
ユーザー・プロビジョニングのベストプラクティス



この資料は、中規模～大規模の企業/組織でのユーザープロビジョニングのベストプラクティスに関して記述し、説明するものです。

この資料では、IT部門の責任者が、複数のシステムやアプリケーションに対して、セキュリティポリシーを設定し、ユーザーアイデンティティと資格を管理するプロセスを設計する際の最適な手引書となることを目的としています。

この資料全体を通じて、ベストプラクティスを参照するには、 マークの箇所を見てください。

アイデンティティ管理とは何か？

アイデンティティ・アクセス管理は、ユーザーの識別データが、組織的に、地域的にまた複数アプリケーションに渡って散在している状態で、組織内のユーザーに関する情報を一貫して管理するテクノロジーとプロセスの集合を意味します。

アイデンティティ・アクセス管理は、基本的なビジネス上の課題を解決します。：従業員、契約社員、顧客、パートナー、ベンダなどのアイデンティティ情報は、どのように識別するか、それぞれ何にアクセスできるか、といった情報とともに、多数のシステムに分散されており、そのため管理するのが難しい。

エンタープライズ・アイデンティティ管理とは何か？

エンタープライズ・アイデンティティ・アクセス管理 (IAM) は、複数のシステムに散らばった多数のユーザーと資格情報を、効果的にかつ一貫して管理するプロセスとテクノロジーの集合として定義します。この定義では、通常ユーザーは、百万人よりも少ない規模ですが、複数のシステムと複数のアプリケーションをアクセスします。

一般的な企業向けアイデンティティ・アクセス管理シナリオでは次の機能を含みます。：

- ▶ パスワード同期とセルフサービス・パスワードリセット
- ▶ ユーザープロビジョニング、アイデンティティ同期、自動プロビジョニング、自動アクセス停止、セルフサービス・セキュリティ申請、承認ワークフロー、統合的リポーティング機能。
- ▶ エンタープライズ・シングルサインオン・クライアントアプリケーションのログイン画面への自動的応答
- ▶ ウェブ・シングルサインオン ・ 認証と承認プロセスを複数ウェブアプリケーション間で統合

エンタープライズ IAMは、エクストラネット(B2C や B2B)シナリオのアイデンティティ・アクセス管理とは異なる課題があります。：

特徴	エンタープライズ IAM	エクストラネット IAM
ユーザー数	100万人以下	100万人超
システム及びディレクトリ数	2 -- 10,000	1 -- 2
IAMシステム展開前に設定済みユーザー	数千人規模	新規ユーザーのみ
ログイン ID の一致	既存アカウントは、異種システムに異なるIDとして存在	ユーザー毎に単一の一貫したIDが存在
データの品質	孤立アカウント、休眠アカウントが存在。データのシステム間の一貫性欠如	ユーザー毎に単一か少数のオブジェクトのみ。一貫したデータ。休眠アカウントが頻繁に問題となる。
ユーザーの多様性	多数のユーザーが固有の要件を持つ。	ユーザーは少数カテゴリーに分類できる。

まとめると、エンタープライズ IAMは、ユーザー数は比較的少ないがより複雑であり、エクストラネット IAMは、より多数のユーザーでトランザクションレートが高いが、複雑度は低いと言えます。

資格管理とは何か？

バートングループ(The Burton Group)では、資格(entitlement)を次のように定義しています。:

資格とは、ユーザーアカウントがシステム内で複数のアクションを実行するため(場合によっては実行できないようにするため)に付与されるか、付帯する、システムのセキュリティモデルのオブジェクトである。この資格の定義は、Active DirectoryのグループメンバーシップやSAPのロールなどのように、システムのセキュリティモデルで最も高いレベルで付与されるオブジェクトであり、個々のファイルの許可設定などの低レベルのオブジェクトではない。

Ian Glazerによる定義は、*Access Certification and Entitlement Management v1*, September 9, 2009.

<http://www.burtongroup.com/Client/Research/Document.aspx?cid=1732> に記述があります。(このサイトは、ログインが必要です。)

資格管理は、組織の全般に関わるユーザーのセキュリティ権限を一貫して管理するのに用いるテクノロジーとプロセスの集合です。その目的は、管理コストの削減、サービスの向上、及びユーザーが適切な資格を持つことを保証することにあります。

これらの目的は、複数システムやアプリケーションを通して、資格を付与したり、解除を行う、堅牢で一貫性を持つプロセスを作ることによって達成することができます。:

1. 資格情報の統合的なデータベースを生成し、適宜更新する。
2. 業務ユーザーに解りやすい形でユーザーの資格が割り当てられるようなロールを定義する。
3. 資格に関する判断は、ITスタッフが行うのではなく、ビジネスユーザーが自身の知識でできるセルフサービス申請、承認を可能とする。
4. 複数のシステム間で適宜、資格の同期化を行う。
5. 業務関係者に対し、定期的に、関連する各ユーザーに付与された資格とロールを審査するよう依頼し、その関係者は、検証し、削除すべきかを判断するために、無効となっている情報を特定する。

企業/組織がより広範にIT基盤を展回していくなかで、その基盤を管理すること、特にユーザー、そのアイデンティティ・プロフィールや、ユーザーのシステムにおけるセキュリティ権限等を管理することは、一層難しくなっています。

図1. [\[link\]](#)は、複数のシステムにまたがる複数のユーザーを管理するために組織が直面する課題を表しています。

図1.



図1. ユーザーライフサイクル管理の課題

この図では、ユーザー・ライフサイクルの各フェーズでビジネス上の課題を示します。:

1. 新規ユーザーの登録:

a. **遅延と生産性:**

新規ユーザーには、即座に業務が実行できる環境を与えることが必要です。新規ユーザーへのアクセス権の設定の遅れは、余分なコストの発生となり、ひいては生産性の低下につながります。

b. **申請と承認:**

IT従事者は、新しく設定されたアカウントが適当であることを保証する必要があります。これは、通常、書面による申請、セキュリティ変更の審査と承認、を必要とします。この承認プロセスは煩雑で要求者側と承認者側の双方に追加作業が必要であり、結果として遅延を生じる可能性があります。

c. **冗長した管理:**

ユーザーは、通常、複数システムにまたがったアクセス権限を必要とします。新規ユーザーは、ネットワークログイン、e-メールのメールボックス、ファイアウォールアクセス、複数のアプリケーションへのログイン権限などを必要とします。これらのアカウント定義は、通常異なる管理者に、異なるツールによって行われています。こうした冗長作業は高価であると同時に時間が掛かります。

2. **変更の管理:**

ユーザーは組織の中では、頻繁に役職と責任範囲の変更を生じます。また、アイデンティティ属性の変更(例:ユーザーの名字、連絡先、部門、管理者等の変更)が生じます。これらの変更はアイデンティティ・プロフィールやセキュリティ権限への反映といったIT作業が必要となります。

企業/組織は、既存ユーザーの管理においても、新規ユーザーの定義と同様な課題に直面します。:

a. **遅延:**

ユーザーの異動により、申請にチャッチアップするためにITの対応を待つのは、時間の無駄です。

b. **申請変更:**

申請を提出したから承認に時間がかかるのは煩わしいものです。

c. **冗長した管理:**

同じような変更処理が異なるシステムで度々必要になります。

3. **IT サポート:**

システムを定常的に使うなかで、ユーザーは技術的なサポートが必要な問題にしばしば直面します。:

a. **パスワードを忘れる。**

b. **侵入者ロックアウトが掛かる。**

c. **エラーによりアクセス拒否される。**

まとめると、これらの問題は通常ITヘルプデスクの問い合わせボリュームの大半を占めています。これは、直接コスト(サポートスタッフ)と間接コスト(ユーザーの生産性)の浪費を意味します。

4. **終了処理:**

すべてのユーザーはいつかは離職します。その時、そのセキュリティ権限を見つけ出し、取り除く信頼出来るプロセスが必要になります。このプロセスは次のようであればなりません。:

a. **信頼性:**

組織がある部門ユーザーのアクセス権限の停止に失敗すると、このユーザーか、これになりすました侵入者が基盤を悪用し、重要データに危害を加えるかも知れません。

b. **適時性:**

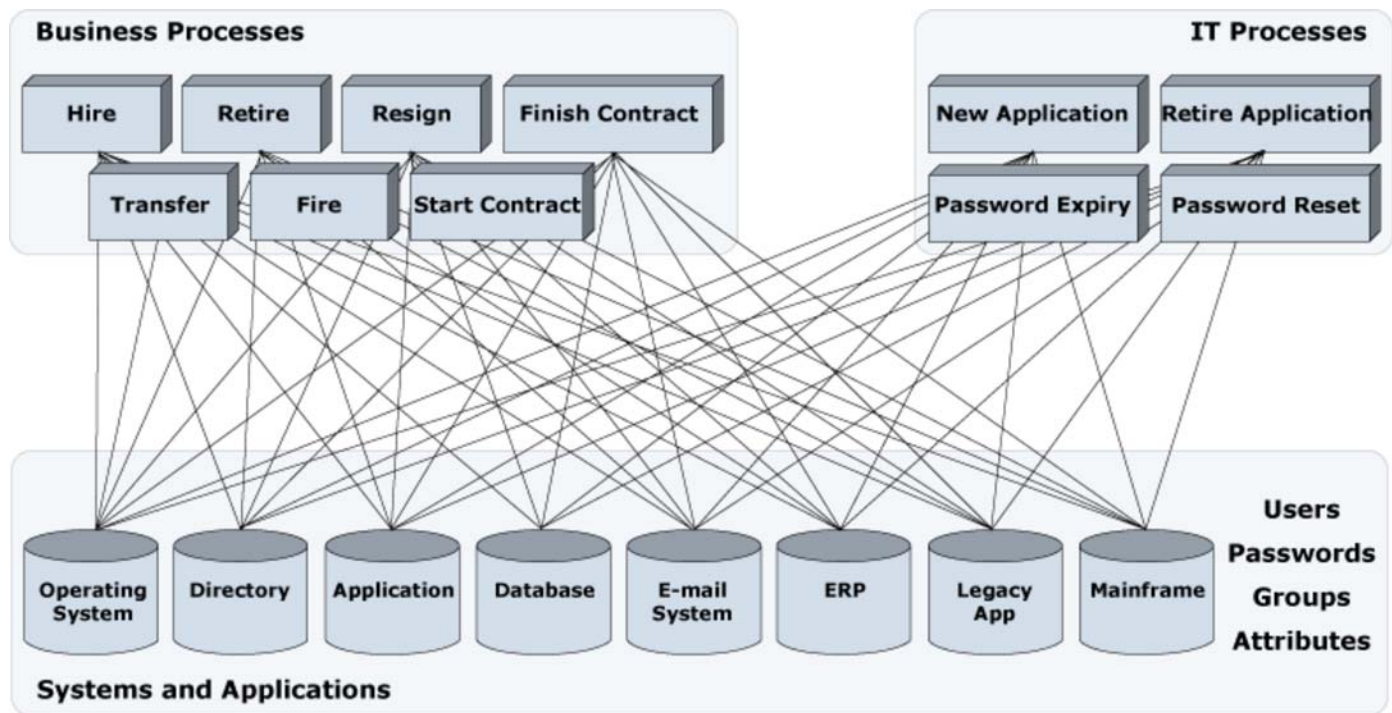
アクセスの終結は、前述の不正利用が可能な期間を最小とするために即座に行わなくてはなりません。

c. **完全性:**

不要となったユーザーログインIDを主要システム上で停止するだけでは十分ではありません。ネットワーク内のユーザーによる悪用の可能性をなくすためには、すべてのアクセス権限を無効にする必要があります。

アイデンティティ・アクセス管理システムが存在しないと、各システムやアプリケーション上で、ユーザーは別々の管理者に、異なるソフトウェアツールや異なるビジネスプロセスで管理されることとなります。これを図2. [\[link\]](#)に示します。

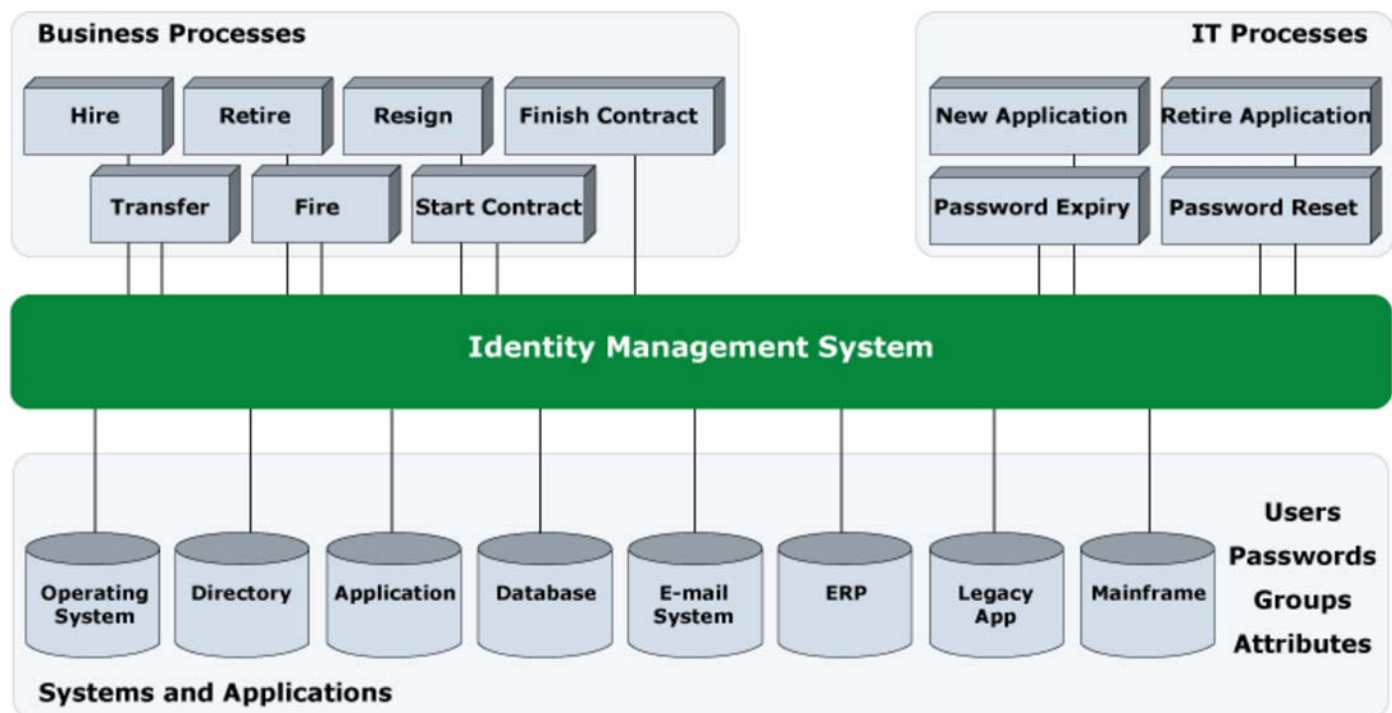
図2.



各アプリケーションをそれぞれのサイロ(格納庫)で管理

アイデンティティとアクセス管理システムは、各システムに実行されているプロセスをすべてのユーザーに統一して適用される新しいプロセスに置き換えることによって、ユーザーオブジェクトの管理を外出します。この簡略化したプロセスを図3. [link] に示します。

図3.



ユーザーと資格情報の管理を独立化

ユーザー・プロビジョニング・システムは、個々のシステムやアプリケーションから、ユーザー、アイデンティティ属性、資格の管理機能を外出し、共通に用いられる共用IT基盤です。

ユーザー・プロビジョニングは、複数のシステムに散在した、ログインアカウントとそれ以外のユーザーオブジェクトの生成、管理、解除をより早く、安価に、高信頼で行うためのものです。新規登録や終了のビジネスプロセスを自動化し、複数

システムと連携することでこれを実現します。

ユーザープロビジョニングシステムでは、次のようなプロセスを自動化します。:

▶ **アイデンティティ同期:**

あるシステム上でのユーザー情報の変更、例えば、電話番号、所属部門コードなど、を検知し、自動的に他のシステム上で当該ユーザーの該当する情報を変更する。

▶ **自動プロビジョニング:**

信頼できるシステム(例えば人事情報)上で新規ユーザーを検知し、自動的に他のシステムやアプリケーション上に当該ユーザーの適切なアクセスを提供する。

▶ **自動解除処理:**

信頼できるシステム上で削除、停止ユーザーを検知し、自動的にすべてのシステムやアプリケーションから当該ユーザーを停止する。

▶ **セルフサービスリクエスト:**

ユーザーが自ら自身のプロフィール情報(例えば、新しい自宅の電話番号など)を変更可能とし、新しい資格(例えば、アプリケーションやシェアへのアクセス権)の申請を可能とする。

▶ **委譲管理:**

管理者、アプリケーション所有者、または他の関係者に権限の範囲内においてユーザーや資格の変更を可能とする。


▶ **承認ワークフロー:**

すべての変更依頼を、それがだからからかに関わらず、検証し、統合システムやアプリケーションに反映される前にビジネス関係者に承認を依頼する。

▶ **統合リポーティング:**

複数システム、アプリケーションに跨って、どのユーザーがどんな資格を持っているか、休眠、孤立アカウントはどれか、変更の履歴、等に関するデータを提供する。

さらに、ユーザー・プロビジョニング・システムは、これらのプロセスをコネクタを使ってシステムやアプリケーションにつなげる必要があります。コネクタは次のことができます。:

- ▶ ターゲットシステム上のユーザーとグループを列挙
- ▶ 新規ログインアカウントの生成と既存のログインアカウントの削除
- ▶ ユーザーオブジェクトにあるアイデンティティ属性のリードとライト
- ▶ "アカウント有効化/無効化"、"アカウントロック"、"侵入者ロックアウト"といったフラグのリードと設定
- ▶ 既存アカウントのログインIDの変更(ユーザーのリネーム)  (note)
- ▶ ユーザーのグループメンバーシップのリード
- ▶ グループメンバーユーザーのリストをリード
- ▶ グループへのユーザーの追加または、グループからユーザーの削除
- ▶ グループの属性の生成、削除と設定
- ▶ ユーザーのディレクトリ OU 間の移動


アイデンティティ管理は、ユーザーが誰であり、どんなアクセスが可能かといった、ユーザーに関する情報管理を改善するものに他なりません。当然のことですが、ユーザーの情報の正確性を保証するには、ユーザー自身の関与を必要とします。そのため、アイデンティティ管理の自動化を成功に導くには、ユーザーフレンドリーなシステムを提供することが重要になります。

ユーザーが、旧来のマニュアルのシステムの利用に戻ることなく、このシステムを使うように動機付ける必要があります。ユーザーの観点からは、ヘルプデスクをコールするよりも、自動化プロビジョニング・システムを使う方が、簡単で、確かで、かつ価値がある必要があります。

次のような場合のユーザーに対するシステムの影響を配慮してください。:

- ▶ 登録プロセス、例えば、セルフサービス・ログインID・リコンサイレーション等、が必要な場合、
- ▶ セキュリティ権限の定期的なレビューを実現する場合、
- ▶ 管理者か、アプリケーション所有者がセキュリティ変更申請の承認を行う場合、
- ▶ すべてのセキュリティ変更申請が、システムを通してのみ行われる場合、つまり申請を行うユーザーがシステムの場

所、利用法を知っている場合。

ユーザーの関与が広く必要になる場合では、成功させるためには、次のような特別な対応が必要になります。: 

- ▶ ユーザーに、このシステムがなんであるか、何をしようとしているか、どこにあるか、どのように使うものかを理解させるための、ユーザー周知プログラムが必要です。
- ▶ ユーザートレーニングも必要になります。大部分のユーザーが影響する場合、そして、一般的に計画した教育の実施から実際のシステム利用まで長い時間が掛かる場合、高価で、忘れられやすい、人による教育よりも、システムに組み込まれたコンピュータ支援教育(CBT)を提供するのが好ましいでしょう。
- ▶ ユーザーには、システムへの参画が必要な場合に、システムを使うように自動リマインドする必要があります。
- ▶ 多忙なユーザーや、単純にシステムに回答できないユーザーを許容する必要があります。--リマインダー、権限の委譲、一人のユーザーから他へのエスカレーション。

ユーザーの不適切な行動を防ぐために、それを防止する手段を講じるのも一案です。例えば、アクセス変更に関しては、書面形式を引き続き用いるが、オンライン申請よりもずっと長い処理時間を掛けるようにするなどです。

マニュアルによるユーザー管理の弱点の一つは、人間の行動に一貫性がないことです。つまり時として過ちを犯します。その結果、セキュリティ管理者は、ユーザーがどのアクセス権限を持つか、に関して、確かな基準で管理していると信頼されなくなります。


アイデンティティ管理システムは、変更はどのように申請され、ユーザーは何を持っており、どのように承認され、どのように実行されたかの全体に渡る基準を実現できなければなりません。これには次のものがあります。:

- ▶ 固有な識別子の割当:
 - 従業員番号
 - ログイン ID
 - E-mail アドレス
- ▶ オブジェクトの配置:
 - 新規ユーザーを正しいディレクトリに配置する
 - 適切なサーバー上にメールボックスとホームディレクトリを生成。
- ▶ デフォルトセキュリティの設定:
 - 初期グループメンバーシップとロールメンバーシップ
 - ホームディレクトリ上の初期セキュリティ ACL / 許容値設定
 - セキュリティ設定- アプリケーションとディレクトリ上の関連属性
- ▶ 変更承認:
 - 変更申請が認証されたユーザーにより提出されるか、または、適切にサインされていること。
 - 適切な関係者がすべての変更申請の許可に関与すること。
 - 承認が完了するまで実行が延期されていること。
- ▶ その他:
 - メール配布リストのデフォルトメンバーシップ
 - ディスク割り当て
 - テーブルスペース割り当て

気をつけなければならないのは、ユーザー・プロビジョニング・システムを展開する前に、次の章で述べるように、前記項目の標準ポリシーを定義することです。

ベストプラクティス

固有な識別子の割当て

実際のセキュリティ基準は、組織によって異なります。多くの組織が有効だと感じるベストプラクティスは、次のようなものがあります。: 

- ▶ ユーザーに短いログインIDを割り当て、すべてのシステムのIDとして用います。

- IDは、レガシーシステムを含むすべてのシステムで互換性があるように、短く、タイプが容易であること、
 - つまり、ログインIDには、最長7か8文字で、文字、数値だけを用います。
- ▶ フルネーム、SMTPアドレスや、フルディレクトリパスのような長いログインIDをサポート、但し、セコンダリIDとして用います。長いIDも、用いますが、短い固有IDを置き換えるべきではありません。
 - ▶ ログインIDをグローバルにユニークにします。同じIDを異なる二つのシステムまたは、異なるOU上の異なるユーザーにアサインしてはなりません、後日、相互関係が発生したときに問題が起こるからです。
 - ▶ IDを再利用しないこと。一度、IDがあるユーザーにアサインされたら、そのユーザーを永続的に表すものとすべきです。これにより、記録監査の正当性が保てます。
 - ▶ 階層的なネームスペースが用いられている場合でも、各ユーザーにはグローバルなユニークIDを、通常、ディレクトリCNの中で与えます。言い換えれば、ディレクトリのなかでは、二つのコンテキストの中では、CNは二度出現することはありません。これは、IDを階層的なシステムとフラットなネームスペースの間でマッピングするときに便利です。
 - ▶ ログインIDを人にアサインし、組織のポジションにアサインしないこと。人はロールを変更しますが、コミュニケーション（例、同じe-mail アドレス）や、監査ログとの関係の明確性のために、ログインIDは、人に付随して行くべきです。

沢山のアルゴリズムが上記ガイドラインに従ったログインIDを割り当てるために使うことができます。例えば、：

- ▶ ユニークIDの生成のために、ユーザーの名前の最初の3文字と、5桁のシーケンス数字を用いる。
- ▶ ユーザーの氏名の最初の4文字と、最初と2番目のイニシャル、それに続いてユニーク性を保証するための2文字を使う。

グローバルな固有性を保証し、再利用を防ぐには、いくつかのシステムでは、すべての現在利用中のID、予約されまだ使われていないもの、過去に使われたが現在は有効でないもの、を管理するテーブルが必要となります。このようなテーブルによりIDの再利用を防ぎます。

オブジェクトの配置

新規アカウントを正しいディレクトリに配置し、新規メールボックスやホームディレクトリフォルダーを適切なサーバーとディスクボリュームに配置するのは、直接的に行えます。--ディレクトリ構造、メールサーバー基盤の構造等から直接導くことができます。

より難しいのは、ユーザーのアイデンティティ属性の変更をトラックし、ユーザーのアイデンティティ属性の変更に従って、アカウントを新規ディレクトリに自動的に移動したり、メールボックスを新規サーバーに移動したり、ホームディレクトリに再配置することにあります。

例えば、もしディレクトリ構造が組織の部門を表し、ユーザーの部門変更があれば、そのユーザーのディレクトリ上のオブジェクトは、同様に変更しなければなりません。

同様に、ユーザーのメールボックスをホストする適切なメールサーバーの選択が、ユーザーの物理的な所在地に依存しており、ユーザーがある支社から別の場所に異動した場合、ユーザーのメールボックスも自動的に移動しなければなりません。

セキュリティ資格

同様に、新規ユーザーのデフォルトセキュリティ資格は、ユーザーの所在地や業種コード等の基準に基づいて設定します。重要なのは、前述と同様、アイデンティティ属性の変更を検知し、自動的に適切な変更を講ずることです。

例えば、ユーザーの初期グループメンバーシップがユーザーの所在地と部門から導かれており、ユーザーが後で所在地や部門の変更が起こったとき、いくつかのグループメンバーシップは取り除かれ、別のものを追加する等が自動的に行われなければなりません。

現実には、これは、言うは易く行うは難しで、この場合、ロールは部門と所在地のあらゆる組み合わせで定義する必要があり、そうしたロールの変更が起こる得るからです。多数のユーザーに共有されるロールがあり、それを組み合わせることによってのみ、ロール利用の経済性が保てます。

変更承認

最初に、“変更申請”とは何かを明らかにする必要があります。

変更申請は、基本的には書類で、何人かの参加者を介して行われます。：

1. 単一申請者 .. 人による、または、自動化申請
2. 単一受領者 .. システム上のプロフィールに影響がある人または、人に例えたもの
3. ゼロ人以上の承認者(審査、承認、拒否を依頼される人)

このほかに別の参加者がいる場合があります。--権限がエスカレートされたか委譲された代替承認者; 申請キューをモニターし、問題に回答するワークフローマネージャー; いくつかの申請などをマニュアルで実行する実行者。

受領者プロフィールへの一つ以上の変更を指定する申請は次を含みます。:

1. 新規プロフィールの生成
2. 既存プロフィールの無効化または削除
3. アイデンティティ属性の変更、場合により、ユーザーの固有の識別子やディレクトリコンテナの変更などの特殊ケースを含む。
4. 新規ユーザーオブジェクト(ログインアカウント)の生成やプロフィールへの追加
5. ユーザーオブジェクトの無効化または削除
6. 受領者をセキュリティグループに追加
7. セキュリティグループからの受領者の削除
8. 受領者をロールに追加
9. ロールから受領者の削除

申請は即座に処理される.. つまり即座に実行が行われる.. かもしれませんが、後日にスケジュールされるかも知れません。それぞれ相当な理由があります。

ビジネス要件に沿った変更要求だけが受け付けられるようにすべきです。これは通常次の二つのステップで行われます。:

1. 要求検証:

要求がいかなるビジネスルールをも侵害しないかチェックする自動検証。例えば、要求はSoDルールを侵害しないこと、不正な部門や場所のコードを指定していないこと等。

2. 要求承認:

単純な要求、例えば、ユーザーの電話番号などのセルフサービス更新、は即座に実行する。

信用できるシステムや人間からの要求.. 例えば、人事システムからの信頼に足るデータ(HR部門が生成)や.., 信頼度が高い人間.. 例えば、CFO.. 等による入力.. 等による入力はこれ以上の承認を必要としない。

他のすべての要求は、それが実施される前にビジネス関係者によるレビューが必要である。

承認が必要な申請に対しては、当然、「誰が承認するか？」という疑問が湧きますが、これには、次のわずかな可能性しかありません。:

- **管理者** -- 通常、申請受領者、場合によっては申請者でもあります。
- **リソース所有者** -- 申請により影響を被るリソースの、注記:リソースは次のものを含みます。:
 - アプリケーション
 - セキュリティグループ
 - ロール
 - SoD ルール (申請がSoD侵害を含む場合)
- **セキュリティ管理者** -- 例えば、ある企業/組織では、各部門や各事業部の中で、セキュリティ変更承認する主要業務ユーザーを指名しています。

ある申請の承認に関して、複数のタイプの承認者に依頼するのは賢明な方法です。通常、リソースの所有者に加えて、申請者のマネージャーを加えるなどします。✔

このルールには必然的に例外があります。特に、ある一定以上の幹部は、変更申請に経営幹部の承認を必要とさせ

んし、また、その地位が故に、リソース所有者に承認を依頼するのは無意味です(その場合、デフォルトで許可される)。

また、ユーザーは自分自身に関する変更申請の承認を自ら行うべきではありません。 -- 回答はつねに“イエス”だからです。ワークフローエンジンが承認依頼を送信するまえに承認者のリストをチェックし、もし申請者が承認者であると認識されれば、申請の一部として事前承認を行うこととなります。✔

承認者の時系列も配慮しなければなりません。マニュアルのシステムでは、承認者は、申請は一人ずつ承認依頼されます。・・申請は、申請書用紙の回覧によって行われます。自動ワークフローシステムでは、すべての承認者に一括依頼をすることが可能です。これは、申請提出から実行までのトータル時間を最小限にするためには魅力的な方法です。申請の承認が全員から得なければならぬ場合には、申請審査承認をシリアルライズするセキュリティ上のメリットはありません。・・BさんのまえにAさんが、あるいはAさんの前にBさんが承認をしようがありません。✔

迅速な応答を保証するため、複数の承認者候補に承認依頼をし、その中の一部が承認したことで承認されたと扱うことも一般的に用いられます。最も早い承認が早い時期に申請の実行を促すことから、承認レースを作り出すこととなります。言葉を変えれば、一つのリソースに対して、N人のリソース所有者グループへ申請を促し、M人からの容認回答 ($M <= N$) が得られれば承認されたと見なされるということです。✔

複数の、同時承認者をサポートすることは、ある承認者は申請を許容し、別の承認者は却下するという状況を作り出すことにもなります。この状況に対する簡単な解決方法は、申請が許可されるまでの期間に発生した拒否は、拒否権のように扱い、申請を全体的にブロックすることです。申請が必要最少数の承認者により承認されると、申請は完了しすべての他の承認者に通知されます。この事例は、承認者は他の承認行為に対し、タイムリーに行動している限りは、異議を申し立てることができ、申請の状態の不確実性をなくします。このアプローチは、承認者に迅速な回答を促します。--ある人が申請をブロックしているとしたら、迅速に審査をしなければなりません。✔

図 [link] に示すような統合ユーザー・プロビジョニング・システムは、管理上の複雑さを軽減するように意図しています。(_label_what-up) に記述される、次のようなビジネスプロセスを実現することにより可能です。

以降のセクションは、各プロセスが、いつ、どのように使うか、・・また同様に重要ですが・・、どういうときに動作しないかを記述しています。

アイデンティティ同期

一つのシステム上の電話番号や、部門コードなどの個人データの変更を検知し、自動的に他のシステムの当該ユーザーの同様なデータ変更を行います。

用いるタイミング

複数システムが同じアイデンティティ情報を持ち、最低一つのシステム上で、情報が、信頼できる、タイムリな方法で更新されたとき。

範囲

ログインIDを最低二つのシステム上で持つすべてのユーザーで、最低そのうち一つのシステムのアイデンティティ情報が信頼しうる、タイムリな更新があった場合。

使用法

定期的におすすめのシステム上のアイデンティティ情報を読み、システム間の情報の相違を検知し、“より信頼に足る”システムからのデータを受付け、同じ情報に関して“より信頼に劣る”システムに反映します。

注意すべき点



1. 各アイデンティティデータの各項目について、“信頼に足る”システムを明確に定義する必要があります。
2. 他の情報に関しては、別のシステムがより信頼に足るかもしれないことを考慮すべきです。例えば、電話番号の信頼できるソースとしてはホワイトページアプリケーションが、e-mailアドレスについてはe-mail システムが、部門番号は、人事(HR)システムが信頼できるソースでしょう。
3. アイデンティティ管理システム自身が“最も信頼できる”ものとしてはいけません。セルフサービス更新を任意の情報に

許す場合は、最初にはこれは正しいかもしれませんが。残念ながら、すべてに対してセルフサービスを信頼おけるものとするには、大変に時間がかかりますし、好ましくはありません。その代り、セルフサービスUIによる“一時的な上書き”を許すようにすべきです。

自動プロビジョニングと自動解除

信頼できるシステム(HRなど)での新規ユーザーを検知し、該当ユーザーに自動的に他のシステムやアプリケーション上に適切なアクセス権を与えます(プロビジョン)。

信頼できるシステムでの削除されたか、停止されたユーザーを検知し、該当ユーザーを自動的に他のシステムやアプリケーション上から無効化します。

用いるタイミング

自動変更プロパゲーションは、次のときにのみ有効です。:

- ▶ 登録システムが存在する。
- ▶ 登録システムは、ユーザーに関する正確な情報を持つ。
- ▶ 登録システムのユーザーデータの更新がタイムリーに行われる。

もしこれらのいずれかの条件が満たされないときは、自動変更プロパゲーションは用いるべきではありません。

登録システムがある分類のユーザーのみに存在するとき、自動化はこの分類のユーザーのみに有効です。例えば、信頼できる、タイムリーな登録システムがHRのみであるとき、自動変更プロパゲーションは、契約社員やベンダ等に対しては適当ではありません。

最後に、自動変更プロパゲーションは、登録システムに提供されているデータの粒度でのみ用いることができます。前述の例では、もし人事(HR)アプリケーションがユーザー名称、雇用日付と終了日のみを管理しているとする、HRのデータを基に、詳細な資格情報の割当てをすることはできません。

範囲

ほとんどの企業/組織では、登録システムは、人事(HR)アプリケーションです。またほとんどの組織では、こうしたシステムのデータは、従業員にのみ適用されており、また概略情報です(詳細なセキュリティアクセス要件に関する情報はありません)。したがって、そうした組織では、自動変更プロパゲーションは次の範囲でしか使えません。:

- ▶ 新規ユーザーの基本的なシステムアクセスを自動的にプロビジョンします。例えば、ネットワークログインID, e-mail アカウントや、インターネットアクセスは、すべての新規従業員に設定されます。
- ▶ 退職したユーザーのすべてのシステムアカウントを、どんなアカウントを持っていようとも、自動的にディプロビジョンします。..組織を離れた時に、ほとんどの環境では、すべてのアクセスは無効とするべきです。

より細かいアクセス権の扱いは、通常他のプロセスに任せられます。

使用方法

自動的に登録システムのデータ品質、タイミング、範囲、詳細度のレビューが開始されます。

参照データのタイプと内容がレビューされ、受け付けられると、次のステップで登録システムのどのデータ変更が関係し、各ターゲットシステムのどんな変更にもマッピングされるが特定します。

次に、登録システムをモニターするためのデータ提供手段が設定され、変更を抽出します。

最後に、登録システムを入力とし、管理システム更新が出力となる、転送手段を定義します。

注意すべき点

HRシステムと他の登録システムは、ユーザーの組織でのロール(役割)を表現するジョブコードや同様なフィールドを持っています。これは、ロールを資格の集合にマッピングし、ユーザーは、ジョブコードやそれに従ったロールに必要なすべての資格を与えるロールベースアクセスコントロール(RBAC)を連想させます。

RBACは、多数のユーザーが同じ権限を持つ場合に有効なメカニズムです。これは、通常、販売員など、リスクが低く、管理負担が大きいところで用いられます。✔

RBACは、ユニークな仕事をしている管理業務ユーザーにはお勧めできません、多くのユーザーが同じ要件を持って

いないため、RBACを提供するメリットがないからです。✔

セルフサービス申請と管理の委譲

ユーザーには、自分自身のプロフィール(例えば、新しい自宅の電話番号)を更新したり、新しい資格(例えば、アプリケーションやシェアへのアクセス)を申請できるようにします。

マネージャー、アプリケーション所有者や、他の関係者には、ユーザーや資格を自分の責任範囲において、修正できるようにします。

使用するタイミング

セルフサービス申請と管理の委譲は、コンピュータ、特にWebブラウザを使い慣れた知的従事者が、現在の情報をレビューしたり、変更を申請したりするには効果的です。✔

セルフサービスは、コンピュータへのアクセスが困難で、未習熟で、また、新規アプリケーションの利用に際し新たなトレーニングを必要とするユーザーには適していません。✔

範囲

ユーザー、その同僚、及びその管理者は、ビジネス要件での変更に関して最も信頼に足る情報源となります。ユーザーが特定の資格、例えば、新規ロール、アカウント、グループメンバーシップ等に関する変更申請をできるようにするのは現実的です。個人のプロフィール情報・氏名、電話番号等の保守をユーザー自身に委譲することも現実的です。

自動化は、総じて従業員の管理に適している一方、他の分類のユーザー・契約社員、ベンダー等、登録システムが用意されていないユーザの管理には、多くの場合、委譲管理が唯一のオプションです。

ユーザー管理の委譲は、管理者または、IT管理者が異なる場所で従事している場合や、ビジネスユニットがその範囲で、ユーザー管理の責任と専門知識を持っている場合に現実的です。

使用法



1. 申請システムは出来るだけ簡単に、ユーザフレンドリに作ります。
2. ユーザーにシステムの有効性について周知します。
3. アプリケーション・ログイン・プロンプトや、ユーザーがアプリケーションやデータにアクセスするかもしていない他の画面からシステムにリンクし、不適切な資格を得ることがないようにします。
4. ユーザーが申請できる資格のセットを申請者や受領者にとって適切な範囲に限ったものとします。確実に拒絶される資格の申請を許す余地を残すこと現実的ではありません。
5. すべての申請を評価し、侵害(例、SoDまたは同様なことに対する)を引き起こす申請を事前に拒否します。

セルフサービス申請システムや委譲管理システムを展開するために、次のデザイン要件を決める必要があります。:

- 申請を許されるユーザーはだれか。
- ユーザーは他のユーザーのために申請を行えるか？ そうでなければ、申請者と受領者の関係を制限するものは何か？
- ユーザーはリソースを要求することができるか？ そうでなければ、申請者とリソースを制限するものは何か？
- どんな申請が申請者に許されているか(生成/雇用？ 削除/停止？ 変更？)？

注意すべき点

組織では、一般に申請可能なリソース集合を階層的に組み立てようとしています。しかし、多くの場合、ユーザーが利用したいリソースが階層のどこにあるかわからないために、これは望ましい方法ではありません。“トリー”構造を見せるよりも、検索機構を用意するのが望ましいです。✔

承認ワークフロー

すべての申請された変更を検証し、その申請元に関わらず、それが統合システムやアプリケーションに反映されるまえに、業務上の関係者に承認を依頼します。

使用法

ユーザー・プロビジョニング・システムが自動的に許可できない申請を受け付けた時には、常に承認が必要となります。これには、管理者やアプリケーション/データ所有者によるセルフサービスの新規資格や管理委譲申請を含みます。

範囲

ユーザー・プロビジョニング・システムにより処理されるすべての変更は、100%信用に足る要求元からであろうと、ビジネスリスクがないとされていても、承認審査の対象になります。

操作性の面からは、承認は、通常管理者、アプリケーションやデータ所有者、及びセキュリティ管理者に対し、その一部がすべてに対して、変更申請をレビューし、許可するか拒否するかが依頼されます。

使用法

エンタープライズ規模の承認プロセスを展開するには、次のデザイン要件を決める必要があります。✔

- 申請の各タイプをどのように検証するか。(例えば、入力フィールドのシンタックスチェック、コード選択、一貫性チェック)
- 申請の各タイプで適切な承認者はだれか？ 通常、受領者の命令システムのなかの管理者とアプリケーション所有者の組み合わせが使われます。
- 承認者から期待される応答はなにか？ リマインダはいつ送付されるべきか？ 応答しない承認者を他の代替承認者にエスカレーションするタイミングはいつか？
- 承認のどの部分を承認者が見ることが許されるか？ たとえば、社会保障番号やそれに類するものは申請にある場合がありますが、これらの表示は適切ではありません。
- 申請のどの部分を承認者が変更できるか？ もし承認者の一人が申請を承認して、二番目の承認者がそれを変更すると、最初の許可が無効になってしまうかも知れません。
- 承認者をグループで定義し、実行に移すまえに、グループ内の何人が承認しなければいけないかを決めます。

注意すべき点

現実のフルスケールのユーザ・プロビジョニング展開においては、多様なアプリケーションのアカウントの生成、変更、削除に関わる、何百種類もの申請が発生します。この規模のため、すべての種類の申請に対して個別のプロセスを定義するのは大変高価です。だれが何百ものフローチャートを書き、だれが保守できるでしょう？

その代りに、各申請の内容・申請者、受領者、リソース等・に基づき、申請を適切な承認者にグローバルにルーティングする適用可能なロジックを定義するのが現実的です。✔

承認者は、単なる人間であり、したがって、総じて信頼に足るべきプロセスのなかの、信頼できない当事者ということになります。申請許可の依頼に対して、承認者が適時に応答しなかったり、または、全く応答しない可能性があることを、システム設計時に考慮することが必要です。これは特に次を意味します。: ✔

1. 実際に必要な数以上の承認者に依頼する(Nの承認が必要なところMに依頼、 $M \geq N$)
2. 承認者は一時的、例えばしばらくの間、対応できない場合に、その責任を委譲することを許す。
3. 応答しない承認者に自動リマインダを送付する。
4. 数回のリマインダーを送付した後、非応答承認者から代替者にエスカレーションする。
5. 承認者が不在とわかった場合(例、“不在”メッセージがe-mailシステムで送付されたとき)、依頼状を送付する前にエスカレーションする。

統合リポーティング

複数システムやアプリケーションに対して、ユーザーがどんな資格を持っているか、どのアカウントが休眠か孤立か、変更履歴に関して等の情報を提供します。

利用するタイミング

ユーザー・プロビジョニング・システムを展開するすべての組織は、システム全体におけるアイデンティティや資格データをレポートできるメリットを享受できます。

範囲

システムによって管理されるすべてのユーザー、アイデンティティ、資格は、レポートで可視化する。

使用法

IAMレポートを有効にするには次の二つのシナリオがあります。:

- ▶ 一回限りのものとしては、特定な回答に答える必要があります。これは、監査やアクセス問題の調査のためのものです。
- ▶ 繰り返しや定期的なものは:
 - SoDルールなどのポリシー侵害するユーザーを特定する。
 - 重要なデータやアプリケーションをアクセスするユーザーを特定する。
 - IAMシステムの性能や利用率をモニターする。(例:処理した申請数、アカウント生成/削除数 等)

注意すべき点

ユーザー、アイデンティティ属性や資格等のデータは標準化した形式で、リレーショナルデータベースにドキュメント化された形式で格納します。これにより、IAMに組み込まれていないカスタムレポート形式も作成することができます。✔

企業統制と個人情報保護の両者は、多数のアプリケーションとIT基盤上に展開する堅固なセキュリティに依存します。そのようなセキュリティなくしては、内部統制は実現せず、法令遵守も保証されません。

ITセキュリティは、ユーザー認証、アクセス認可、及び監査(これらは、Authentication, Authorization, Auditの頭文字を取ってAAAと一般に呼ばれています。)のインフラストラクチャに大きく依存しています。AAAは、つまり、ユーザーに関する正確で適切な情報、..誰か?、どのように認証されるか?、何にアクセスできるか?..に依存しています。

組織に課題があるのは、こうした識別情報を管理することにあります。とても沢山のユーザーがおり、多数のシステムへのアクセスがあり、さらに新規採用、異動、ビジネスプロセスの終了等の結果、それらが変化し続けるからです。

AAAインフラストラクチャは、特段新しいものでなく、古くからすべてのマルチユーザーアプリケーションに存在しているものです。システムやアプリケーションの数が増えるにしたがって、またスタッフの異動が頻繁になるなかで問題が顕在化し、既存のAAAインフラストラクチャのなかでは、ユーザーデータの管理がより難しいものとなってきました。

弱いパスワード、ヘルプデスクに掛かってくる人の識別に対する信頼性欠如、孤立アカウント、不適切なアクセス権限、不整合のログインIDなどの存在によって、AAAシステムは、しばしば誤ったタイミングで誤ったルールを強いてきました。この脆弱性は、AAAの技術にあるわけではありません。—AAAが対象としているユーザーデータを管理するビジネスプロセスが問題なのです。

AAAデータの問題に焦点を当てると、ユーザに関するデータを、正しいユーザが正しいデータのみを、正しい時間にアクセスすることが出来る、しっかりとしたプロセスを実装することに尽きます。

これは、次の手段によって実現できます。:

- ▶ より堅固なパスワード
- ▶ ヘルプデスクに掛かってくる人を特定、認証する信頼性のあるプロセス
- ▶ ユーザーがどのようにログインアカウントとセキュリティ権限にアクセスしたり、それがいつ失効するかについてのより良い制御手段
- ▶ 管理者及びアプリケーションオーナーに委ねられるユーザーアクセス権限への定期的な監査
- ▶ 監査記録が人に対応付けられるように、システムに跨るユーザーオブジェクトの関連付け、
- ▶ 法的監査をサポート可能な、現在及び過去のアクセス権限の記録

ロールを使い適切な資格を付与する

ロール・ベース・アクセス・コントロール(RBAC)は、資格情報を管理する一つの手段で、ユーザーが適切な資格のみ持ち、必要のない資格を確実に適時に取り除くことで、セキュリティ管理コストの削減を狙っているものです。

単一のシステムやアプリケーションでは、RBACは権限を直接ロールに付与しユーザーにロールを結び付けることを意味します。ユーザーは、ロールメンバーシップを介して権限を取得します。単一システムの中では、ロールはセキュリティグループ、あるいは、ユーザーグループと呼ばれます。

単一システムRBACでは、グループユーザーまたは、グループの管理者は、個々のユーザーに個々の権限を割り当てるのではなく、権限をまとめてユーザーのグループに権限のグループを割り当てることができます。

ユーザー・プロビジョニング・システムでは、RBACを単一アプリケーションの範囲を超えて用います。ユーザー・プロビジョニング・システムでのルールは、複数システムや複数アプリケーションに及ぶ資格情報を取り扱います。ルールの主な特長は、多くの専門的な資格情報を、ビジネスユーザーが理解し得るより少数のルールにより置きかえることにあります。ビジネスユーザーは、どのユーザーがどのルールを持つべきかを適切に判断することができます。これは、ユーザーがどの専門的な資格情報を持つかを間接的に指定することになります。


ルールは、資格情報・ログインアカウントとセキュリティグループ・メンバーシップにより構成されます。ルールは、ネストすることが可能です。つまり、ひとつのルールは他を包含することができます。ルールのネ스팅により、ルール管理のコストを削減することができます。

ルールを使うと次が可能となります。:

1. 正しいルールを割り当てることによって、新規ユーザーに適切な資格を付与する。
2. ユーザーのルールが変わったときに古い資格を除去する。
3. ビジネスユーザーが必要な資格を申請が、理解しやすいルールを使うことで簡単になる。



1. すべてのユーザーにとって大まかな基本的な資格
2. 大きな集団に属し、共通な要件を持つユーザーに対する詳細で完全な資格

ルールベースド・アクセス・コントロール は、技術的にはすべてのユーザーに用いることができますが、すべてのユーザーに固有なルールを定義するのは、コスト的に現実的ではありません。 

権限分離 (SoD) ポリシー

権限分離 (SoD) ポリシーにより、組織は、一人のユーザーが保持してはならない、問題ある資格の組み合わせを定義することができます。こうしたポリシーは、多くの一般ビジネスでは、詐欺行為の防止に役立ちます。つまり、詐欺行為は、少なくとも二人のユーザーの共謀なしでは、行えないようにします。

効果的なSoDエンジンは次のコンポーネントを持ちます。:

▶ ポリシー定義:

- 一人のユーザーが持つてはいけない二つ以上の資格の集合を定義。
- ルール、ログインアカウント、グループなどの異なるタイプの資格をサポート。
- 大きな集合による定義をサポート、例えば、“どのユーザーも、この30種の資格のなかから二つ以上を持つてはいけない。”

▶ 例外の許容:

SoDポリシーが、正当な理由で、侵害される状況が必要なのは避けられません。SoDルールに例外を許容することが出来るようにすべきです。

▶ 事前施行:

IAMシステムに渡される変更申請は、SoDポリシーチェックにかけられます。SoD侵害を起こす変更は申請段階でロックされます。


▶ 事後検知:

多くのユーザーと資格はIAMシステムが展開される前から、または、SoDポリシーが定義される前から存在しています。それ以上に、システム管理者は、IAMシステムの外からユーザーに資格を与える場合があります。こうした状況は、すべてのSoD侵害が未然に防げないことを意味します。事後に検知し、マニュアルで修正する必要があります。

効果的なSoDエンジンでは、ポリシーはルールとして指定されいれば、低レベルの資格としては定義されていなくても、侵害を検知すべきです。あるいは、その逆も可能とすべきです。

R1は、Ea と Ebの資格を持つロールで、R2は、Ec と Edを持つロールのとき、次の侵害を考えてみてください。:

SoD ポリシー	ユーザーの所有	申請された資格	侵害理由
R1 と R2 は、相互排他	R1	R2	直接侵害
(同上)	R1	Ec, Ed	ユーザーは実質的にR2を持ってしまう。
Eb と Ec は、相互排他	R1	R2	ユーザーは、R1によりEbを、R2によりEcを持ってしまう。
(同上)	Ea, Eb	R2	ユーザーは直接Ebを持ち、R2によりEcを得てしまう。
(同上)	Ea, Eb	Ec	ユーザーはEbを直接持っており、Ecを持ってしまう。

SoD施行のベストプラクティスは次です。 

1. 組織の財務及びコンプライアンス担当管理者と連携し適切な SoD ポリシーを定義します。
2. SoD ルールを事前に施行し、新たな侵害を防ぎます。
3. SoD ルールが定義される前にあったか、ユーザー・プロビジョニング・システムの範囲外の既存の資格に対して、SoD侵害がないかどうか定期的にスキャンします。
4. SoD エンジンが、明示的なもの以外でも、効果的に侵害を検知するか検証します(上記参照)。

定期的なレビューと資格の関連付け

法令準拠要件とセキュリティポリシーの要求は、ますます高まっており、組織は、誰が重要な企業情報や従業員や顧客の個人データにアクセスしたかに関する効果的な制御が行えるように求められています。:

- システムは、適切なユーザーが適切な時間でのみアクセスできるよう制限しなければなりません。
- 組織は、こうした制御が正しく、効果的に行われていることを監査しうる証拠を提供しなければなりません。Sarbanes-Oxleyのセクション404では、毎年、マネージメントは、内部統制の効果を査定しなくてはなりません。
- 組織は、重要データへのアクセス権を、現在、どの内部ユーザーが持ち、過去はどうだったかを報告できなければなりません。

こうした要件を満たすのは、チャレンジングです。ユーザーは、各人業務責任が異なり、また、絶えず変化しており、資格を形式的なロールやルールを使いモデル化するのは、難しいからです。

複雑で、異種の資格をモデル化する難しさは、ユーザーに時間経過とともに多くの資格が蓄積されるにも拘わらず、IT部門に対して、古く不要になった権限を無効にするように依頼することはまれであるという実態により、さらに助長されます。さらに難しいのは、ユーザーの責任範囲が変わった後に、旧職のバックアップリソースとして役割を果たす必要がなくなって、本当に旧資格を解除してもよい時期を予め判断するのが困難だからです。

これらの課題は、ある時点で複数システム、複数アプリケーションに渡って、ユーザーが必要となる資格をモデル化するのが難しいこと、さらに、複数システムに対して、長期間に渡って、数千人ものユーザーを対象としたニーズをモデル化するのはほとんど不可能であること、と相まってさらに困難なことであることを意味します。

アクセス保証とは、業務関係者に定期的に資格の審査を要求し、資格が適切であることを承認するか、削除すべきであるとのフラグを立てる、などの処理を促すためのプロセスです。

アクセス保証にはいくつかのコンポーネントがあります。:

- **ディスカバリ:**
資格がレビューされる前、システムとアプリケーションから収集されユーザーへのマッピングを行う必要があります。技術的な識別子は、審査者が理解できるように、人が読める記述に置きかえなくてはなりません。資格は常に変更されるため、ディスカバリーは一回のデータロードではなく、定期的に自動化プロセスで行われる必要があります。

➤ 誰が審査を行うか？

オプションとして次の管理者があります。・ 所属員をレビューを依頼される管理者、アプリケーションやデータをアクセスするユーザーのリストのレビューを依頼される管理者、または、高リスクの資格のレビューを依頼されるセキュリティ管理者

➤ 審査をいつ実施するか？

実行頻度は、疑問視される資格のビジネスリスクの有無によって変わります。

➤ どんな資格が審査されるか？


最も上位のレベルの審査は、従業員ステータスです。--ユーザーはこのシステムへのアクセスがまだ出来ていてよいか？ もう少し粗いのは、ロールのレビュー--ユーザーはこれらのロールを持っていてよいか？ 最も低レベルのものは基本資格のレビュー--ユーザーは、このシステムのログインIDを持っていてよいか、または、このセキュリティグループに属してよいか？

➤ どのような資格を審査すべきか？

すべての資格が重大なビジネスリスクを引き起こすわけではありません。例えば、ソーシャルコミティ・メーリングリストのユーザーメンバーシップなどは、レビューをするに値しません。各資格によって生じるリスクレベルにより、レビューするか否か、審査するとしたら、どのくらいの頻度で行うかの判断が必要です。

➤ 却下された資格はどうなるか？

審査者は、不適切な資格について、何か対処するためにフラグを立てます。IT問題管理システムに問題を提起するか、即座に資格を停止するようにコネクタートリガーを掛けるか？ 資格を停止する前にさらに追加審査を行うか？

アクセス保証のベストプラクティスは下記です。: 


- アクセス保証と資格のクリーンアッププロセスを、ロール定義を始める前に最低一回実施します。クリーンアップは、同じセキュリティ資格を持つユーザー集合の特定を容易にします。
- 統合システムから定期的に資格情報を取り出します。・ 例: 24時間毎。これにより、すでに無効となったデータではなく、現在の実際の資格を審査することができます。
- すべての資格に対して、明確で、ビジネスフレンドリーな記述を優先して定義します。
- ロールが定義されると、審査者は、保証プロセスにおいて、詳細な(より多くの)資格を審査するのではなく、割り当てられたロールを審査すればよくなります。
- 小規模なアプリケーションでは、ユーザーのアクセスレビューについてアプリケーション所有者に依頼します。
- 大規模なアプリケーションでは、マネージャーにその所属員のアクセスのレビューを依頼します。なぜなら、アプリケーション所有者はユーザーを個人的に知らないためです。
- マネージャーに、定期的(例えば、四半期毎)にすべてのユーザーの雇用状態を検証を依頼し、孤立ユーザーを取り除きます。

一般的にな中～大規模の組織には、何百ものシステムやアプリケーションへのアクセスが必要な何千人ものユーザーが存在します。

一つのアプリケーションのユーザー・プロビジョニング・システムへのインテグレーションがいかに短時間ででき、低コストだったとしても・ 例えインテグレーション、テスト、IDマッピング等の処理に1日位の工数が掛かる・ すべてのシステムやアプリケーションに統合するには、何百人日が掛かることとなります。これらのインテグレーションが完了するのを待ってから、ユーザー・プロビジョニング・システムを運用開始するのでは、システムの価値を発揮するまでの時間とコストが掛かり過ぎます。

ほとんどの組織のシステムとアプリケーションは、少数の広範に利用されるシステムとアプリケーションと、少数のユーザーしか利用しない何百ものアプリケーションの組み合わせです。:

- **エンタープライズ・システム:** ディレクトリ、メインフレーム、ERPシステム、e-mail システム等があります。これらは、10とか20以上の規模になることはまれです。
- **小～中規模アプリケーション:** 一般に、部門、特定システムで、財務、予算システム、エンジニアリング・アプリケーション、やIT部門内で用いられるシステム等です。

ユーザー・プロビジョニング・システムの効果を最大限にして、システムの入手から実運用までの時間を最短にするには、次をするのが現実的です。: 

- 展開の初期段階で、主要なエンタープライズ規模のシステムとアプリケーションを統合する。
- ユーザー・プロビジョニング・システムの中で小規模アプリケーションへのアクセス申請をサポートし、統一した申請/承認プロセスをポリシー施行とともにすべてのアプリケーションに適用する。
- (人の)システム管理者に未統合のアプリケーションアクセス申請の施行を依頼する。
- 段階的に申請ボリュームの多い順にアプリケーション統合を追加して行く。

このアプローチでは、申請者、承認者の双方に“ワンストップショッピング”を適用し、組織には、統一された承認、監査、SoD、アクセス保障プロセスをすべてのシステムに対して、統合状況の如何に関わらず提供することができます。

これをもう一段階進めると、出来るだけ早く、出来る限り多くのアプリケーションに対し限定したインテグレーションを進めることが現実的であることがわかります。実際には、ユーザー・プロビジョニング・システムを各アプリケーションから自動的にユーザーをリストアップするように構成し、どのユーザーが既に各システムにアクセス権を持っているかといった最新情報を持つようにします。これはSoDとアクセス保証プロセスに意味を持たせるために大変重要です。そうしないとにも保証できなくなります。

一つの疑問は、ユーザーのプロビジョンまたは、ディプロビジョンをするために、システム管理者に依頼をするワークフロープロセスはどこにあればよいでしょうか？一つのオプションは、現在使っているIT管理プラットフォーム、例えば、BMC RemedyやHPサービスマネージャーにワークフロープロセスを置くことが考えられます。別のやり方は、ユーザー・プロビジョニング・システムの中で、アクションの申請をトラックし、タスクを受け付け、完了指示できるようにすることです。

どちらのアプローチも可能ですが、いずれのケースでもプロセスが次の項目をサポートする必要があります。: 

- アプリケーション毎のシステム管理者の存在
- 応答しない管理者に対するリマインダー機能
- 応答しない管理者から他へのエスカレーション機能
- システムが生成するユニークIDなど、ユーザーの生成/削除プロセス中の管理者からのフィードバック情報提供機能

ユーザープロビジョニングシステムはユーザーサービスを改善し、ネットワークセキュリティを強化することによって、ITサポートコストを低減する付加価値を提供します。これは、下記事項により可能です。:

- 一つのシステムから他へのアイデンティティと資格の変更を伝搬する(アイデンティティ同期、自動プロビジョニング及び自動ディプロビジョニング)。
- ビジネスユーザーに資格の変更の申請、審査、許可を可能にする。
- ロール、SoDルール、標準のネーミング規約、等を用いてセキュリティポリシーを施行する。
- ユーザーや資格に関する現在及び履歴情報をレポートする。

このドキュメントで説明したベストプラクティスを用いることによって、企業/組織はユーザープロビジョニングシステムの展開に於いて、付加価値を最大化し、コストを最小限に抑えることができます。

 Hitachi ID Systems, Inc.

500, 1401 - 1 Street SE, Calgary AB Canada T2G 2J3 Tel: 1.403.233.0740 Fax: 1.403.233.0735 E-Mail: sales@Hitachi-ID.com

www.Hitachi-ID.com