に「テスト用」等を目的として作成したアカウントを定期的に削除する処理が求められます．

例示の環境では 3 つの情報システム向けの認証サーバが存在しますが，実際にはもっと多数が存在するため，更に複雑なものとなります．したがって，現在の情報システムに対して，管理者が定期的に手作業でアカウント管理を行うことは現実的ではない，と言えます．

このような問題点を解消するため，2000 年代の後半に「アイデンティティ管理 [6]」と呼ばれる概念を持つシステムが登場しました (図 4)．アイデンティティ管理システムでは，LDAP や AD などの認証サーバ (下位システム) の上位に，組織に所属する利用者情報を一元管理するシステム (ID 管理システム) を配置します．

ID 管理システムは，それぞれの利用者情報に基づき，登録すべき認証サーバの抽出，認証サーバに応じた属性情報の生成を行い，対応する下位システムに利用者情報の登録を行います．この処理は「プロビジョニング」と呼ばれます．利用者情報の変更や削除についても，ID 管理システム上の操作や定期的な自動処理により，下位システムの利用者情報を更新します．

また ID 管理システムは，利用者向けの Web インタフェースを有することが大半であり，利用者自身が情報 (氏名やパスワード) を変更できます．Web インタフェースでパスワードを変更することにより，対応する複数の認証サーバ内のパスワードを変更することが可能です．

このようなアイデンティティ管理による ID 管理システムの登場により，認証方式 (LDAP，AD 等) や複数の認証サーバについて，一元的な利用者情報管理が可能となりました．本学においても，2009 年度に第一世代のシステム (Sun Identity Manager) が導入され，2014 年度に第二世代のシステム (富士通 UnifiedOne) に更新されました[7]．

[7]多くのシステムは有償のパッケージですが，OpenIDM[7] 等のオープンソースによる実装も存在します．

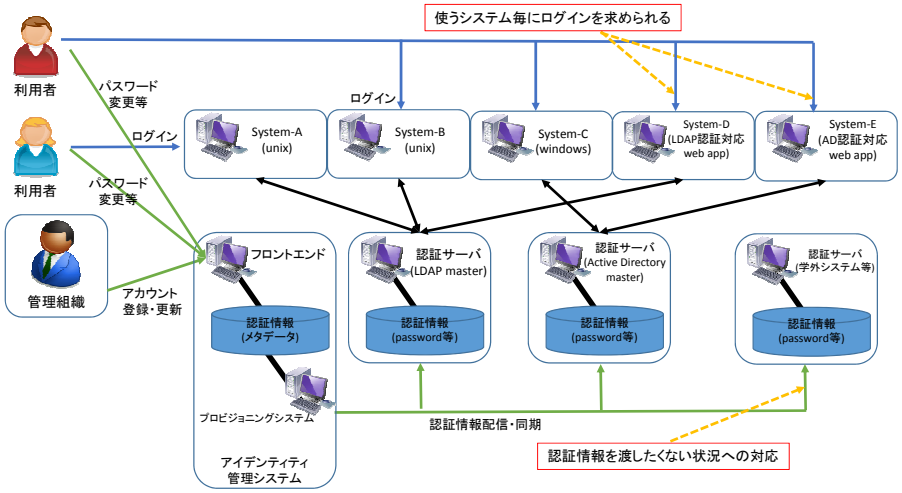

図 4: アイデンティティ管理による異なる認証方式，認証サーバの統合

2.5 2000 年代終盤〜2010 年代初頭：ログイン処理の省力化，認証と認可の分離 (シングルサインオン)

アイデンティティ管理の導入により，組織内の情報システムの利用者情報は一元的に管理可能となりました．利用者認証システムとして不足のない構成ですが，利用者・管理者の双方の視点で，更なる要求が出てきました．

利用者：なぜ何度もログインが必要なのか 通常のシステム構成では，システムを使う際，利用者認証を行い認可 (ログインに成功) された後に利用可能となります．同じアカウント情報を用いる他のシステムを使おうとした際，再び利用者認証が必要です．利用者の視点では，何度もログイン操作を行わずに多くのシステムを利用したいという要求があります．

管理者：全ての利用者情報を渡したくない情報システムが存在する 通常のシステム構成では，情報システムは利用者認証を行うため，指定された認証サーバと連携しています．これは，情報システムが，認証サーバ内の情報に自由にアクセス可能であることを意味します．認証サーバと情報システムが異なる機関によって管理されている場合，セキュリティ上の問題となります．

例えば，利用者認証を行い，本学以外の情報システムを利用する場合を考えます．本学の管理外の情報システムに学内の認証サーバへの連携を許可した場合，「学外のシステムが本学の情報を取得できる」ことになります．これは，情報漏えい等のリスク要因となります．管理者の視点では，認証サーバと連携させず，利用認可が行える仕組みが欲しいという要求があります．

これらの要求に対応する方法として，利用者認証の仕組みを「認証手続きに特化したシステム (IdP：Identity Provider，アイデンティティプロバイダ)」と「idp から送信された情報に基づき利用認可を判定し，サービスを提供するシステム (SP：Service Provider，サービスプロバイダ)」に分離する仕組みが提案されました．このような認証と認可を分離した仕組みは「シングルサインオン (SSO)」呼ばれます (図 5)．
シングルサインオンの仕組みは多岐に渡りますが，現在では以下の 2 方式が主に利用されます．

1. 利用者がソーシャル ID に基づく IdP を選び，SP と連携する方式 (本稿ではソーシャル ID 連携方式と呼びます)
2. IdP と SP が信頼関係に基づく連携システムとして構築される方式 (本稿ではフェデレーション方式と呼びます)

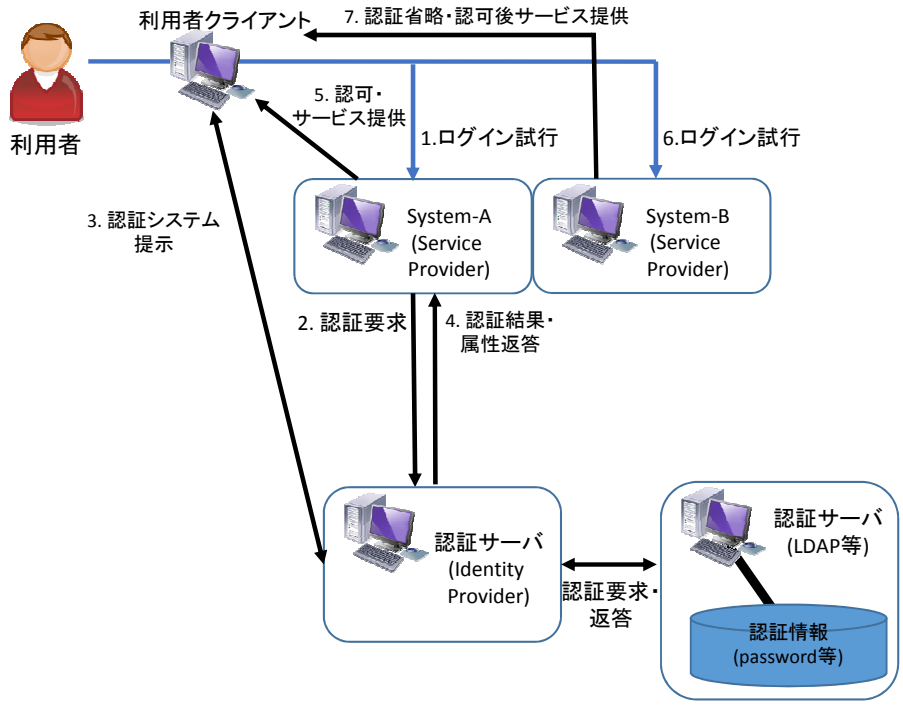

図 5: シングルサインオンによるログイン処理の省力化，認証と認可の分離

ソーシャル ID 連携方式　近年，多くの人が facebook や Twitter に代表されるソーシャルネットワーキングシステムや，Yahoo!や Google 等のポータルサービス (ソーシャルサービス) を利用しています．これらのサービスを利用する際には，発行されたアカウント情報 (ソーシャル ID) による利用者認証が必要です．ソーシャル ID の発行時には，SMS や電子メールを利用した最低限の本人確認が行われます．したがって，ソーシャル ID は「多くの人が利用している」かつ「ある程度の水準の本人確認がなされている」アカウント情報となります．重要な情報を扱わないシステムに対象を限定すれば，ソーシャル ID による利用者認証は妥当である，と言えます．

このような，ソーシャル ID と連携したシングルサインオンの方式として，OpenID，OAuth，OpenID connect が普及しています [8, 9]．いずれの方式も，ソーシャル ID により利用者認証を行うサーバ (IdP) と，ソーシャル ID 連携方式による利用認可に対応した SP に機能を分離し，サインオンを行います．

SP を利用する際，**利用者自身が連携したい IdP を選択し，シングルサインオンの対象とする**点が本方式の特徴です (図 6)．連携した SP に対して，単一のアカウント情報で利用できるため，シングルサインオンが実現できます．

IdP から SP に送信される利用者に関する情報は，各方式によって異なります．OpenID では，「IdP が規定した利用者情報」のみが送信されます．利用者の氏名やメールアドレス等が対象となり，利用者に関する限定的な情報が送信されます．対して，OAuth では，認証が成功した場合，利用者情報の送信に加えて，ソーシャルサービス側で利用者が保持するコンテンツ (facebook や twitter への書き込み等) にアクセス可能となります．SP が利用者に関する多くの情報を取得できるため，利用者は「本当に連携して問題ないか」を判断することが重要です．この問題に対応するため，OAuth を安全に使うための拡張 (コンテンツへのアクセス範囲を限定化できる) を行った OpenID connect が開発されました．現在，多くの IdP が OpenID connect への移行を進めています．

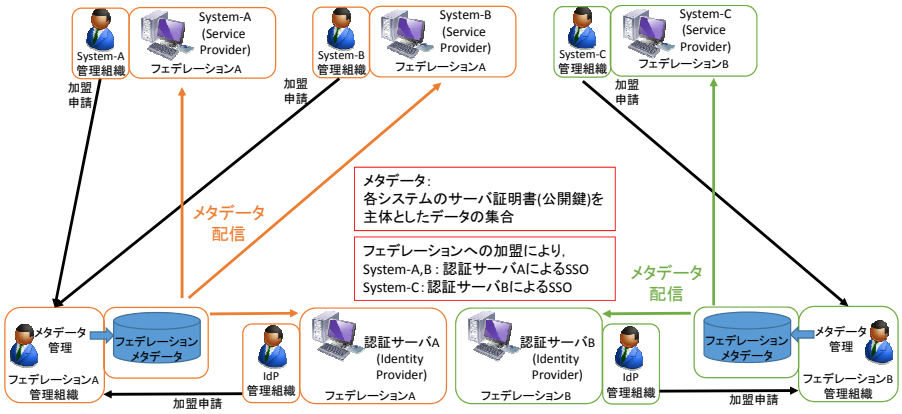

図 6: ソーシャル ID 連携方式によるシングルサインオン

フェデレーション方式　ソーシャル ID 連携方式は，SP と連携する IdP を利用者自身が選択する方式でした．この方式は利用者にとっての利便性は高いと言えますが，連携する SP を範囲を限定化できないとも言えます．

したがって,「IdP と SP 同士が信頼関係を結び (フェデレーション)，その範囲内においてシングルサインオンが可能」というフェデレーション方式が存在します．フェデレーション方式として，Shibboleth と OpenAM が普及しています [10, 11, 12].

いずれの方式も，ソーシャル ID 連携方式と同様に，利用者認証を行う IdP と利用認可を行う SP に機能を分離し，サインオンを行う方式です．ただし，前述のソーシャル ID 連携方式と大きく異なる点は，**信頼関係の管理組織が存在する**ことです．管理組織は加入機関の IdP と SP に関する電子証明書を収集し，メタデータを作成します．共通のメタデータを所有する IdP と SP のみがシングルサインオン可能であり，連携する範囲が限定化できます (図 7).

現在，主に学術機関を中心としたフェデレーションが世界中に存在します．国内においては，国立情報学研究所が管理を行う学術認証フェデレーション (学認 [13]) が存在し，Shibboleth によるフェデレーションを構成しています．学認には，国立大学の半数・国立高専機構の全てが IdP 機関として加入しています．また，電子ジャーナルサービスを中心に，127 の SP が加入しています．本学も平成 22 年に IdP 機関として学認への加入を行いました．学認の仕組みも含めた詳細については，文献 [14] にて解説しています．

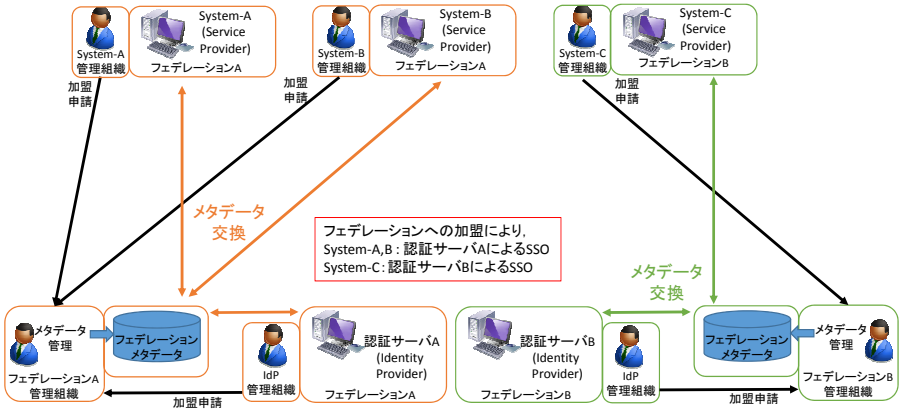

図 7: フェデレーション方式によるシングルサインオン

2.6 2010 年代：シングルサインオン + 情報システムに応じた認証強度，認証方式切り替え，多層化

シングルサインオンの導入により，利用者は「一度ログインを行えば，多くのシステムが利用できる」ことになり，管理者にとっては「認証と認可を分離でき，セキュリティの観点で好ましい」環境が整いました．しかし，「全ての情報システムへのアクセスについて，同じアカウント情報で認証される」ということが，必ずしも良いとは言えない状況があります．

例として，Web メールサービスを提供する SP と，商品購入や購入履歴の参照が可能な EC システムを提供する SP が，シングルサインオン可能な環境を想定します．この環境には単一のアカウント情報でサインオンできるため，「A: Web メールを利用する」，「B: 商品の購入履歴を参照する」，「C: 実際に商品を購入する」という行為が，一度の認証で行えます．しかし，前者二つ (A,B) と後者 (C) では，システムが利用者に与える影響が大きく異なります．即ち，C は「商品の購入」という金銭の動きが発生するため，仮に利用者情報が漏えいした場合，利用者の知らない間に商品を購入される等の不正利用のリスクが高まります．

この問題に対応するためには，情報システム毎に求める認証の強度を設定し，許可された認証強度を超えたシステムへのアクセス時には，新たに別の認証を求める仕組みが必要となります．このような背景から，「**シングルサインオンの利便性を保ちつつ**，必要に応じて認証方式が変化する・追加される」仕組みである「二段階認証，多要素認証」が検討されました (図 8，9)．

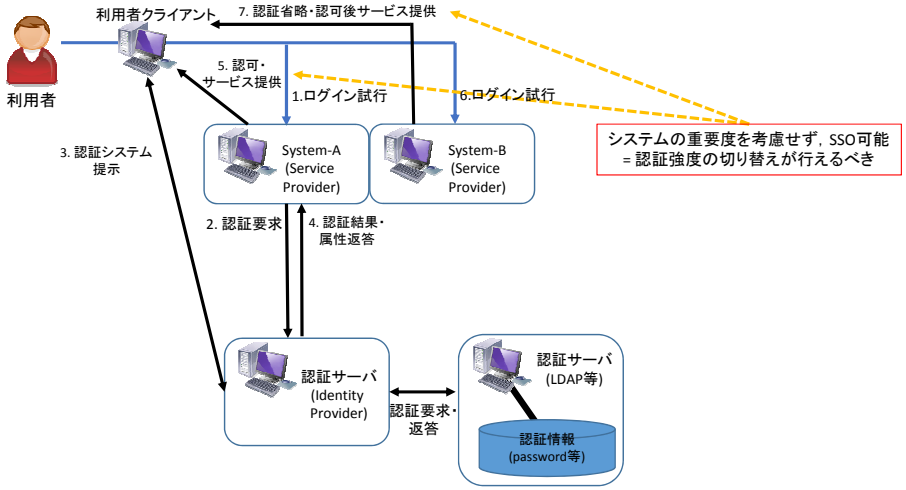

図 8: シングルサインオンの問題点

二段階認証・多要素認証の枠組みでは，IdP の利用者認証機構ごとに，その機構が保証する認証強度が設定されます．また，SP 毎に，利用時に求める認証強度が設定されます．利用者が得た認証強度を超えた SP を利用する場合，SP は IdP に追加の認証を要求します．これを「二段階認証」と呼びます．

IdP は SP からの要求に応じて，より高強度の認証機構を利用者へ提示し，利用者はその認証を通過する必要があります．このように，認証強度毎に異なる認証機構を用意する方式を，「多要素認証」と呼びます．

ID とパスワードによる認証よりも高強度の認証機構としては，「クライアント証明書認証 (住基ネットカード等に保存した電子証明書)」「静脈や手書きサイン等の生体認証」があります[8]．

[8]以前は指紋認証や虹彩認証が生体認証として用いられていましたが，これらの方式は生体情報の複製が容易であるため，用いられることは少なくなりました．

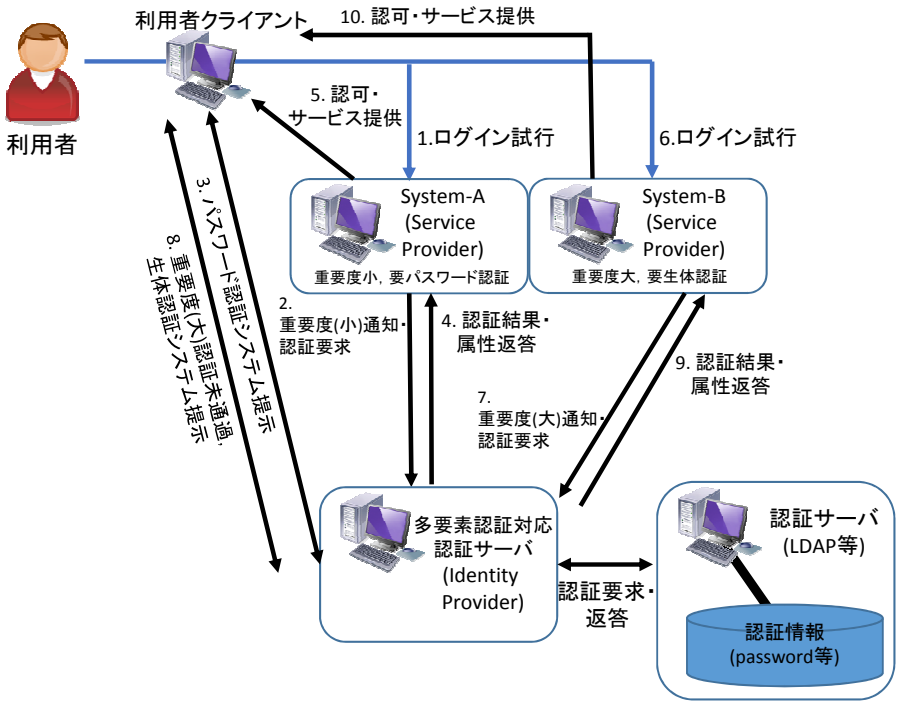

図 9: 認証強度を考慮したシングルサインオン

現在，フェデレーション方式である OpenAM が多要素認証に対応しています．Shibboleth においても，現在多要素認証モジュールの開発が進められています．

2.7　2010 年代中盤：利用者の振舞の判定に基づく認証方式の自動切り替え

前述の二段階認証・多要素認証により，シングルサインオンの利便性を保ちつつ，情報システムに応じて認証を強化する仕組みが確立されました．この方式では，利用者の SP へのアクセス手段等は，追加認証の判定には含まれませんでした．

しかし，「普段と違う場所 (海外など) から商品を購入しようとしている」「普段利用しない時間帯にサインオンしようとしている」等の行動は，利用者の振舞が通常と異なります．この状況は，「不正アクセスの可能性（リスク）が高い」と判断できます．最新の認証システムでは，このような「利用者の振舞」を追跡し，その振舞からリスクの大小を判断するシステム (リスクベースエンジン) をシングルサインオンの機構に組み込む拡張が進んでいます．リスクベースエンジンの判定結果による認証の切り替えに対応した方式を「リスクベース認証方式」と呼びます (図 10)．

リスクベース認証には，フェデレーション方式である OpenAM が対応しています [15]．

3　認証方式の今後の動向

認証方式の進歩により，現在では多くの情報システムがシングルサインオンに対応しています．利用者は最小限のログイン処理のみで，複数のシステムにアクセス可能です．

しかし，サインオンの仕組みが異なる場合や，異なるフェデレーションに属する情報システム間は，シングルサインオンの対象とはなりません．この問題に対応するためには，フェデレーションや認証方

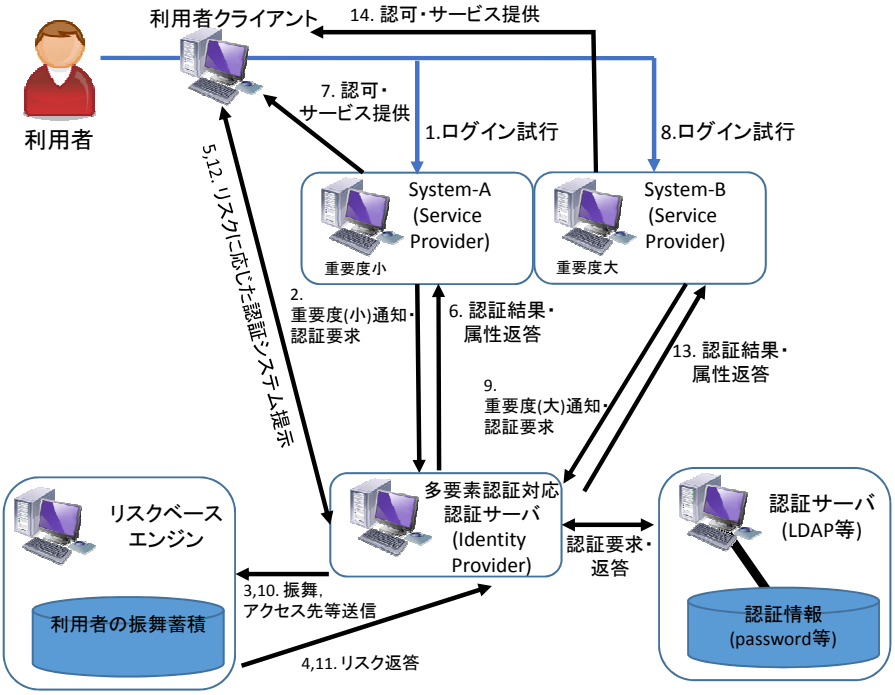

図 10: リスクベース認証方式

式の連携が必要となり，その技術的検証が進められています．現在，以下の連携が検討および実用化されています．

- **学認と eduGAIN(https://wiki.edugain.org/) との連携** 既に，学認のメタデータが海外のフェデレーションである eduGAIN[16] に送信されています．したがって，eduGAIN の SP は，学認 IdP が eduGAIN に加入していると判断します．学認 IdP 導入機関は，eduGAIN 加入の SP にサインオンが可能です．

- **Shibboleth と OpenAM との連携** Shibboleth，OpenAM の双方とも，フェデレーション方式の認証連携を行いますが，異なる仕組みで動作するため，相互のシングルサインオンは行われません．この問題への対応例として，福岡大学では「OpenAM IdP の認証プラグインとして，Shibboleth の認証システムを読み込む」という方式を採用しています [17]．OpenAM にログインすることにより，同時に Shibboleth 側のログイン処理が行われ，利用者からは双方にシングルサインオンされたように見える，という実装例です．

- **フェデレーション方式とソーシャル ID 方式の連携**フェデレーション方式とソーシャル ID 方式の連携が可能となれば，シングルサインオンの適用範囲が劇的に拡大します．対応例として，Shibboleth, SAML，OpenID Connect (OAuth) の連携方式に対応した認証サーバ (Gluu Server，http://www.gluu.org/gluu-server/overview/) が開発，公開されています [18].

既に多くのシステムが異なるサインオン方式で稼働しているため，方式の統一は容易ではありません．したがって，今後は「異なる方式が連携し，一つの ID によって，今よりも多くの情報システムが利用できる」方式がもっとも現実的でしょう．

4 まとめ

本稿では，利用者認証方式について,「時代と共にどのように変遷・進歩してきたのか」の観点で解説を行いました．かつては，単一のワークステーションや情報システム自体が，それぞれ独自の ID やパスワードを管理する方式が用いられていました．その後，認証の統合，認証システムの統合，シングルサインオン，多要素認証と技術は進歩しました．

現在では，利用者は最小限のログイン処理のみで複数のシステムが利用可能です．また，管理者にとっても，情報システム自体に利用者認証のバックエンドを渡さずに利用の可否が判定できるため，高いセキュリティを保つことが可能となりました．利用者・管理者共に注意すべき，認証強度の設定についても，リスクベース認証によって自動化が進んでいます．

新しい技術は突発的に誕生するものではなく，人々が「もっと良いもの」（本稿では，使い易い認証方式）を求めた結果，提案されるものです．利用者認証技術は時系列に並べると，利便性の観点において妥当な技術が提案され，進歩を遂げていることがわかります．今後も，様々な方式が提案され，私たちはより良い認証システムに触れることができるでしょう．

参考文献

[1] Solaris Naming Administration Guide，Part I，Chapter 1 Introduction to Name Services,
http://docs.oracle.com/cd/E19455-01/806-1387/6jam6926b/index.html

[2] Windows NT Server Product Documentation，Managing Windows NT Server Domains,
http://www.microsoft.com/resources/documentation/windowsnt/4/server/proddocs/en-us/concept/xcp01.mspx?mfr=true

[3] X.500 Directory standard,
http://www.x500standard.com/

[4] Lightweight Directory Access Protocol (LDAP): The Protocol，RFC4511,
https://tools.ietf.org/html/rfc4511

[5] Active Directory 技術情報,
http://technet.microsoft.com/ja-jp/windowsserver/bb466131

[6] White Paper：Oracle Identity Management,
http://www.oracle.com/technetwork/jp/middleware/id-mgmt/idm-tech-wp-11g-r1-334850-ja.pdf

[7] OpenIDM - open source provisioning solution,
http://forgerock.com/products/open-identity-stack/openidm/

[8] OpenID Foundation，http://openid.net/

[9] What is OpenID Connect?，http://openid.net/connect/faq/

[10] OASIS standards SAML V2.0 Standard,
https://wiki.oasis-open.org/security/FrontPage

[11] OpenAM - open-identity-stack,
http://forgerock.com/products/open-identity-stack/openam/

[12] Shibboleth - world's most widely deployed federated identity solutions,
https://shibboleth.net/

[13] 学術認証フェデレーション，国立情報学研究所,
http://www.gakunin.jp/

[14] 林 豊洋，本学における学術認証フェデレーション (学認) の導入について，九州工業大学情報科学センター広報 第 24 号,
http://www.isc.kyutech.ac.jp/kouhou/kouho24/pdf/kaisetu4.pdf，2012

[15] 田村 広平，OpenAM が提供する様々な認証方式,
http://codezine.jp/article/detail/6853

[16] The eduGAIN service，http://www.geant.net/service/eduGAIN/Pages/home.aspx

[17] 日立製作所,「Shibboleth」と「OpenAM」を組み合わせたハイブリッド型シングルサインオン認証基盤,
http://www.hitachi.co.jp/Div/jkk/kyoiku/casestudy/fukuoka/casestudy1.html

[18] gluu server，http://www.gluu.org/