CA SiteMinder®

Web エージェント設定ガイド

r12.0 SP3

technologies

CA 製品リファレンス

このマニュアルが参照している CA の製品は以下のとおりです。

- CA SiteMinder®
- CA Wily Introscope®
- CA Identity Manager
- CA SOA Security Manager

CA への連絡先

テクニカル サポートの詳細については、弊社テクニカル サポートの Web サイト（http://www.ca.com/jp/support/）をご覧ください。

目次

第 1 章: Web エージェント

このセクションには、以下のトピックが含まれています。

Web エージェントがリソースを保護する方法

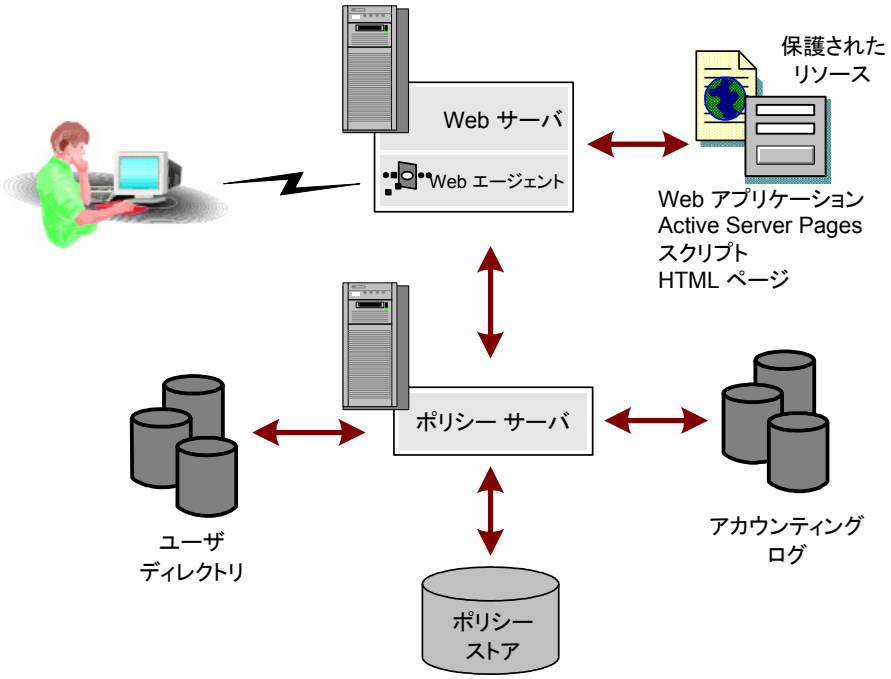
SiteMinder Web エージェントは、URL によって識別できる任意のリソースへのアクセスを制御するソフトウェア コンポーネントです。Web エージェントは Web サーバに常駐し、リソースの要求をインターセプトして、そのリソースが SiteMinder によって保護されているかどうかを判断します。その後、Web エージェントはポリシー サーバと連携し、保護された Web サーバ リソースへのアクセスを要求するユーザの認証および許可を行います。

Web エージェントは以下のタスクを実行します。

- 保護されているリソースへのアクセスリクエストをインターセプトし、ポリシー サーバと連携して動作し、ユーザがアクセス権を持っているかどうかを判断します。
- ユーザに対するコンテンツの提示方法 (ポリシー ベースのパーソナライゼーション) とアクセス権限の配信方法を指示する Web アプリケーションに情報を提供します。
- ユーザが情報に迅速かつ安全にアクセスできるようにします。Web エージェントはユーザ アクセス権限に関するコンテキスト情報をセッション キャッシュに格納します。キャッシュ設定値を変更すると、パフォーマンスを最適化できます。
- 単一の cookie ドメインまたは複数の cookie ドメインにある複数の Web サーバ間でシングル サインオンを有効にして、ユーザの再認証を不要にします。

SiteMinder Web エージェントおよびサポートされている Web サーバ プラットフォームの一覧については、「テクニカル サポート」にアクセスし、SiteMinder サポート マトリックスを検索してください。

Web エージェントは、以下の図に示すように、Web サーバ上にあります。



Web エージェントとポリシー サーバが連携する仕組み

アクセス制御を実行する場合、Web エージェントは、ポリシー サーバと対話を行います。実際に許可と認証の判断を行うのは、ポリシー サーバです。

Web エージェントはリソースに対するすべてのユーザ リクエストをインターセプトし、要求されたリソースが保護されているかどうかをポリシー サーバに確認します。リソースが保護されていない場合は、アクセス要求は Web サーバに直接渡されます。リソースが保護されている場合、次の動作が発生します。

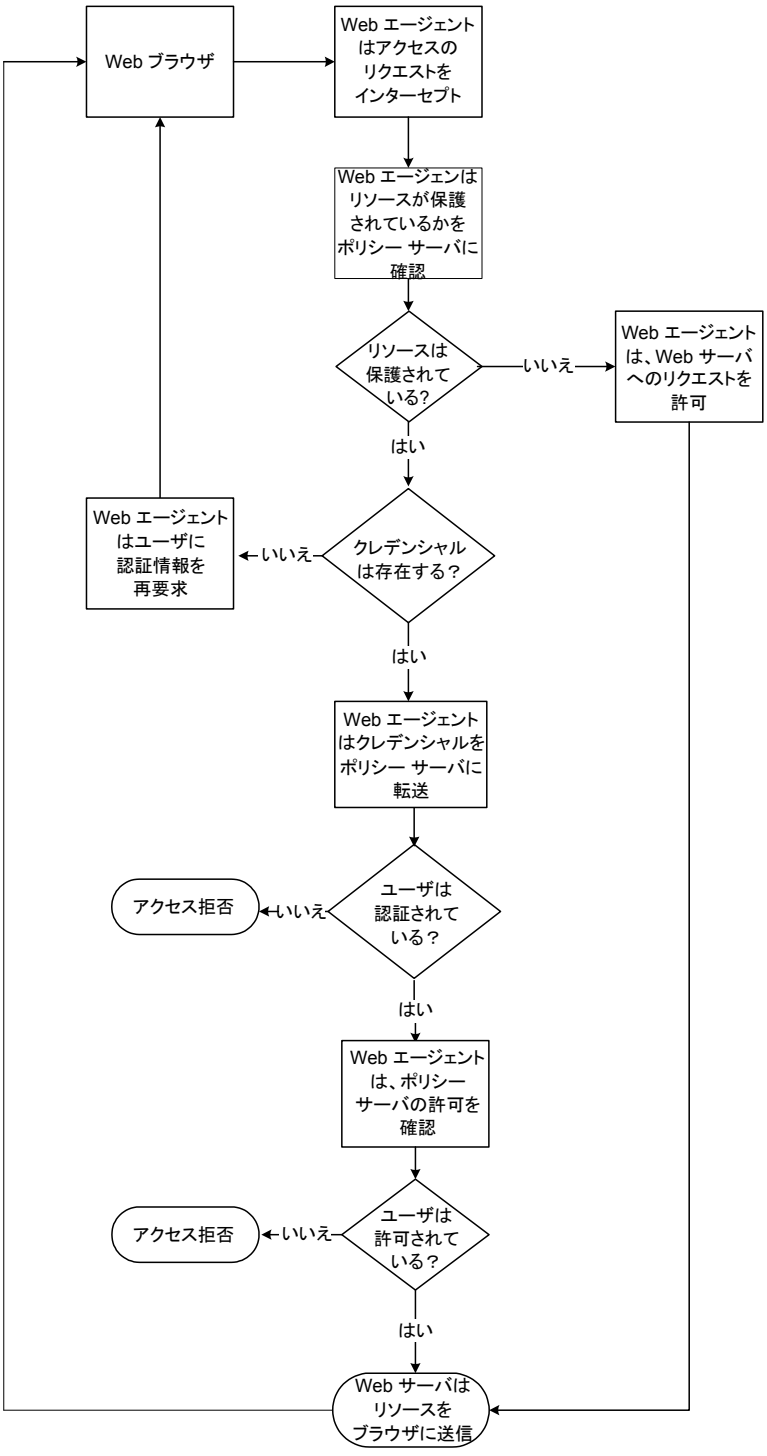
1. Web エージェントは、そのリソースに関して、どの認証方法が必要とされているのかチェックします。一般的な認証情報は名前とパスワードです。しかし、証明書、トークンカードの PIN（個人用識別番号）のような他の認証情報が必要になる場合もあります。
2. Web エージェントは、認証情報の入力をユーザに要求します。

   ユーザはそれに応答し、適切な認証情報をレスポンスとして返します。
3. Web エージェントは、これらの認証情報をポリシー サーバに渡します。ポリシー サーバは、その認証情報が正しいかどうかを判断します。
4. ユーザが認証フェーズに合格した場合、ポリシー サーバは、ユーザがそのリソースにアクセスすることを許可されているかどうか判断します。ポリシー サーバがアクセスを許可した後、Web エージェントは、その要求を Web サーバに渡します。

Web エージェントは、ユーザ固有の属性も*レスポンス*の形式で受信します。これにより、Web コンテンツのパーソナライズ機能とセッション管理が可能になります。レスポンスは、ユーザの許可後にポリシー サーバから Web エージェントに返されるパーソナライズされたメッセージやユーザ固有の情報です。レスポンスは、名前/値という属性ペアで構成されています。この属性ペアは、Web アプリケーションで使用するために、Web エージェントが HTTP ヘッダに追加したものです。レスポンスの例には、以下のようなものがあります。

- Web エージェントは、Web アプリケーションへのアクセスをユーザに許可した後、ユーザ セッションの持続時間を指示する情報も Web アプリケーションに送信できます。
- また、以前に登録したサイトに再びアクセスした場合に、Web エージェントはユーザの購買志向に関する情報を返すこともできます。

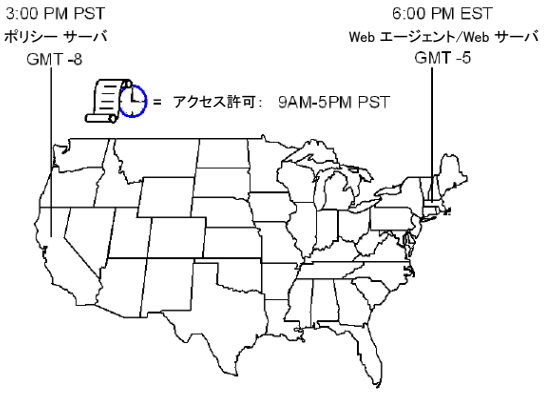
以下の図は、Web エージェントとポリシー サーバの間の通信を示しています。


Web ブラウザ
Web エージェントはアクセスのリクエストをインターセプト
Web エージェンはリソースが保護されているかをポリシー サーバに確認
リソースは保護されている?
いいえ
Web エージェントは、Web サーバへのリクエストを許可
はい
Web エージェントはユーザに認証情報を再要求
いいえ
クレデンシャルは存在する?
はい
Web エージェントはクレデンシャルをポリシー サーバに転送
アクセス拒否
いいえ
ユーザは認証されている?
はい
Web エージェントは、ポリシー サーバの許可を確認
アクセス拒否
いいえ
ユーザは許可されている?
はい
Web サーバはリソースをブラウザに送信

異なるタイムゾーンにある Web エージェントとポリシー サーバについての考慮事項

デフォルトでは、ポリシー サーバと Web エージェントは GMT (グリニッジ標準時)を基準にして時間を計算します。したがって、ポリシー サーバまたは Web エージェントがインストールされている各システムのシステム クロックを、その地域のタイム ゾーンに基づいて設定しておく必要があります。

以下の図に、ポリシー サーバが時間を基準にポリシーをどのように実行するかを示します。リソースは、マサチューセッツにある Web サーバに格納されていて、カリフォルニアにあるポリシー サーバで保護されています。このポリシーでは、午前 9 時から午後 5 時までの間のリソースへのアクセスを許可します。ただし、マサチューセッツのユーザは午後 6 時でもリソースにアクセスできます。これは、このポリシーがポリシー サーバのタイムゾーンである PST(太平洋標準時)に基づいており、PST は Web エージェントのタイムゾーンである EST(東部標準時)よりも 3 時間遅れているためです。



ポリシー サーバのタイムゾーンでは午後 3 時なので、マサチューセッツのユーザは午後 6 時でもリソースにアクセスすることが可能。

注: Windows システムでは、タイムゾーンと時刻([日付と時刻]コントロール パネルで設定)の両方が整合している必要があります。たとえば、米国でシステムを東部標準時から太平洋標準時にリセットするには、以下の順にタスクを実行します。

a. タイムゾーンを太平洋標準時に設定します。

b. システム クロックに正しい時間(東部標準時より 3 時間早い時間)が表示されていることを確認します。

これらの設定値が整合していない場合、複数のドメイン間でのシングル サインオンや、エージェント キー管理機能が正しく動作しません。

エージェントが SiteMinder の cookie を読み取る方法

Web エージェントは、エージェント キーを使用して、SiteMinder の cookie の暗号化と復号化を行い、cookie に格納されているデータを読み取ることができるようにします。エージェントはそのキーを使用して cookie を暗号化した後、その暗号化済み cookie をユーザのブラウザへ送信し、他の Web エージェントから受信した cookie を復号化します。

すべての Web エージェントが同じキーを知っている必要があります。また、1 つのポリシー サーバと通信を行うすべてのエージェントのキーを、同じ値に設定する必要があります。このルールは、シングル サインオン環境にあるエージェントの場合、特に重要です。キーの安全を確保するため、ポリシー サーバはキーの「*ロール オーバー*」を実行します。キーのロールオーバーとは、新しいキーを生成して暗号化し、SiteMinder 環境内のすべての Web エージェントにそれらのキーを配布することです。

Web エージェントが起動し、管理呼び出し(リクエスト)を行った時点で、ポリシー サーバは現在のキーセットを提供します。Web エージェントは、ポリシー サーバをポーリングするたびに、管理呼び出しを繰り返します。Web エージェントは、更新済みのキーを受け取ります。

ポリシー サーバは、以下のタイプのキーを提供します。

ダイナミック キー

ポリシー サーバのアルゴリズムにより生成され、接続された他のポリシー サーバや関連する Web エージェントに配布されるキーです。ダイナミック キーは、一定の間隔で自動的にロールオーバーできます。また、管理 UI を使用して手動で変更することもできます。

スタティック キー

常に同一であるキーです。ポリシー サーバのアルゴリズムによって生成するか、手動で設定することができます。SiteMinder では、cookie に情報を長期間保存する必要のある機能のサブセットに、このタイプのキーを使用します。

自動キー変換を使用すると、1 つのキー ストアを共有する大規模な SiteMinder インストール環境でエージェント キーの管理プロセスを簡易化することができます。キー ストアとは、すべてのキー情報のストレージ ロケーションです。ポリシー サーバは、このキー ストアにアクセスして現在のキーを取得し、そのキーが Web エージェントに渡されます。シングル サインオンを設定されたエージェントの場合、キーストアを複製して、シングル サインオン環境のすべてのポリシー サーバでキーストアを共有する必要があります。自動キー変換により、キーの完全性も確保されます。

注: 詳細については、ポリシー サーバ ドキュメントを参照してください。

Web エージェントとダイナミック キーのロールオーバー

管理 UI を使用して、ダイナミック エージェント キーのロールオーバーを設定することができます。Web エージェントは、キーの更新があるかどうかポリシー サーバを定期的にポーリングします。キーが更新されている場合、Web エージェントはポーリング時に変更内容を取得します。デフォルトのポーリング時間は 30 秒ですが、この値は、Web エージェントの PSPollInterval パラメータの値を変更することで、カスタマイズできます。

Web エージェントは、キーのロールオーバーが発生したことを検出した時点で、以下のエージェントキーの新しい値を取り出します。

前回キー

現在の値の前にダイナミック エージェント キーに使用していた最後の値が入ります。

現在キー

現在のダイナミック エージェント キーの値が入ります。

予定キー

ダイナミック エージェント キーのロールオーバーで*現在*キーとして使用する次回の値が入ります。

スタティック キー

エージェントが、ユーザを識別してその情報を長期間保存する必要がある SiteMinder 機能に使用できる、長期間キーが入ります。ダイナミックキーが使用できない場合、スタティックキーは、シングル サインオンに関連して cookie の暗号化もサポートします。

Web エージェントでは、cookie データを保持したり、古いキーから新しいキーへスムーズに移行するために、複数のキーが必要です。

キーストア

ポリシー サーバは、生成したダイナミック キーを、キー ストアに保存して管理します。キー ストアはリポジトリであり、すべてのポリシー サーバは最新のキーをここから取得します。Web エージェントは、ポリシー サーバから現在キーを取得します。キー ストアは、SiteMinder ポリシー ストアの一部に組み込むことも、スタンドアロン キー ストアとして保持することもできます。

注: 管理者が、エージェント キーの複数のロールオーバーを短時間に続けて実行した場合、このアクションにより、シングル サインオン用のすべての cookie が無効になり、現在ログイン中のすべてのユーザのシングル サインオンが無効になる可能性があります。これらのユーザが再認証されると、シングル サインオンは正常に動作するようになります。

Web エージェントのタイプ(トラディショナルおよびフレームワーク)

すべての SiteMinder Web エージェント、2 つのアーキテクチャのどちらかに基づいています。トラディショナル Web エージェントは、オリジナルの SiteMinder エージェント アーキテクチャに基づいています。フレームワーク エージェントは、SiteMinder バージョン 5.x QMR 6 で導入されました。Web エージェントにより提供される機能はアーキテクチャにかかわらず基本的に同じです。ただし、いくつか異なる点があります。たとえば、フレームワーク エージェントは、トラディショナル Web エージェントでは使用されていない、異なる WebAgent.conf ファイルと LocalConfig.conf ファイルを使用します。

トラディショナル Web エージェントは、以下の Web サーバにインストールされます。

- Domino

フレームワーク エージェントは、以下の Web サーバにインストールされます。

- IIS 6.0
- Apache 2.0
- Apache 2.0 ベースのサーバ: IBM HTTP Server、HP Apache サーバ
- Oracle iPlanet Web サーバのバージョン 6.0 以降

  注: Oracle iPlanet Web Server は、以前は Sun Java Systems Web サーバまたは SunONE Web サーバと呼ばれていました。

詳細情報:

ユース ケース展開の前提条件を満たす方法 (P. 47)
フレームワーク エージェントと従来のエージェントの間の POST 維持の有効化 (P. 283)

変更時にサーバの再起動を必要とするパラメータ

一部のエージェント パラメータは、動的に更新されます。以下のパラメータに変更を適用した場合は、Web サーバを再起動する必要があります。

AgentConfigObject

ローカル エージェント設定ファイル内にエージェント設定オブジェクト (ポリシー サーバに格納された) の名前を定義します。このパラメータはエージェント設定オブジェクトでは*使用されません*。

デフォルト: デフォルトなし

CacheAnonymous

Web エージェントが匿名のユーザ情報をキャッシュするかどうかを指定します。このパラメータは、たとえば以下の状況に対して設定できます。

- Web サイトのユーザのほとんどが匿名ユーザで、それらのユーザのセッション情報を格納したい場合。
- 登録ユーザと匿名ユーザの両方が Web サイトにアクセスする場合。

  匿名ユーザの情報のみでキャッシュが満杯になり、登録ユーザ用の領域がなくなってしまう可能性がある場合は、このパラメータを無効にすることをお勧めします。

デフォルト: No

HostConfigFile

トラステッド ホスト コンピュータがポリシー サーバに正常に登録された後に作成される SMHost.conf ファイル (IIS 6.0 または Apache のエージェント内) のパスを指定します。コンピュータ上のすべての Web エージェントが SMHost.conf ファイルを共有します。

デフォルト: デフォルトなし

MaxResourceCacheSize

Web エージェントがそのリソース キャッシュ内で保持するエントリの最大数を指定します。エントリには以下の情報が含まれます。

- リソースが保護されるかどうかに関するポリシー サーバのレスポンス
- レスポンスで返される追加属性

最大値に達すると、新しいリソース レコードが最も古いリソース レコードと置き換わります。

これらをより大きな数値に設定する場合は、十分なシステム メモリがあることを確認してください。

OneView モニタを使用して Web エージェント統計を表示している場合は、ResourceCacheCount に表示される値が MaxResourceCacheSize パラメータで指定された値より大きいことがあります。これはエラーではありません。Web エージェントは、MaxResourceCacheSize パラメータを 1 つのガイドラインとして使用します。また、値は状況により異なります。これは、MaxResourceCacheSize パラメータはリソース キャッシュ内の平均サイズのエントリの最大数を示すためです。実際のキャッシュ エントリは、あらかじめ識別された平均サイズより大きかったり小さかったりする可能性があります。したがって、実際の最大エントリ数は指定された値より多い場合や少ない場合があります。

注: フレームワーク エージェントなど、共有メモリを使用する Web エージェントの場合、キャッシュは MaxResourceCacheSize の値に基づいて一定サイズが事前に割り当てられ、それより増えることはありません。

デフォルト:（Domino Web サーバ）1000

デフォルト:（IIS および Sun Java System Web サーバ）700

デフォルト:（Apache Web サーバ）750

MaxSessionCacheSize

エージェントがそのセッション キャッシュ内で保持するユーザの最大数を指定します。セッション キャッシュには、認証するユーザのセッション ID が正常に格納されます。それらのユーザが同じセッション中に同じレルム内の別のリソースにアクセスした場合、エージェントはポリシー サーバをコールする代わりにセッション キャッシュの情報を使用します。この最大数に達すると、エージェントは最も古いユーザ レコードを新しいユーザ レコードと置き換えます。

このパラメータの値は、持続期間にリソースにアクセスしてそれを使用する予定のユーザの数に基づいて設定します。これらをより大きな数値に設定する場合は、十分なシステム メモリがあることを確認してください。

デフォルト:（Domino Web サーバ）1000

デフォルト:（IIS および Oracle iPlanet Web サーバ）700

デフォルト:（Apache Web サーバ）750

PostPreservationFile

以下の POST 維持テンプレート ファイルの*いずれか*に対するパスを指定することで、トラディショナル エージェントとフレームワーク エージェントとの間の POST 維持データの転送を有効にします。

- tr2fw.pptemplate - トラディショナル エージェントが稼働しているサーバでホストされているリソースが、フレームワーク エージェント上で実行されている FCC によって保護されていることを示します。
- fw2tr.pptemplate - フレームワーク エージェントが稼働しているサーバでホストされているリソースが、トラディショナル エージェント上で実行されている FCC によって保護されていることを示します。

デフォルト: デフォルトなし

例: *web_agent_home*/samples/forms/fw2tr.pptemplate

ResourceCacheTimeout

リソース エントリがキャッシュに保存される秒数を指定します。時間間隔の値を超えると、Web エージェントはキャッシュされたエントリを削除します。その後、保護されているリソースにユーザがアクセスしようとすると、Web エージェントはポリシー サーバに問い合わせます。

デフォルト: 600（10 分）

詳細情報:

中央設定の実装 (P. 30)

Web エージェントの設定

Web エージェントの設定タイプによって、Web エージェントがそのパラメータの値を取得する方法が決定します。以下のいずれかの方法（または両方の組み合わせ）を使用して、Web エージェントを設定することができます。

中央

ポリシー サーバ上のエージェント設定オブジェクトからパラメータ値を受け取ります。

ローカル

Web サーバをホストしているシステムにインストールされているファイルからパラメータ値を受け取ります。

Web エージェントは以下の手順に従ってその構成設定を読み取ります。

1. Web エージェントは有効になった時点で、（ポリシー サーバ上の）エージェント設定オブジェクトを検索し、設定情報を取得します。
2. Web エージェントは、AllowLocalConfig パラメータの値を確認します。
3. AllowLocalConfig を no に設定した場合、Web エージェントはエージェント設定オブジェクトからその構成設定をすべて取得します。AllowLocalConfig を yes に設定した場合、Web エージェントは、変更されたパラメータまたは追加のパラメーがないか、対応するエージェントのローカル設定ファイルを検索します。ローカル設定ファイル内の設定は、エージェント設定オブジェクト内の設定を上書きします。
4. Web エージェントは、集中的なソースおよびローカル ソースの設定を使用して、エージェント設定オブジェクトの統合されたローカル コピーを作成します。このローカル コピーは、ポリシー サーバ上の元のエージェント設定オブジェクトに変更を加えることはありません。

詳細情報

中央設定とローカルの設定の組み合わせ (P. 40)

中央設定

中央エージェント設定では、ポリシー サーバのエージェント設定オブジェクトから 1 つ以上の Web エージェントを管理します。ポリシー サーバにあるエージェント設定オブジェクトには、Web エージェントが使用するパラメータが入っています。中央設定の 1 つの利点は、複数のエージェントのパラメータ設定を同時に更新できることです。ほとんどのパラメータ変更は動的に発生しますが、フレームワークのパラメータの中には、変更後に Web サーバの再起動を必要とするものもあります。

エージェント設定オブジェクトの作成および編集には、管理 UI を使用します。ポリシー サーバと通信するそれぞれの Web エージェントは、エージェント設定オブジェクトに関連付ける必要がありますが、複数の Web エージェントで 1 つのエージェント設定オブジェクトを使用することができます。

注: エージェント設定オブジェクトの作成方法の詳細については、ポリシー サーバのマニュアルを参照してください。

中央設定の実装

中央設定はデフォルトで有効になっています。Web エージェントでは、設定ウィザードで Web エージェントを設定したときに指定した既存のエージェント設定オブジェクトの構成設定が使用されます。パラメータの設定は、必要に応じていつでも変更できます。

中央設定を実装する方法

1. 管理 UI にログインします。

   セットアップ画面が表示されます。

2. [インフラストラクチャ]-[エージェント設定]をクリックします。

   エージェント設定タスクのリストが表示されます。

3. [エージェント設定の変更]をクリックします。

   [検索]ウィンドウが表示されます。

4. (オプション)ユーザの検索条件を絞り込むために検索フォームに入力します。

5. [検索]をクリックします。

   エージェント設定オブジェクトのリストが表示されます。

6. 中央設定を実装するエージェントと関連付けられたエージェント設定オブジェクトの左側にあるラジオ ボタンをクリックします。[選択]をクリックします。

   [エージェント設定の変更]ウィンドウが表示されます。

7. AllowLocalConfig パラメータの値が no に設定されていることを確認します。

8. 必要に応じて、管理 UI を使用して他のパラメータの設定を追加、編集、または削除します。

9. 変更が終了したら、[サブミット]をクリックします。

   [エージェント設定の変更]ウィンドウが閉じ、確認メッセージが表示されます。中央設定が実装されます。 ほとんどのパラメータは動的に変更されますが、一部の変更を有効にするには Web サーバの再起動が必要です。

ローカル エージェント設定

ローカル設定

ローカル エージェント設定では、Web サーバをホストしているシステム上にインストールされているローカル ファイルを使用して、Web エージェントを管理します。ローカル ファイル内のパラメータ設定は、ポリシー サーバ上のエージェント設定オブジェクトに格納されているすべての設定に優先します。エージェント設定オブジェクト内の設定は変更されません。ローカル エージェント設定を考慮する状況には以下があります。

- たとえば、Apache Web エージェントが 3 つあり、最初の 2 つ(A と B)は同一のパラメータ設定を使用しており、3 つ目の Apache エージェント(C)は、A と B の設定の大半を使用する一方で、リバース プロキシとして動作している場合。これを実行するには、Apache エージェント A および B のセントラル エージェント設定を使用し、Apache エージェント C にはローカル設定を使用します。

- ポリシー サーバ管理者がエージェントを設定する同一人物(またはグループ)でない場合。たとえば、社内の IT 部がポリシー サーバをメンテナンスしているけれども、財務部がエージェントを使用して会計アプリケーションへのアクセスを制御している場合。IT 部の担当者は、ポリシー サーバ上でエージェントのローカル設定を実行できるのに対し、財務部の担当者は、会計アプリケーションを保護するエージェントの特定の設定を制御します。

フレームワーク Web エージェントでは、ローカル設定に以下のファイルを使用します。

WebAgent.conf

ポリシー サーバの起動および接続にフレームワーク Web エージェントが使用する中心的な設定が含まれます。

LocalConfig.conf

フレームワーク Web エージェントの設定が含まれます。

トラディショナル Web エージェントでは、ローカル設定に以下のファイルを使用します。

WebAgent.conf

従来の Web エージェントの設定がすべて含まれます。

WebAgent.conf ファイルのロケーション

以下の表に、各種の Web サーバ上で WebAgent.conf ファイルが作成される場所を示します。

Web サーバ	ファイルの場所
IIS	Program Files¥ca¥webagent¥bin¥IIS
Oracle iPlanet（iPlanet/SunOne）	*Oracle_iPlanet_server_home*/https-*hostname*/config *Oracle_iPlanet_home* は、Oracle iPlanet Web サーバのインストール場所で、*hostname* はサーバの名前です。
Apache IBM HTTP Server Oracle HTTP Server	*web_server_home*/conf *web_server_home* は、Web サーバのインストール場所です。
Domino	Windows：c:¥lotus¥domino UNIX：$HOME/notesdata

フレームワーク エージェントの WebAgent.conf ファイル

AgentConfigObject、HostConfigFile、および EnableWebAgent の各パラメータのほかに、以下のパラメータもフレームワーク エージェントの WebAgent.conf ファイルに追加されます。

重要: Web エージェント以外の他の SiteMinder 製品を参照しているファイルのセクションは変更しないでください。ただし、ファイル内の Web エージェント パラメータの値は変更できます。

LocalConfigFile

LocalConfig.conf ファイルのロケーションを指定します。このファイルに、エージェント設定の大半が含まれます。

ServerPath

エージェントに対して Web サーバ（Apache 2.0 および Oracle iPlanet Web サーバ）のディレクトリを特定します。

LoadPlugin

IIS 6.0 エージェントおよび Apache 2.0 エージェントに対してロードするプラグインを指定します。プラグインはさまざまな種類のエージェント機能をサポートします。以下のプラグインを使用できます。

HttpPlugin

Web エージェントが HTTP エージェントとして動作するかどうかを指定します。

デフォルト: Enabled

SAMLAffiliatePlugin

Web エージェントと SAML アフィリエイト エージェントの間の通信を許可します（Federation セキュリティ サービスを購入している場合）。

デフォルト: Disabled

Affiliate10Plugin

Web エージェントと 4.x アフィリエイト エージェントの間の通信を許可します。これは SAML アフィリエイト エージェントでは使用されません。

デフォルト: Disabled

他の LoadPlugin エントリを有効にするには、行の先頭からポンド記号（#）を削除します。

詳細情報

複数の Web サーバ インスタンスを持つ Web エージェントの管理 (P. 62)

LocalConfig.conf ファイルの場所（フレームワーク エージェント）

フレームワーク Web エージェントをインストールすると、SiteMinder インストール プログラムは、以下のディレクトリに LocalConfig.conf ファイルを作成します。

Windows

web _agent_home¥config

UNIX

web _agent_home/config

重要: このファイルにはすべてのデフォルト設定が含まれています。このファイルを直接変更しないでください。後で参照したりリカバリしたりするために、このファイルのバックアップ コピーを作成することをお勧めします。

Web エージェントを設定するときに、設定ウィザードにより、LocalConfig.conf ファイルは以下のディレクトリにコピーされます。

IIS Web サーバ

web_agent_home¥bin¥IIS

Oracle iPlanet Web サーバ

Oracle_iPlanet_home/https-*hostname*/config

Apache Web サーバ

Apache_home/conf

Web エージェントは、LocalConfig.conf ファイルのこのコピーからその設定を取得します。

ローカル設定ファイルのみにあるパラメータ

中央エージェント設定の場合、ローカル設定ファイル内のパラメータの大半は、エージェント設定オブジェクトにも含まれています。以下のパラメータは、ローカル設定ファイルのみで使用され、エージェント設定オブジェクトには*ありません*。

AgentConfigObject

ローカル エージェント設定ファイル内にエージェント設定オブジェクト（ポリシー サーバに格納された）の名前を定義します。このパラメータはエージェント設定オブジェクトでは*使用されません*。

デフォルト: デフォルトなし

EnableWebAgent

Web エージェントをアクティブにし、それがポリシー サーバと通信することを可能にします。すべての設定パラメータの変更を完了してから、このパラメータを yes に設定します。

デフォルト: No

HostConfigFile

トラステッド ホスト コンピュータがポリシー サーバに正常に登録された後に作成される SMHost.conf ファイル（IIS 6.0 または Apache のエージェント内）のパスを指定します。コンピュータ上のすべての Web エージェントが SMHost.conf ファイルを共有します。

デフォルト: デフォルトなし

エージェント設定ファイルを編集する方法

エージェント設定ファイルは、ローカルで設定された Web エージェントの設定を制御します。それらの設定を変更するには、以下の手順に従います。

1. WebAgent.conf（従来のエージェントの場合）または LocalConfig.conf ファイル（フレームワーク エージェントの場合）のバックアップ コピーを作成します。
2. テキスト エディタで元のエージェント設定ファイルを開きます。
3. 以下のいずれかを実行することによって、パラメータを有効または無効にします。
   - パラメータを有効にするには、行の先頭からポンド記号（#）を削除する。
   - パラメータを無効にするには、行の先頭にポンド記号（#）を追加する。
4. 以下のガイドラインを使用して、パラメータの値を変更します。
   - パラメータ名、等号（=）、パラメータ値の間に空白を挿入しないでください。
   - パラメータ値を引用符で囲みます。
   - WebAgent.conf ファイルおよび LocalConfig.conf ファイルでは、大文字と小文字が区別されません。エージェントと共にインストールされるサンプル ファイルについては、大文字小文字を一致させる必要はありません。
   - 多くの値は、ファイル内で <Agent Name>,<IPAddress> のようにわかりやすい変数の形で記載されています。山形かっこ <> とテキストの両方を、希望の値に置き換えます。
   - 値が空白の場合、空白はデフォルトとして有効です。パラメータの先頭にポンド記号（#）が記述されていない場合に限って、デフォルト値が適用されます。
5. 変更作業が完了し、ファイルを保存して閉じてから、EnableWebAgent を yes に設定します。

   これにより、ローカルでのすべての設定変更が有効になります。エージェントが有効になった後で、さらに変更を加えた場合は、それらの変更結果を有効にするために Web サーバを再起動する必要があります。

ローカル設定の実装

以下のパラメータを使用して、ローカル設定が許可されるかどうかを制御できます。

AllowLocalConfig

ローカル設定ファイルを読み取って Web エージェントの設定パラメータを取得するように、ポリシー サーバ上のエージェント設定オブジェクトに指示します。このパラメータはエージェント設定オブジェクトでのみ使用されます。

このパラメータの複数の値をエージェント設定オブジェクトに追加して、ローカル設定ファイルで変更可能なパラメータを制御することもできます。複数の値がこのパラメータに設定される場合、それらは以下の順序で処理されます。

- 同じエージェント設定オブジェクト内の他の設定パラメータに yes の値が存在する場合は、yes が優先されます。他の設定パラメータはローカル設定ファイル内で変更できる唯一のパラメータです。
- 同じエージェント設定オブジェクト内の他の設定パラメータに no の値が存在する場合は、no が優先されます。これにより、エージェント設定オブジェクトの他の設定パラメータのいずれも削除せずに、迅速かつ完全にローカル設定を無効にすることができます。
- yes と no の複数の値が同じエージェント設定オブジェクトに存在する場合は、no が優先されます。これにより、エージェント設定オブジェクトの他の設定パラメータのいずれも削除せずに、迅速かつ完全にローカル設定を無効にすることができます。

デフォルト: No

例: yes、EnableAuditing、EnableMonitoring（前の 2 つのパラメータについてのみ、ローカル制御が可能です）

ローカル設定を実装する方法

1. 管理 UI にログインします。

   セットアップ画面が表示されます。

2. [インフラストラクチャ]-[エージェント設定]をクリックします。

   エージェント設定タスクのリストが表示されます。

3. [エージェント設定の変更]をクリックします。

   [検索]ウィンドウが表示されます。

4. (オプション)ユーザの検索条件を絞り込むために検索フォームに入力します。
5. [検索]をクリックします。

   エージェント設定オブジェクトのリストが表示されます。
6. ローカル設定を実装するエージェントと関連付けられたエージェント設定オブジェクトの左側にあるラジオ ボタンをクリックし、[選択]をクリックします。

   [エージェント設定の変更]ダイアログ ボックスが表示されます。
7. AllowLocalConfig パラメータの左側の矢印をクリックします。

   [パラメータの編集]ダイアログ ボックスが表示されます。
8. [値]フィールドのテキストを yes に変更し、[OK]をクリックします。

   [パラメータの編集]ダイアログ ボックスが閉じます。
9. [サブミット]をクリックします。

   確認のメッセージが表示されます。ローカル設定が有効になります。
10. Web サーバ上で該当するローカル設定ファイルを開き、対象のパラメータ設定を変更します。
11. 従来のエージェントに限り、EnableWebAgent パラメータの値を yes に設定します。
12. ローカル設定ファイルを保存して閉じます。
13. Framework エージェントに限り、以下の手順に従います。
    a. WebAgent.conf ファイルを開きます。
    b. EnableWebAgent パラメータの値を[はい]に設定します。
    c. WebAgent.conf ファイルを保存して閉じます。
14. Web サーバを再起動します。

    ローカル設定が有効になり、更新されたすべてのパラメータが変更されます。

ローカル設定パラメータの変更の制限

集中的なエージェント設定では、ローカル Web サーバ管理者によって設定可能なパラメータを制限できます。組織の SiteMinder 管理者が SiteMinder エージェントがインストールされている Web サーバの担当者と同じではない場合に、これを行うことをお勧めします。

ローカル設定パラメータの変更を制限する方法

1. 管理 UI にログインします。

   セットアップ画面が表示されます。

2. [インフラストラクチャ]-[エージェント設定]をクリックします。

   エージェント設定タスクのリストが表示されます。

3. [エージェント設定の変更]をクリックします。

   [検索]ウィンドウが表示されます。

4. (オプション)ユーザの検索条件を絞り込むために検索フォームに入力します。

5. [検索]をクリックします。

   エージェント設定オブジェクトのリストが表示されます。

6. 対象のエージェント設定オブジェクトの左側にあるラジオ ボタンをクリックします。[選択]をクリックします。

   [エージェント設定の変更]ダイアログ ボックスが表示されます。

7. AllowLocalConfig パラメータの左側の矢印をクリックします。

   [パラメータの編集]ダイアログ ボックスが表示されます。

8. [値]フィールドのテキストを yes に変更し、[複数値]ラジオ ボタンをクリックします。

9. [追加]をクリックします。

   空のフィールドが表示されます。

10. フィールドでのアクセスを許可するパラメータの名前を入力します。リスト内のそれらのパラメータのみ、ローカルで変更できます。

11. (オプション)手順 9 ～ 10 を繰り返して、さらにパラメータを追加します。

12. [OK]をクリックします。

    [パラメータの編集]ダイアログ ボックスが閉じ、[エージェント設定の変更]ダイアログ ボックスが表示されます。

13. [サブミット]をクリックします。
14. [エージェント設定の変更]ダイアログ ボックスが閉じ、確認メッセージが表示されます。変更は、次回 Web エージェントがポリシー サーバをポーリングしたときに適用されます。

中央設定とローカルの設定の組み合わせ

中央で設定したい Web エージェントの数が多いけれども、それらのうちの少数の Web エージェントの設定を他の Web エージェントの設定と変える必要がある場合は、中央とローカルの設定を組み合わせて使用することができます。

たとえば、エージェントを個別に設定せずに、SiteMinder ネットワークを介して複数の cookie ドメインのシングル サインオンを設定する必要がある場合は、すべてのエージェントに中央設定を使用して、別の設定を必要とする少数のグループにローカル設定を使用することができます。

前述の例で、エージェント設定オブジェクトの CookieDomain パラメータが example.com に設定されているとします。ただし、ネットワーク内の 1 つの Web エージェントでは、CookieDomain パラメータを .example.net に設定し、その他のすべてのパラメータにはエージェント設定オブジェクトに設定されている値をそのまま使用する必要があります。

このサンプル設定を実装する方法

1. 管理 UI を使用して、環境に必要なパラメータをすべて指定してエージェント設定オブジェクトを作成します。CookieDomain パラメータを .example.com に設定します。
2. エージェント設定オブジェクトの AllowLocalConfig パラメータを yes に設定します。
3. 1 つの Web エージェントで、CookieDomain パラメータの値として example.net を使用するように(Web サーバ上の)ローカル設定ファイルのみを変更します。他のパラメータはいずれも変更*しない*でください。

その単独のエージェントのローカル設定ファイル内の CookieDomain パラメータの値は、エージェント設定オブジェクト内の値より優先されますが、他のすべてのパラメータに関しては、エージェント設定オブジェクトによって設定値が決まります。

第 2 章: Web エージェントで使用される設定ファイル

SiteMinder Web エージェントでは、特定の設定について設定ファイルを使用します。これらの設定ファイルの一部は、Web エージェントと共に Web サーバにインストールされます。他の設定ファイルは、SiteMinder Web エージェント設定ウィザードによって作成されます。これらの Web エージェント ファイルは、Web サーバをホストするコンピュータ上にインストールされた特定の Web サーバに関連付けられます。

たとえば、Apache Web サーバを実行する 32 ビット Windows システムに Web エージェントをインストールした場合、Web エージェント設定ウィザードは、SiteMinder Web エージェントによって必要とされる変更を既存の Apache Web サーバに加えます。

エージェント接続管理設定ファイル

Web エージェント インストール ウィザードは、エージェント接続マネージャ設定ファイル(AgentConMgr.conf)を以下の場所にインストールします。

web_agent_home/config

注: 以下の例のとおり、*web_agent_home* 変数は、Web Agent のインストール場所を示します。

- Windows インストールのデフォルトの場所: C:¥Program Files¥CA¥webagent

  UNIX インストールのデフォルトの場所 : /opt/ca/webagent

このエージェント接続マネージャ設定ファイルによって、Web エージェントが動作中に接続に関する詳細なトレース ログを作成することが可能になります。

詳細情報:

Agent Connection Manager のトレース ログによる詳細なエージェント接続データの収集 (P. 203)

Connection API 設定ファイル

Connection API ファイル（conapi.conf）は、Connection API を介したサービスを設定するために使用されます。これらのサービスには OneView モニタが含まれます。

Web エージェント インストール ウィザードは、Connection API 設定ファイルを以下の場所に作成します。

web_agent_home/config

注: 以下の例のとおり、*web_agent_home* 変数は、Web Agent のインストール場所を示します。

- Windows インストールのデフォルトの場所: C:¥Program Files¥CA¥webagent

  UNIX インストールのデフォルトの場所： /opt/ca/webagent

注: OneView モニタの使用の詳細については、「DNA Always Current Scheduler」を参照してください。

ローカル エージェント設定ファイル

Web エージェント インストール ウィザードは、ローカル エージェント設定ファイル（LocalConfig.conf）を以下の場所にインストールします。

web_agent_home/config

注: 以下の例のとおり、*web_agent_home* 変数は、Web Agent のインストール場所を示します。

- Windows インストールのデフォルトの場所: C:¥Program Files¥CA¥webagent

  UNIX インストールのデフォルトの場所： /opt/ca/webagent

このファイルによって、Web エージェントがインストールされているのと同じ Web サーバ上に Web エージェント設定パラメータを設定することができます。そのため、関連するポリシー サーバ上のエージェント設定オブジェクトに格納されているパラメータを使用する必要はありません。

IIS Web エージェントの場合、Web エージェント設定ウィザードでは、ローカル エージェント設定ファイルの重複コピーを以下の場所に作成します。

web_agent_home¥bin¥IIS

詳細情報:

ローカル エージェント設定 (P. 31)
LocalConfig.conf ファイルの場所（フレームワーク エージェント） (P. 34)
ローカル設定の実装 (P. 37)

トレース設定ファイル

トレース設定ファイル(trace.conf)を使用して、以下の項目についてトレース ログを設定できます。

- 接続 API
- IPC プロバイダ
- TCP/IP (Transport)
- API の監視

Web エージェント インストール ウィザードは、トレース設定ファイルを以下の場所に作成します。

注: 以下の例のとおり、*web_agent_home* 変数は、Web Agent のインストール場所を示します。

- Windows インストールのデフォルトの場所: C:¥Program Files¥CA¥webagent

  UNIX インストールのデフォルトの場所 : /opt/ca/webagent

詳細情報:

IPC セマフォ関連メッセージ出力の Apache エラー ログへの制限 (P. 70)

Web エージェントトレース設定ファイル

Web エージェントトレース設定ファイルを使用して、Web エージェント操作のさまざまな要素についてトレース ログを作成することができます。たとえば、SiteMinder シングル サインオン(SSO)機能に関連するさまざまな Web エージェント動作についてトレース ログを作成できます。

Web エージェント インストール ウィザードは、Web エージェントトレース設定ファイルを以下の場所に作成します。

注: 以下の例のとおり、*web_agent_home* 変数は、Web Agent のインストール場所を示します。

- Windows インストールのデフォルトの場所: C:¥Program Files¥CA¥webagent
- UNIX インストールのデフォルトの場所 : /opt/ca/webagent

詳細情報:

トレース ロギングをセットアップする方法 (P. 190)

SiteMinder ホスト設定ファイル

Web エージェント設定ウィザードは、ホスト設定ファイル(SmHost.conf)を、SiteMinder Web エージェントが設定された Web サーバごとに、以下の場所に作成します。

web_agent_home/config

注: 以下の例のとおり、*web_agent_home* 変数は、Web Agent のインストール場所を示します。

- Windows インストールのデフォルトの場所: C:¥Program Files¥CA¥webagent
- UNIX インストールのデフォルトの場所 : /opt/ca/webagent

SmHost.conf ファイルには、Web エージェントが関連付けられているポリシー サーバに対する最初の接続で使用される情報が含まれます。

注: 詳細については、「SiteMinder Web エージェント インストール ガイド」を参照してください。

Web エージェント設定ファイル

SiteMinder Web エージェント設定ウィザードは、Web エージェント設定ファイル（WebAgent.conf）を、SiteMinder Web エージェントが設定された Web サーバごとに、以下の場所に作成します。

web_agent_home¥conf

注: 以下の例のとおり、*web_agent_home* 変数は、Web Agent のインストール場所を示します。

- Windows インストールのデフォルトの場所: C:¥Program Files¥CA¥webagent
- UNIX インストールのデフォルトの場所： /opt/ca/webagent

このファイルを使用して、Web エージェントを有効または無効（開始または停止）することができます。

IIS Web エージェントの場合、Web エージェント設定ウィザードでは、ローカルエージェント設定ファイルの重複コピーを以下の場所に作成します。

web_agent_home¥bin¥IIS

詳細情報:

Web エージェントの有効化 (P. 73)
Web エージェントの無効化 (P. 74)

第 3 章: ユース ケース

このセクションには、以下のトピックが含まれています。

ユース ケース展開の前提条件を満たす方法

すべての Web エージェント ユース ケースには以下の前提条件があります。Web エージェントを展開する前にそれらを完了する必要があります。

1. 管理 UI を使用して以下を実行します。
   - 保護の対象とするリソースを定義します。
   - 認証方式を実装します。
2. 各 Web エージェントの設定メソッドを選択します。以下の*いずれか*を選択してください。
   - 中央設定
   - ローカル設定
   - 中央設定とローカル設定の組み合わせ
3. 同じ環境内で従来のエージェントとフレームワーク エージェントの両方を使用する予定がある場合は、Web エージェントの各タイプに適用されるサーバ プラットフォームと Web エージェントのバージョンを確認する必要があります。

ユース ケース 1: 1 つのエージェントで 1 つのリソースを保護する

このユース ケースでは、1 つの Web エージェントで Web サイトやアプリケーションなどの 1 つのリソースを保護するように設定する方法について説明します。このユース ケースの環境は、以下のコンポーネントおよび設定で構成されています。

- 1 つの Web エージェント
- 1 つの Web サーバ

ユース ケース 1 を実装する方法

Web エージェントのユース ケース 1 を実装するには、以下の手順に従います。

1. 前提条件を満たしていることを確認します。
2. エージェント名と デフォルト エージェント名識別情報 (P. 54) を設定します。
3. 以下のロギング タイプのいずれかを実装します。
   - エラー ロギング (P. 186)
   - 追跡ロギング (P. 190)
   - OneView モニタ (P. 59)
4. Web エージェント (P. 73) を有効にします。

ユース ケース 2: 複数のエージェントで 1 つのドメイン内の複数のアプリケーションを保護する

このユース ケースでは、複数の Web エージェントで 1 つのドメイン内の複数のリソースを保護するように設定する方法について説明します。このユース ケースの環境は、以下のコンポーネントおよび設定で構成されています。

- 複数の Web エージェント
- 複数の Web サーバ
- 複数のリソース
- シングル サインオン
- cookie プロバイダ

ユース ケース 2 を実装する方法

Web エージェントのユース ケース 2 を実装するには、以下の手順に従います。

1. 前提条件を満たしていることを確認します。
2. (オプション) 任意の 仮想サーバ (P. 75)を設定します。
3. 各 Web エージェントのエージェント名とデフォルトのエージェント名 識別情報 (P. 54)を設定します。
4. シングル サインオン (P. 93)を有効にします。
5. cookie プロバイダ (P. 97)を設定します。
6. 以下のいずれかの機能を実装します。
   - エラー ロギング (P. 186)
   - 追跡ロギング (P. 190)
   - OneView モニタ (P. 59)
7. Web エージェント (P. 73)を有効にします。

ユース ケース 3: フレームワーク エージェントとトラディショナル エージェントで 1 つのドメイン内の複数のアプリケーションを保護する

このユース ケースでは、フレームワーク Web エージェントとトラディショナル Web エージェントを組み合わせて、1 つのドメイン内の複数のリソースを保護するように設定する方法について説明します。このユース ケースの環境は、以下のコンポーネントと設定およびドメインで構成されています。

- 複数のフレームワーク Web エージェントとトラディショナル Web エージェント
- 複数の Web サーバ
- 複数のリソース
- シングル サインオン
- cookie プロバイダ

ユース ケース 3 を実装する方法

Web エージェントのユース ケース 3 を実装するには、以下の手順に従います。

1. 前提条件を満たしていることを確認します。
2. (オプション) 任意の 仮想サーバ (P. 75)を設定します。
3. 各 Web エージェントのエージェント名とデフォルトのエージェント名 識別情報 (P. 54)を設定します。
4. 従来のエージェントとフレームワーク エージェントの間の適切な互換性を確保するには、以下の手順に従います。
   - 保護されていないリソースにおける cookie プロバイダの無視 (P. 112)
   - POST 要求における cookie プロバイダの無視 (P. 113)
   - 混在環境 (P. 269)での FCC と NTC の使用
5. シングル サインオン (P. 93)を有効にします。

   注: SSO 環境のすべてのエージェントの SecureURLs パラメータを同じ値に設定します。
6. cookie プロバイダ (P. 97)を設定します。
7. 以下のいずれかの機能を実装します。
   - エラー ロギング (P. 186)
   - 追跡ロギング (P. 190)
   - OneView モニタ (P. 59)
8. Web エージェント (P. 73)を有効にします。

ユース ケース 4: フレームワーク エージェントとトラディショナル エージェントで複数のドメインにおける複数のアプリケーションを保護する

このユース ケースでは、フレームワーク Web エージェントとトラディショナル Web エージェントを組み合わせて、複数のドメインにおける複数のリソースを保護するように設定する方法について説明します。このユース ケースの環境は、以下のコンポーネントおよび設定で構成されています。

- 複数のフレームワーク Web エージェントとトラディショナル Web エージェント
- 複数の Web サーバ
- 複数のリソース
- シングル サインオン
- cookie プロバイダを設定
- （省略可）SSO ゾーンを使用

ユース ケース 4 を実装する方法

Web エージェントのユース ケース 4 を実装するには、以下の手順に従います。

1. 前提条件を満たしていることを確認します。
2. (オプション) 任意の 仮想サーバ (P. 75)を設定します。
3. 各 Web エージェントのエージェント名とデフォルトのエージェント名 識別情報 (P. 54)を設定します。
4. 従来のエージェントとフレームワーク エージェントの間の適切な互換性を確保するには、以下の手順に従います。
   - 保護されていないリソースにおける cookie プロバイダの無視 (P. 112)
   - POST 要求における cookie プロバイダの無視 (P. 113)
   - 混在環境 (P. 269)での FCC と NTC の使用
   - フレームワーク エージェントと従来のエージェントの間の POST 維持の有効化 (P. 283)
5. シングル サインオン (P. 93)を有効にします。

   注: SSO 環境のすべてのエージェントの SecureURLs パラメータを同じ値に設定します。
6. cookie プロバイダ (P. 97)を設定します。
7. (オプション) SSO ゾーン (P. 328)を実装します。
8. 以下のいずれかの機能を実装します。
   - エラー ロギング (P. 186)
   - 追跡ロギング (P. 190)
   - OneView モニタ (P. 59)
9. Web エージェント (P. 73)を有効にします。

第 4 章: 基本 Web エージェント設定

このセクションには、以下のトピックが含まれています。

Web エージェント設定パラメータのデフォルト設定

別の値が指定されている場合を除き、Web エージェント設定パラメータのデフォルト設定が常に使用されます。

エージェント設定オブジェクトにもローカル設定ファイルにもパラメータがない場合は、デフォルト値が使用されます。

エージェント名とデフォルト エージェント名識別情報の設定

エージェント名はそのエージェントの識別情報です。ポリシー サーバは、この識別情報を使用してポリシーを Web エージェントに関連付けます。以下のパラメータを使用して Web エージェントの名前を定義できます。

AgentName

Web エージェントの ID を定義します。エージェント名は、その名前と、エージェントをホストしている各 Web サーバ インスタンスの IP アドレスの間で、マッピングを確立します。

このパラメータに対して値が設定されていない場合、または列挙された値の中の一致を Web エージェントが検索しない場合、Web エージェントは、DefaultAgentName パラメータ内で設定されている値を代わりに使用します。

注: このパラメータは複数の値を持つことができます。エージェント設定オブジェクト内でこのパラメータを設定する場合は、複数値オプションを使用します。ローカル設定ファイルでは、パラメータ名に続いて各値をファイルの個別の行に追加します。

デフォルト: デフォルトなし

制限: 32-127 の範囲内に 7 ビット ASCII 文字が含まれている必要があり、1 つ以上の印刷可能文字が含まれている必要があります。アンパサンド(&)およびアスタリスク(*)文字は含めることができません。大文字と小文字は区別されません。たとえば、MyAgent と myagent という名前は、同じように処理されます。

例: myagent1,192.168.0.0

例: myagent, www.sitea.com

DefaultAgentName

AgentName パラメータに指定されたエージェント名がない IP アドレスまたはインターフェースのリクエストを受け取った場合に Web エージェントが使用する名前を定義します。

仮想サーバを使用している場合、各仮想サーバに別々の Web エージェントを定義する代わりに、DefaultAgentName を使用することによって、SiteMinder 環境を迅速にセットアップできます。

重要: DefaultAgentName パラメータに値を指定しない場合は、AgentName パラメータにすべてのエージェント ID をリストする必要があります。そうしないと、ポリシー サーバは Web エージェントにポリシーを結び付けることができません。

デフォルト: デフォルトなし

制限: 32-127 の範囲内に 7 ビット ASCII 文字が含まれている必要があり、1 つ以上の印刷可能文字が含まれている必要があります。アンパサンド(&)およびアスタリスク(*)文字は含めることができません。大文字と小文字は区別されません。たとえば、MyAgent と myagent という名前は、同じように処理されます。

仮想サーバ サポートを設定する場合、AgentName または DefaultAgentName パラメータに対して、値を指定する必要があります。

エージェント名とデフォルト エージェント名識別情報を設定する方法

1. 以下のどちらかを実行することによって、エージェント名識別情報を指定します。
   - 集中的なエージェント設定の場合は、管理 UI 上でエージェント設定オブジェクトを開き、目的の値を AgentName パラメータに追加します。
   - ローカル エージェント設定の場合は Web サーバ上でローカル設定ファイルを開きます。目的の値をファイル内の個別の行に追加します。
2. 以下のどちらかを実行することによって、デフォルトのエージェント名識別情報を指定します。
   - 中央エージェント設定の場合は、管理 UI 上でエージェント設定オブジェクトを開き、目的の値を DefaultAgentName パラメータに追加します。
   - ローカル エージェント設定の場合は Web サーバ上でローカル設定ファイルを開きます。目的の値を DefaultAgentName パラメータに追加します。

   エージェント名とデフォルト エージェント名識別情報が設定されます。

エージェント名の一致の確認

SiteMinder のルールおよびポリシーは、エージェント名に結び付けられています。あるホストに対して要求を送信し、そのホストが持つエージェント名がポリシー サーバ上で不明の場合、ポリシー サーバはポリシーを実装できません。したがって、Web エージェントの DefaultAgentName パラメータまたは AgentName パラメータの値は、ポリシー サーバ上で定義されたエージェント エントリの名前と一致している必要があります。

エージェントは、管理 UI を使用してポリシー サーバで定義します。[エージェント プロパティ]ダイアログ ボックスの[名前]フィールドに入力する値は、DefaultAgentName または AgentName 設定項目で定義された値と一致している必要があります。それぞれは、Web エージェントがローカルで(エージェント設定ファイル)、またはポリシー サーバから集中的に(エージェント設定オブジェクト)設定されていることを意味します。

エージェント名の暗号化

URL により、ユーザがフォーム、SSL、または NTLM 認証情報コレクタへリダイレクトされる場合、Web エージェントはデフォルトで、その URL に自らの名前を追加します。EncryptAgentName パラメータを使用して、エージェントが URL 内のその名前を暗号化するかどうかや、認証情報コレクタが URL を受信する場合に名前を復号化するかどうかを制御できます。

EncryptAgentName パラメータのデフォルト設定は yes です。以下のいずれかの状況では、このパラメータを no に設定してください。

- 認証情報コレクタと共にサードパーティのアプリケーションを使用していて、そのアプリケーションが処理を進めるためにエージェント名を読み取れるようにする必要がある場合
- フォーム認証を実行し、認証の対象となる 1 つのリソースへユーザを振り向けるために、Web エージェントをフォーム認証情報コレクタ(FCC)として構成する場合 シングル リソース ターゲットを設定する手順では、暗号化されていないエージェント名が必要です。

Web エージェント名を暗号化するには、EncryptAgentName パラメータを yes に設定します。

Web エージェントとポリシー サーバ間の通信を管理する方法

以下の手順のいずれかを使用して、Web エージェントとポリシー サーバの間の通信を管理できます。

- CA Wily Introscope を使用して、Web エージェントを監視します。
- 監査の目的で、SiteMinder OneView モニタを使用します。
- エージェントがポリシーの変更結果を受け取る頻度を指定します。
- ネットワーク遅延問題に対応します。
- 複数の Web サーバ インスタンスを持つ Web エージェントを管理します。

CA Wily Introscope を使用した Web エージェントの監視

すでに CA Wily Introscope を使用している場合は、以下のパラメータを使用して SiteMinder Web エージェントの稼働状況を監視できます。

EnableIntroscopeApiSupport

SiteMinder Web エージェントに関する情報を収集し、プラグインを使用して CA Wily Introscope に送ります。このパラメータは以下の設定を使用します。

- yes に設定されたとき、Wily プラグインは、データを収集するために API をコールします。
- no に設定されたとき、Wily プラグインはデータを備えた HTTP ヘッダを作成します。
- both に設定されたとき、Wily プラグインは API をコールし、かつデータを備えた HTTP ヘッダを作成します。
- none に設定されたとき、データは収集されません。

デフォルト: no

制限: yes、both、no、none

例:（HTTP ヘッダ）sm-wa-perf-counters = *server_name*.example.com:6180,86117203,86118343,1,0,0,1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,1125,0,15,1,1,750,750,

CA Wily Introscope を使用して Web エージェントの稼働状況を監視するには、EnableIntroscopeApiSupport パラメータの値を以下のいずれかに設定します。

- Yes
- Both
- No

OneView モニタによる Web エージェントの監視

SiteMinder OneView モニタは、キャッシュ統計情報と他の情報をポリシー サーバへ送信します。管理者はポリシー サーバを使用して、Web エージェントを分析し、微調整することができます。以下のパラメータを使用して SiteMinder OneView モニタを制御します。

EnableMonitoring

SiteMinder Web エージェントが監視情報をポリシー サーバに送信するかどうかを指定します。

デフォルト: No

Web エージェントで SiteMinder OneView モニタが使用されるようにするには、EnableMonitoring パラメータを yes に設定します。

注: 詳細については、ポリシー サーバ ドキュメントを参照してください。

エージェントがポリシーまたはキーの更新をチェックする頻度の変更

Web エージェントは、以下のアイテムをチェックするために定期的にポリシー サーバをポーリングします。

- 更新された管理情報
- 更新されたポリシー
- 動的に更新されたエージェント キー

この間隔は必要に応じて以下のパラメータを使用して変更できます。

PSPollInterval

ポリシー変更に関する情報または動的に更新されたキーを取得するために Web エージェントがポリシー サーバと通信する間隔（秒単位）を指定します。数値が大きい（間隔が長い）ほど、ネットワークトラフィックは減少します。数値が小さい（間隔が短い）ほど、ネットワークトラフィックは増加します。

デフォルト: 30

制限: 1

Web エージェントが更新がないかポリシー サーバをチェックする頻度を変更するには、PSPollInterval パラメータの秒数を変更します。

重要: PSPollInterval パラメータを増加させると、Web エージェントで SiteMinder ポリシー変更が提供されるタイミングにも影響があります。たとえば、勤務が終了した従業員のアクセスを無効にするため、10:30 にポリシーを変更し、PSPollInterval パラメータの値が 3600（1 時間の秒数）であるとします。この場合、Web エージェントでは、変更されたポリシーを 11:30 まで適用しません。

詳細情報:

Web エージェントとダイナミック キーのロールオーバー (P. 23)

ネットワーク遅延への対応

ネットワーク遅延の問題が存在する場合、Web エージェントはポリシー サーバに接続できません。この問題を回避するには、エージェント設定オブジェクトまたはローカル設定ファイル内で以下のパラメータを使用します。

AgentWaitTime

Low Level Agent Worker Process（LLAWP）が使用可能になるまで、Web エージェントが何秒待つかを指定します。指定した時間を過ぎると、Web エージェントはポリシー サーバに接続しようとします。

このパラメータの設定は、LLAWP 接続に関連するエージェント スタートアップ エラーを解決するのに役立つ場合があります。デフォルト値で設定を開始し、エージェントが正常に開始されるまで、5 秒ずつ間隔を増やすことをお勧めします。
エージェント設定オブジェクトまたは LocalConfig.conf ファイルにこのパラメータを設定することが望ましくない場合は、代わりに WebAgent.conf ファイルに設定することもできます。

デフォルト: 5

例: プライマリおよびセカンダリのポリシー サーバを使用している場合は、30 から 40 の値で開始してみてください。

制限: なし

注: ネットワーク遅延の問題が発生している場合、このパラメータをフレームワーク エージェントで使用できます。設定値を大きくするのは、ネットワーク遅延の問題が存在する場合のみにしてください。設定値を大きくすると、予期しない Web サーバの動作が発生する可能性があります。

ネットワーク遅延に対応するには、エージェント設定オブジェクトまたはローカル設定ファイルに AgentWaitTime パラメータを追加し、希望の秒数を指定します。

複数の Web サーバ インスタンスを持つ Web エージェントの管理

複数の Web サーバ インスタンス上に Web エージェントを設定する場合、各サーバ インスタンスは、独自の Web エージェント キャッシュ、ログ ファイル、および正常性監視リソースを持つ必要があります。リソースを一意にするために、以下のパラメータを設定します。

ServerPath

Web エージェントが Web サーバの複数インスタンスを使用するように設定されている場合に、各 Web サーバ インスタンスの一意のパスを指定します。ServerPath は、Web エージェントのキャッシュ、ロギング、および状態監視のリソースに関して、一意の識別情報を作成します。

デフォルト: 空白

例: 4 つの Web サーバ インスタンスがあり、それぞれが Web エージェントをロードする場合、各サーバの WebAgent.conf ファイルの ServerPath パラメータには一意の値を設定する必要があります。ServerPath パラメータは *server_instance_root*/logs など、Web サーバのログ ファイルが保管されるディレクトリに設定できます。

注: IIS 6.0 サーバの LocalConfig.conf ファイルおよび Apache 2.0 サーバの LocalConfig.conf ファイル（Apache サーバのインスタンスが 1 つのみの場合）には、この設定を追加しないでください。

複数のサーバ インスタンス上に Web エージェントを設定するには、ServerPath パラメータに一意のパスを追加します。

Windows システムに関する ServerPath パラメータの設定

複数のサーバ インスタンスがある場合に、これらのインスタンスが同じエージェント リソース(キャッシュ、ログ、正常性監視)を共有しないようにするには、Windows 上の Apache Web サーバに、ServerPath パラメータが必要です。Web サーバ インスタンスが独自のリソースを持つようにするには、ServerPath の値を指定します。

ServerPath パラメータに設定する値は、システム上で実行されているサーバ インスタンス間で一意の英数字文字列である必要があります。たとえば、2 つのサーバ インスタンスがある場合、一方のインスタンスの ServerPath パラメータの値を MyAgent1 に設定し、もう一方のインスタンスの値を MyAgent2 に設定できます。

注: ServerPath パラメータに指定する文字列に円記号(¥)を使用しないでください。その他の文字はすべて使用できます。

ServerPath パラメータは以下の Windows プラットフォームには必要*ありません*。

- IIS 6.0 サーバ(常にサーバ インスタンスが 1 つのみあります)
- Apache 2.0 サーバ (Web サーバ インスタンスが 1 つのみある場合)。このパラメータはこれらのシステム上でサポートされていますが、使用されてはいません。
- Oracle iPlanet または Domino Web サーバ

  これらのサーバは ServerPath パラメータを使用しません。これは、このパラメータが Windows のマルチプロセス モードで動作しないためです。

UNIX システムに関する ServerPath パラメータの設定

ServerPath パラメータは WebAgent.conf ファイル内にあります。フレームワーク エージェントの LocalConfig.conf ファイルには、この設定を追加*しない*でください。

UNIX プラットフォーム上の Web サーバでは、各サーバ インスタンスが独自のエージェント リソースを持つようにすることをお勧めします。

UNIX 上の以下のサーバについて、ServerPath パラメータを設定します。

- Apache 2.0 (IBM HTTP Server など、すべての Apache 2.0 ベースのサーバを含む)
- Oracle iPlanet Web サーバ インスタンス

注: ServerPath は、UNIX システム上の Domino Web サーバには必要ありません。

ServerPath パラメータに設定する値は、システム上で実行されているサーバ インスタンス間で一意の英数字文字列である必要があります。たとえば、2 つのサーバ インスタンスがある場合、一方のインスタンスの ServerPath パラメータの値を MyAgent1 に設定し、もう一方のインスタンスの値を MyAgent2 に設定できます。

ServerPath パラメータを必要とする追加設定

ServerPath の設定が必要になる状況は、以下のとおりです。

- Web エージェントは、1 つのセマフォを使用して共有メモリを制御しています。セマフォとは、オペレーティング システム(またはカーネル)のストレージにある 1 つの値であり、各プロセスはその値を参照してリソースが利用可能かどうかを確認し、その値を変更することができます。セマフォは、一意に生成されるものではないため、複数のエージェントが同じメモリ領域を指そうとすると、エージェントは正常に機能しなくなります。それに対して、サーバ パスに命名した場合、インスタンスのルートが決定され、Web エージェントはセマフォに関連して固有のキーを作成するためのファイルを見つけることができます。
- サーバ インスタンス(Windows を除くすべてのプラットフォーム)が複数あると、エージェントは以下のいずれかの実行に失敗します。
  - エージェント名の暗号化(00-0012 エラー)
  - SMSESSION cookie または SMIDENTITY cookie の暗号化
  - 起動時のエージェント キー更新の取得
- Apache では(Windows を除くすべてのプラットフォーム)、Apache が再起動したときに、エージェントは 6 つの共有メモリ セグメント(セマフォ)を解放しません。
- 各 Web エージェントが同じシステム上の異なる Web サーバ タイプ(Apache 2.0 サーバおよび IIS 6.0 サーバなど)について設定されている場合、各サーバの設定について一意の ServerPath 値を指定する必要があります。異なる Web サーバ タイプ同士では、エージェント リソースを共有できません。

第 5 章: サーバ固有の Web エージェントの設定

このセクションには、以下のトピックが含まれています。

IIS Web エージェントの特別な設定

環境に以下の状況または条件のいずれかがある場合は、IIS Web エージェントに対して追加の設定変更を行う必要がある場合があります。

- リソースの要求が、404 not found エラーと共に戻された
- P3P コンパクト ポリシーを使用する
- IIS 6.0 セキュリティ コンテキストを使用する
- POST データのサイズを制限する必要がある

404 Not Found エラーの管理(IIS 6.0 エージェント)

IIS 6.0 Web サーバ上の Web エージェントが、リソース リクエストに対するレスポンスとして 404 Not found エラーを返す場合、または Web エージェントが IIS 6.0 Web サーバによって正しく呼び出されない場合は、IIS コンソールを開いて、ワイルドカード アプリケーション マッピングが有効であり、Web エージェント DLL を参照するように設定されていることを確認してください。また、マッピングを有効にする前に、[ファイルの存在を確認する]チェック ボックスがオフになっていることを確認してください。

Web エージェントの設定による P3P コンパクト ポリシーへの対応

以下のパラメータを使用して、Web エージェントからのカスタム レスポンスが P3P レスポンス ヘッダに準拠するかどうかを決定できます。

P3PCompactPolicy

カスタム レスポンスがプライバシ優先プロジェクト用のプラットフォーム（P3P）レスポンス ヘッダに準拠するかどうかを決定します。P3P コンパクトポリシーは、P3P の用語に基づく特定の要素を表すトークンを使用します。P3PCompactPolicy パラメータを適切なポリシー構文に設定した場合、Web エージェントに関して P3P レスポンス ヘッダが指定されている状況で、正しい P3P レスポンス ヘッダを使用して、カスタム レスポンスが設定されることを保証できます。

デフォルト: デフォルトなし

例: NON DSP COR CURa TAI（これらはそれぞれ、none、disputes、correct、current/always、および tailoring を表します）

P3P コンパクト ポリシーに対応するには、P3PCompactPolicy パラメータに適切なポリシー構文を追加します。

エージェントで機能する IIS 6.0 セキュリティ コンテキストの有効化

IIS 6.0 Web サーバ上の SiteMinder Web エージェントは ISAPI 拡張として機能します。HTTP リクエストが行われると、IIS 6.0 Web サーバがユーザに認証を要求した後、Web エージェントがリクエストに応答します。IIS サーバはネイティブの認証方式である基本認証を、認証要求に使用します(その方式がサーバの管理コンソールで選択されている場合)。

エージェントで機能する IIS 6.0 セキュリティ コンテキストを有効にする方法

1. IIS 管理コンソールを開き、ローカル コンピュータおよび[Web サイト]フォルダを展開します。
2. Web サイトのフォルダを右クリックし、[プロパティ]を選択します。

   (Web サイトの)プロパティ ダイアログ ボックスが表示されます。
3. [ディレクトリ セキュリティ]タブをクリックします。[認証とアクセス制御]セクションで、[編集]をクリックします。

   [認証方法]ダイアログが表示されます。
4. [匿名アクセスを有効にする]チェック ボックスをオンにし、[OK]をクリックします。

   [認証方法]ダイアログが閉じます。
5. [OK]をクリックします。

   プロパティ ダイアログ ボックスが閉じます。SiteMinder Web エージェントで機能する IIS 6.0 セキュリティ コンテキストが有効になります。

URLScan ユーティリティを使用する場合のサーバ HTTP ヘッダの削除

Microsoft の URLScan ユーティリティを使用して、IIS Web サーバが送信するレスポンスからサーバ HTTP ヘッダを削除する場合は、IIS Web エージェントの以下のパラメータも設定する必要があります。

SuppressServerHeader

IIS Web エージェントがレスポンスでサーバ HTTP ヘッダを返すことを防ぎます。このパラメータの値が no の場合、Web エージェントはレスポンスと一緒にサーバ ヘッダを送信し、IIS Web サーバはそれをクライアントに渡します。このパラメータの値が yes の場合、Web エージェントは、レスポンスでサーバ ヘッダを送信しません。

デフォルト: No

URLScan utiltiy が IIS サーバのレスポンスからヘッダを削除するのに対し、hte SuppressServerHeader パラメータは Web エージェントのレスポンスからヘッダを削除します。すべてのレスポンスでサーバ ヘッダがクライアントに送信されないようにするには、ユーティリティとパラメータの両方を設定する必要があります。

Web エージェントがレスポンスでサーバ ヘッダを送信しないようにするには、SuppressServerHeader パラメータの値を yes に設定します。

Apache Web エージェントの特殊な設定

ご使用の環境で以下の状況または条件のいずれかが当てはまる場合は、Apache Web エージェントに追加の設定変更を加える必要があります。

- HTTP ヘッダを変更 *せずに*トラフィックを特定の Web サーバにリダイレクトするアプリケーション。
- transfer-encoding をサポートして *いない* レガシー アプリケーション（HTTP 1.1 以前）。
- Web エージェントが Apache エラー ログに記録する情報メッセージが多すぎる

ポート番号に関する HTTP HOST 要求の使用

実際の HTTP ヘッダを*変更せずに*、特定の Web サーバへのトラフィックをリダイレクトすることにより、負荷分散を実行するアプリケーションがある場合は、以下のパラメータを使用して、(ロード バランサによって使用されるポートの代わりに)適切な外部ポートにユーザをリダイレクトするように Web エージェントを設定する必要があります。

GetPortFromHeaders

Web サーバ サービス構造からポート番号を取得する代わりに、HTTP HOST リクエスト ヘッダからポート番号を取得するように Web エージェントに指示します。

デフォルト: No

注: このパラメータは、Apache Web エージェントにとって必須です。

HTTP HOST 要求ヘッダ内でポート番号を使用するには、GetPortFromHeaders パラメータを yes に設定します。

Apache Web エージェントでのレガシー アプリケーションの使用

(HTTP 1.1 をサポートしない)レガシー アプリケーションがあり、それらを Apache Web サーバで実行する場合は、以下のパラメータを設定します。

LegacyTransferEncodingBehavior

Web エージェントが使用するメッセージ エンコーディングのタイプを指定します。このパラメータの値が no の場合、転送エンコーディング(transfer-encoding)がサポートされます。

このパラメータの値が yes の場合、コンテンツ エンコーディングがサポートされます。transfer-encoding ヘッダは無視され、content-length ヘッダのみがサポートされます。

デフォルト: No

Apache Web サーバでレガシー アプリケーションを使用するには、LegacyTransferEncodingBehavior パラメータの値を yes に設定します。

重要: このパラメータの値を yes に設定すると、Federation や、4 KB より長い POST データの維持といった機能が動作せず、大きな証明書が認識されない場合があります。

IPC セマフォ関連メッセージ出力の Apache エラー ログへの制限

デフォルトでは、Apache Web エージェントは、設定された Apache のロギング レベルにかかわらず、Apache のエラー ログへのすべてのレベル（情報およびエラー）の IPC セマフォ関連メッセージを記録します。

Web エージェントの IPC セマフォ関連出力の詳細を Apache のエラー ログに制限するには、*web_agent_home*/config 内にある trace.conf ファイルに以下のパラメータを追加します。

nete.stderr.loglevel

Web エージェントが Apache のエラー ログに記録する IPC セマフォ関連メッセージのレベルを指定します。以下の値を受け入れます。

off

Web エージェントは、IPC セマフォ関連メッセージを Apache のエラー ログに記録しません。

error

Web エージェントは、IPC セマフォ関連のエラー メッセージのみを Apache のエラー ログに記録します。

info

（デフォルト）Web エージェントは、IPC セマフォ関連のエラーおよび情報メッセージを Apache のエラー ログに記録します。

例：trace.conf 内の nete.stderr.loglevel パラメータの定義

trace.conf の以下の抜粋では、IPC セマフォ関連のエラー メッセージのみが Apache のエラー ログに記録されるように Web エージェントを制限するように nete.stderr.loglevel パラメータが設定されています。

```
# CA Web Agent IPC logging levels
# nete.stderr.loglevel=error
```

Oracle iPlanet Web サーバ上でのディレクトリ参照の制限

Oracle iPlanet Web サーバのディレクトリを参照しようとするユーザが SiteMinder によって認証要求されるようにするために、以下のパラメータを設定できます。

DisableDirectoryList

最初に認証情報を要求せずに、ユーザがディレクトリの内容を表示または参照することを Web エージェントが認めるかどうかを指定します。これは、以下の条件が *すべて* 当てはまる場合に発生します。

- レルムがルートリソース(/)を保護するように設定されている。
- ディレクトリのデフォルト Web ページ(index.html など)が名前変更または削除されている。

デフォルト: No

Oracle iPlanet サーバ上のディレクトリ参照を制限する方法

1. エージェント設定オブジェクトまたはローカル設定ファイルに DisableDirectoryList パラメータを追加します。
2. DisableDirectoryList パラメータの値を yes に設定します。

   ディレクトリ参照が制限されます。SiteMinder がディレクトリを参照しようとするユーザの認証を要求します。

詳細情報:

Web エージェントがクライアント要求を 2 度処理する (P. 354)

第 6 章: Web エージェントの起動と停止

このセクションには、以下のトピックが含まれています。

Web エージェントの有効化

Web エージェントのパラメータを設定して、Web エージェントが Web サーバ上のリソースを保護できるようにします。

注: SiteMinder ポリシー サーバにポリシーも定義するまでは、リソースは保護*されません*。

Web エージェントを有効にする方法

1. WebAgent.conf ファイルを開きます。
2. EnableWebAgent パラメータの値を yes に変更します。
3. WebAgent.conf ファイルを保存して閉じます。
4. エージェント設定オブジェクトまたはローカル設定ファイル内の以下のパラメータのいずれかの設定を変更した場合は、Web サーバを再起動します。
   - AgentConfigObject
   - CacheAnonymous
   - HostConfFile(IIS 6.0 エージェントおよび Apache エージェントのみ)
   - MaxResourceCacheSize
   - MaxSessionCacheSize
   - PostPreservationFile
   - ResourceCacheTimeout

   Web エージェントが有効になります。

Web エージェントの無効化

Web エージェントによる Web サーバ上のリソースの保護およびポリシー サーバとの通信を停止する場合は、Web エージェントを無効にする必要があります。

Web エージェントを無効にする方法

1. WebAgent.conf ファイルを開きます。
2. EnableWebAgent パラメータの値を no に変更します。
3. WebAgent.conf ファイルを保存して閉じます。
4. エージェント設定オブジェクトまたはローカル設定ファイル内の以下のパラメータのいずれかの設定を変更した場合は、Web サーバを再起動します。
   - AgentConfigObject
   - CacheAnonymous
   - HostConfFile（IIS 6.0 エージェントおよび Apache エージェントのみ）
   - MaxResourceCacheSize
   - MaxSessionCacheSize
   - PostPreservationFile
   - ResourceCacheTimeout

   Web エージェントが無効になります。

第 7 章: 仮想サーバの設定

このセクションには、以下のトピックが含まれています。

仮想サーバ サポートをセット アップする方法

仮想サーバは、物理サーバに設定する論理エンティティです。論理エンティティは 1 つの独立したサーバとして機能します。仮想サーバを設定すれば、1 つの物理サーバ上で複数の Web サイトをホスティングできます。たとえば、仮想サーバを使って特定のサーバ上に www.mysite.com と www.yoursite.com の両方のサイトをホスティングするようにセットアップすることができます。

仮想サーバには以下の各項目を割り当てることができます。

- 一意の IP アドレス
- 物理サーバと共有する IP アドレス
- 別の仮想サーバと共有する IP アドレス

1 つの Web サーバ インスタンスに設定できる Web エージェントは 1 つのみですが、エージェント ID を設定してすべての仮想サーバを保護できます。あるユーザが www.mysite.com からサーバにアクセスし、別のユーザが www.yoursite.com からサーバにアクセスする場合、それぞれのサーバは個々のエージェント ID によって保護されます。仮想サーバごとにエージェント ID を作成すると、サイトごとに固有のレルムとルールを定義できるという利点があります。

Web エージェントに対して定義した設定は、その Web サーバ インスタンスに対して定義したすべての仮想サーバに適用されますが、要求の処理は仮想サーバが互いに独立して行い、ポリシー サーバでは、それぞの仮想サーバ要求を別々に扱います。仮想サーバおよびその設定方法の詳細については、使用する Web サーバのマニュアルを参照してください。

仮想サーバのサポートを設定するには、以下の*いずれか*のタスクを実行します。

- 各仮想サーバのエージェント ID を定義および追加し、AgentName パラメータの値を指定して、仮想サーバの IP アドレスまたはホスト ヘッダ名に割り当てます。
- 固有のものとして識別されることを必要とする仮想サーバのみに対して、エージェント ID を定義します。
- デフォルトのエージェント名を設定します。

注: Oracle iPlanet Web サーバの複数のインスタンスを使用している場合（HTTP 通信用のサーバと HTTPS 通信用のサーバなど）、2 つの WebAgent.conf ファイルが存在します。各ファイルが複数のエージェント ID を保持することができます。（Oracle iPlanet という名前は、以前は Sun ONE および iPlanet と呼ばれていた Web サーバです）。

詳細情報:

エージェント名とデフォルト エージェント名識別情報の設定 (P. 54)

IIS 6.0 仮想 Web サイトを保護するための SiteMinder ワイルドカード マッピングの追加

SiteMinder では、IIS Web サーバのデフォルトの Web サイト フォルダのみを自動的に保護します。IIS 6.0 Web サーバ上で仮想 Web サイトを実行している場合は、保護する対象の仮想 Web サイトごとにワイルドカード mappings を追加します。ワイルドカード マッピングを仮想サイトに追加したら、Web エージェントを有効にします。

IIS 6.0 仮想 Web サイトを保護するために SiteMinder ワイルドカード マッピングを追加する方法

1. IIS 6.0 Web サーバに対して仮想サーバを設定します。

   注: 詳細については、IIS のマニュアルを参照してください。

2. IIS 管理コンソールを開きます。
3. [仮想 Web サイト]を右クリックし、[プロパティ]を選択します。

   [プロパティ]ダイアログボックスが表示されます。

4. [ホーム ディレクトリ]タブをクリックします。
5. [設定]をクリックします。

   [アプリケーションの構成]ダイアログ ボックスが表示されます。

6. [ワイルド カード アプリケーション マップ]セクションで、[挿入]をクリックします。
7. [参照]をクリックし、以下のファイルに移動します。

   web_agent_home¥SiteMinder Web Agent¥Bin¥ISAPI6WebAgent.dll

   注: 以下の例のとおり、*web_agent_home* 変数は、Web Agent のインストール場所を示します。

   - Windows インストールのデフォルトの場所: C:¥Program Files¥CA¥webagent

     UNIX インストールのデフォルトの場所： /opt/ca/webagent

8. [OK]を 2 回クリックします。

   [アプリケーション構成]および[プロパティ]ダイアログ ボックスが閉じます。

9. 仮想 Web サイトを再起動します。
10. 保護する仮想 Web サイトごとに、手順 3 ～ 9 を繰り返します。

    IIS Web エージェントを有効にする準備ができました。

仮想サーバの Web エージェント ID の割り当て

各仮想サーバに対して、追加の Web エージェントが実際に*「定義」*されているわけではありません。代わりに、Web エージェント ID の*「割り当て」*が行われています。独特のアクセス要件が存在する仮想サーバを保護する場合、または個別のレルムを保護する場合は、各サーバに固有のエージェント ID を割り当て、他のすべての仮想サーバに対しては、デフォルトのエージェント名を使用します。このオプションには、SiteMinder のインストールを短時間で設定できるだけでなく、別個に保護する必要があるレルムをホストする仮想サーバをこれまでどおり保護できるという利点があります。

AgentName パラメータとそれに関連付けられた IP アドレスによって、Web サーバ インターフェースと、ポリシー ストア内で定義済みのエージェント名の間のマッピングが実現されます。Web エージェントは、適用対象となるルールとポリシーの正しいセットを取得するために、適切なエージェント名のコンテキスト内でエージェント API 呼び出しを順に実行する必要があります。ポリシー ストアへのマッピングでエージェント名または IP アドレスが割り当てられない場合、Web エージェントは、仮想サーバにのみ DefaultAgentName パラメータの値を使用します。

固有のエージェント ID を使って仮想サーバを保護するには、AgentName パラメータを使用して、仮想サーバごとに 1 つの Web エージェントを追加します。仮想サーバごとに異なる Web エージェントを追加することにより、各仮想サーバに固有のレルムとルールを定義することができます。

Web エージェント ID を割り当てるには、以下の手順に従います。

1. エージェント名と IP アドレスを、カンマで区切って入力します。
2. 仮想サーバが同じ IP アドレスを共有し、異なるポートを使用する場合は、IP アドレスに関連付けられたポート番号（たとえば、112.12.12.1:8080）を指定します。デフォルトのポートを使用している場合、ポート番号は必要ありません。
3. 複数のエージェントを追加するには、以下の例のように、個々の行にそれぞれ入力します。

```
agentname="agent1,123.123.12.12:8080"
agentname="agent2,123.123.12.12:8081"
agentname="agent3,123.123.12.13"
```

4. エージェント ID を追加する場合は、管理 UI で同じ設定を使用してエージェント ID を定義する必要があります。エージェント ID が、エージェント設定の定義と完全に一致するように、管理 UI で定義されていることを確認してください。

AgentName パラメータに入力がない場合、SiteMinder は、仮想サーバにのみ DefaultAgentName の値を使用します。

注: DefaultAgentName を変更する場合は、エージェントに対する定義と完全に一致するように、管理 UI で定義されていることを確認してください。

Web エージェントで無視する仮想サーバの指定

使用中のサイト内にある 1 台の Web サーバが複数の仮想サーバをサポートしている場合、それらの仮想サーバ上には、Web エージェントで保護したくないリソースが存在している可能性があります。Web サーバ コンテンツのうち、どの部分を保護する必要があるのか Web エージェントが簡単に識別できるように、以下のパラメータを使用します。

IgnoreHost

Web エージェントで無視するあらゆる仮想サーバの完全修飾ドメイン名を指定します。 そのような仮想サーバ上にあるリソースは自動許可され、どのクライアントが要求を行ったかにかかわらず、Web エージェントは常にそれらのリソースへのアクセスを許可します。許可は、ポリシーではなく Web エージェントの設定に基づいて決定されます。

IgnoreExt や IgnoreURL の各設定など、自動許可に関する他の項目より先に、無視されるホストに関する上記のリストが最初にチェックされます。したがって、無視されるホスト上にあるリソースに関しては、ダブルドットルールがポリシー サーバに対する許可の呼び出しをトリガすることはありません。しかし、拡張子に関するルールがそのようなリソースを無視することはありません。

IgnoreHost パラメータに対する URL エントリのホスト部分は、Web エージェントが読み取る、要求されたリソースのホスト ヘッダと完全に一致する必要があります。

注: この値では、大文字と小文字が区別されます。

URL で特定のポートを使用する場合、ポートを指定する必要があります。

一元管理されたエージェントでは、いくつかのサーバを表わすために
エージェント設定オブジェクトで複数値パラメータを使用します。ローカ
ル設定ファイルで設定されたエージェントでは、ファイル内の個別の行
に各ホストを列挙します。

例:（指定したポートと共に表示される URL）

IgnoreHost="myserver.example.org:8080"

例:（ローカル設定ファイル）

IgnoreHost="my.host.com"

IgnoreHost="your.host.com"

デフォルト: デフォルトなし

Web エージェントで無視する仮想サーバを指定するには、次のいずれかのタス
クを行います。

- 中央設定では、無視するサーバをエージェント設定オブジェクトに追加
  します。サーバが複数ある場合は、パラメータに複数値の設定を使用し
  ます。
- ローカル設定では、ローカル設定ファイルでサーバごとに個別の行を追
  加します。

指定された URL を使用するリソースは、Web エージェントによって無視され、
それらのリソースへのアクセスが自動的に付与されます。

IP アドレスによるエージェント ID の解決

仮想 Web サーバ上で、IP アドレスとホスト名を使用してエージェント名を解決する場合、Web エージェントは誤った AgentName の値を使用してリクエストを評価することがあります。これにより、認証されていないユーザが保護されたリソースにアクセスする状況が生じることがあります。

以下のパラメータを使用して、Web エージェントが仮想サーバの物理 IP アドレスに基づいてエージェント名を解決するように設定することができます。

UseServerRequestIp

仮想 Web サーバの物理 IP アドレスに従って AgentName を解決するように Web エージェントに指示します。Web サーバが仮想サーバ マッピングに IP アドレスを使用する場合は、このパラメータを使用してセキュリティを向上させます。このパラメータが no の場合、Web エージェントは、クライアントのリクエストの HTTP ホスト ヘッダ内のホスト名に従って AgentName を解決します。

Domino サーバでは、このパラメータは、Domino 6.x でのみサポートされています。他の Domino バージョン上のエージェントに対してこのパラメータを有効にすると、Web エージェントはデフォルトのエージェント名を使用します。

SSL 通信と仮想ホストを使用するように IIS Web エージェントが設定されている場合は、このパラメータを yes に設定する必要があります。IIS では、SSL を有効にした状態で、ホスト名を使用して仮想ホスト マッピングを行うことはできません。

デフォルト: No

IP アドレスを使用して Web エージェントの ID を解決するには、UseServerRequestIp パラメータを yes に設定します。

第 8 章: SiteMinder Web エージェントでのプライバシ優先プロジェクト用のプラットフォーム(P3P)のコンパクト ポリシーの使用

このセクションには、以下のトピックが含まれています。

Web エージェントの設定による P3P コンパクト ポリシーへの対応

以下のパラメータを使用して、Web エージェントからのカスタム レスポンスが P3P レスポンス ヘッダに準拠するかどうかを決定できます。

P3PCompactPolicy

カスタム レスポンスがプライバシ優先プロジェクト用のプラットフォーム(P3P)レスポンス ヘッダに準拠するかどうかを決定します。P3P コンパクトポリシーは、P3P の用語に基づく特定の要素を表すトークンを使用します。P3PCompactPolicy パラメータを適切なポリシー構文に設定した場合、Web エージェントに関して P3P レスポンス ヘッダが指定されている状況で、正しい P3P レスポンス ヘッダを使用して、カスタム レスポンスが設定されることを保証できます。

デフォルト: デフォルトなし

例: NON DSP COR CURa TAI (これらはそれぞれ、none、disputes、correct、current/always、および tailoring を表します)

P3P コンパクト ポリシーに対応するには、P3PCompactPolicy パラメータに適切なポリシー構文を追加します。

第 9 章: シングル サインオン（SSO）

このセクションには、以下のトピックが含まれています。

単一ドメインでのシングル サインオンの仕組み

SiteMinder には、単一および複数の cookie ドメインで使用するシングル サインオン機能が用意されています。ユーザは、シングル サインオン環境内であれば移動の際に再認証する必要がないため、この機能は、異なる Web サーバおよびプラットフォーム間でのアプリケーションの使用を簡略化し、また、ユーザ エクスペリエンスを向上させます。

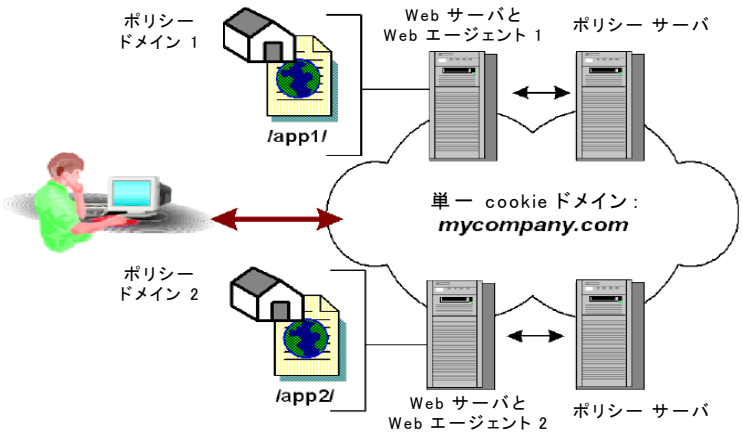
単一ドメイン環境では、すべてのリソースが 1 つの cookie ドメインに存在します。各 Web エージェントの設定で同じ cookie ドメインを指定すれば、同じ cookie ドメイン内の複数の Web エージェントに対してシングル サインオンを設定できます。

シングル サインオンが有効な場合、以下の手順が使用されます。

1. ユーザが一度認証します。
2. Web エージェントは成功した認証をキャッシュし、ユーザのブラウザ宛てにシングル サインオン cookie を発行します。
3. シングル サインオン cookie はセッション情報を提供します。その結果、ユーザは再認証なしで以下のタイプのリソースにアクセスできます。
   - 他のレルムにある保護されたリソース(ただし、保護レベルが*同等*または*それ以下*であるもの)
   - この cookie ドメイン内の別の Web サーバ

   保護レベルが*さらに高い*リソースにアクセスしようとするユーザは、アクセスが付与される前に再認証を受ける必要があります。

以下の図は、単一 cookie ドメインにおけるシングル サインオンを示しています。

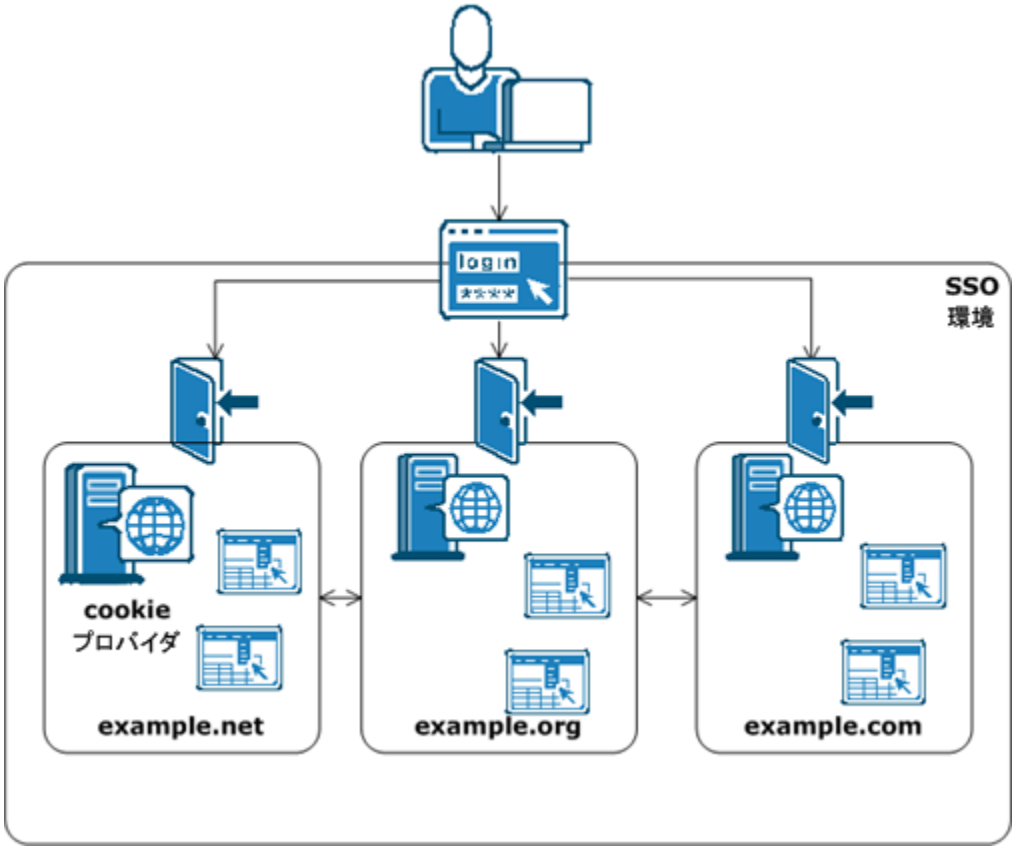


注: 複製ユーザ ディレクトリと共に非複製ポリシー ストアを使用している場合、ユーザ ディレクトリ名は、すべてのポリシー ストアで同一である必要があります。また、セッション チケットを暗号化するセッション チケット キーは、SSO 環境におけるすべてのキー ストアに対して同一でなければなりません。セッション チケットにより、有効なユーザ セッションの持続時間が決定されます。

複数のドメインにおけるシングル サインオン

シングル サインオンを使用しないと、ユーザは、異なる cookie ドメインの別個のサーバ上にある別のアプリケーションやリソースにアクセスするときに、何度もログオンして認証情報を入力することが必要になります。複数の cookie ドメイン間でシングル サインオン情報を渡す機能を使用すると、ある cookie ドメインのサイトで認証されたユーザは、再認証を要求されることなく、別の cookie ドメインのサイトに移動できます。このようなシームレスな移動により、ユーザが関連サイトを使用するときの利便性が向上します。

以下の図に、複数の cookie ドメインにおけるシングル サインオンを示します。



複数の cookie ドメインにおけるシングル サインオン

SiteMinder では、cookie プロバイダとして設定された SiteMinder Web エージェントを使用して、複数の cookie ドメインにおけるシングル サインオンを実装します。

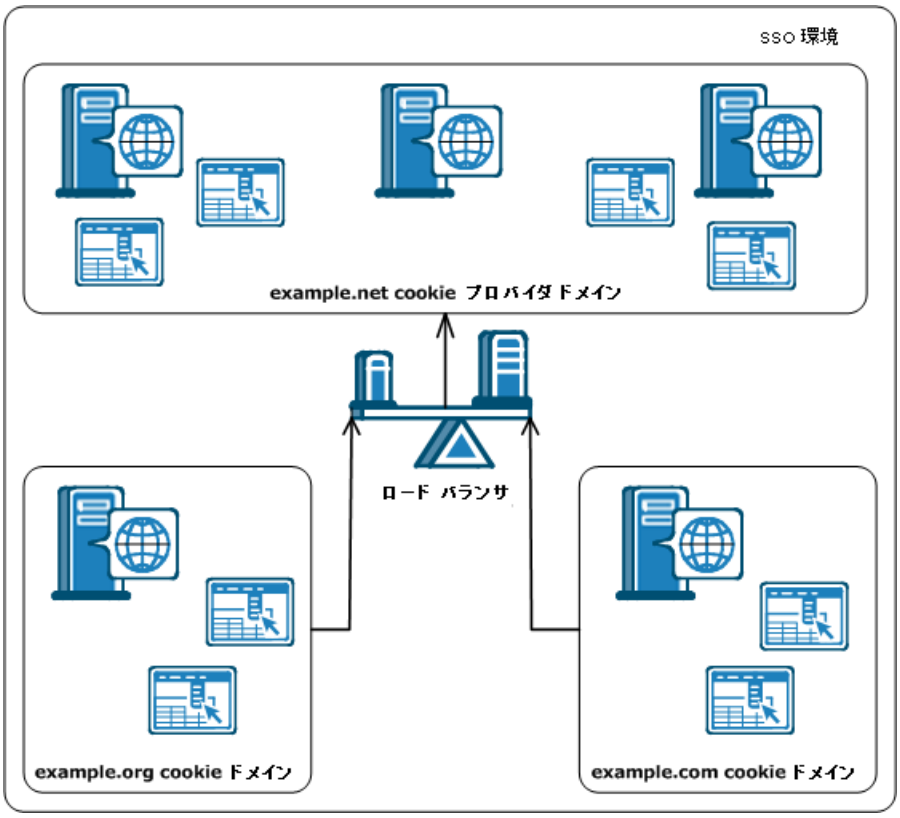
cookie プロバイダの Web エージェントが存在する cookie ドメインのことを、cookie プロバイダ ドメインといいます。シングル サインオン環境において、他の cookie ドメインにあるその他すべての Web エージェントは、1 つの cookie プロバイダを参照しています。

SiteMinder cookie プロバイダは以下の手順を使用して動作します。

1. ユーザがシングル サインオン環境内のドメインの保護されているリソースを要求すると、認証情報が要求されます。
2. ユーザが認証される場合、以下の cookie がユーザのブラウザ内で設定されます。
   - ユーザが認証したドメインのローカル cookie
   - cookie プロバイダによって設定された cookie
3. 以下のいずれかのイベントが発生するまで、ユーザは、再認証を要求されることなく、シングル サインオン環境内のドメイン間を移動できます。
   - ユーザのセッションはタイムアウトになります。
   - (通常はブラウザを閉じることによって)ユーザがセッションを終了します。

シングル サインオン環境内の Web エージェントは、負荷分散する必要があるでしょうか。

SSO 環境内のすべての Web エージェントは 1 つの cookie プロバイダ ドメインを参照する必要があるため、以下の図のように、SSO 環境内の cookie プロバイダ ドメインとその他のドメイン間で、Web サーバのロード バランサを追加します。



example.org cookie ドメイン内の Web エージェント、および example.com cookie ドメイン内の Web エージェントは両方とも、example.net の同じ cookie プロバイダ ドメインを参照しています。ロード バランサは、example.net cookie プロバイダ ドメイン内のすべての Web サーバ間でトラフィックを均等に分散させます。

注: 複数の cookie ドメインにおける SSO では、SSO 環境で使用するユーザ ディレクトリが同一である必要はありません。ただし、複製ユーザ ディレクトリと共に非複製ポリシー ストアを使用している場合、ユーザ ディレクトリ名は、すべてのポリシー ストアに対して同一であることが必要です。また、セッション チケットを暗号化するセッション チケット キーは、SSO 環境におけるすべてのキー ストアに対して同一でなければなりません。セッション チケットにより、有効なユーザ セッションの持続時間が決定されます。

シングル サインオンと認証方式の保護レベル

シングル サインオンを使用することで、あるレルムの認証済みユーザは、2 つ目のレルムが同等かそれ以下の保護レベルの認証方式で保護されていれば、再認証せずにその 2 つ目のレルムのリソースにアクセスすることができます。ユーザが、より高い保護レベルの認証方式で保護されているリソースにアクセスしようとすると、SiteMinder は、認証情報の再入力を求めるプロンプトをユーザに表示します。

SiteMinder では、管理者が 管理 UI を使用して認証方式に保護レベルを割り当てることができます。保護レベルの範囲は 1 から 20 です。1 が最も安全性が低く、20 が最も安全性が高くなります。これらの保護レベルにより、管理者はシングル サインオン環境下のセキュリティおよび柔軟性に関する基準を追加して、認証方式を実装することができます。

たとえば、すべてのユーザに提供されている一連のリソースでは、保護レベル 1 の基本認証方式が使用されているとします。また、会社の重役のみに提供されている別の一連のリソースでは、保護レベル 15 の X.509 証明書方式が使用されています。その場合に、ユーザがベーシック方式で認証を受けた後、証明書方式で保護されたリソースにアクセスを試みると、そのユーザは再認証を求められます。

注: 詳細については、ポリシー サーバ ドキュメントを参照してください。

OPTIONS メソッドを使用するリソースへの自動アクセス許可

SiteMinder Web エージェントでは、OPTIONS メソッドを使用するリソースに対してアクセスが試行されると、認証済みユーザでもクレデンシャルが要求されます。OPTIONS メソッドを使用するリソースの例には以下があります（これに限られるわけではありません）。

- Microsoft® Word 文書
- Microsoft® Excel® スプレッドシート

この要求は、リソースと関連付けられたアプリケーションが OPTIONS メソッドを使用してリクエストを Web サーバに送信するために発生します。このリクエストには SiteMinder cookie が含まれていないため、Web エージェントでは要求が発行されます。

これらのリソースに対する認証要求を防ぐ方法

1. 以下のパラメータの値を yes に設定します。

   autoauthorizeoptions

   HTTP OPTIONS メソッドを使用するリソースに対する全てのリクエストを自動的に許可します。

   このパラメータの値を yes に設定した場合、PersistentCookies パラメータの値は no に設定します。

   制限: yes、no

2. PersistentCookies パラメータの値を no に設定します。

匿名レルム間でのユーザ ID の追跡

匿名ユーザがリソースにアクセスする場合、そのユーザは、SMIDENTITY（匿名）cookie の割り当てを受けます。ユーザは、他のドメインに移動するときに、認証情報を要求されます。正常にログインすると、SMSESSION（ログイン済み）cookie が割り当てられます。

このユーザは、保護された「匿名の」リソース、つまりユーザーが認証情報を提示する必要のないレルム内に存在しているリソースにアクセスする場合、1 人のユーザに対応する両方の cookie を保持している 1 つのドメインに入ることができます。5.x QMR 3 以降の Web エージェントによって保護されているリソースの場合、Web エージェントは SMIDENTITY cookie ではなく、SMSESSION cookie を使用してユーザを識別します。

ユーザが、完全にアップグレードされたドメインから、古いバージョンのエージェントによって SMIDENTITY cookie でユーザが識別されるドメインへと移動する場合、使用される cookie は要求を処理する Web エージェントのバージョンによって異なります。

別個の cookie ドメインに関しては、保護されているリソースがマスタ cookie ドメインに含まれ、匿名リソースが第 2 ドメインに含まれている場合、ユーザは、以下のタスクを行うと、引き続き匿名ドメインの匿名ユーザと見なされます。

1. 最初に匿名ドメインにアクセスします
2. マスタドメインに移動し、ログインします。
3. 匿名ドメインへ戻ります。

シングル サインオンとエージェント キー管理

Web エージェントは、Web エージェント間で情報を渡す cookie の暗号化と復号化にキーを使用します。エージェントは、SiteMinder cookie を受け取ると、キーを使用して cookie の内容を復号化します。キーは、ポリシー サーバと通信するすべての Web エージェントで、同じ値に設定してください。

キーの安全性を確保するため、ポリシー サーバは、これらのキーを生成し、暗号化して、SiteMinder 環境下のすべての Web エージェントに配布することができます。同じキー ストアにすべてのキー情報を保持して共有する大規模な SiteMinder インストール環境でも、自動キー変換によって、エージェント キー管理を簡単に実施することができます。自動キー変換により、キーの完全性も確保されます。

シングル サインオンの設定方法

シングル サインオン環境を設定するには、以下の手順に従います。

1. シングル サインオン環境を構成する cookie ドメインを指定します。
2. シングル サインオン環境において、cookie プロバイダ ドメインとして使用する cookie ドメインを指定します。
3. (管理 UI を使用して) エージェント設定オブジェクトにアクセスするか、(Web サーバ上の) Web エージェント設定ファイルを開き、以下の手順でパラメータを変更します。
   a. RequireCookies パラメータに yes を設定します。

      cookie を要求せずにタイムアウト パラメータを設定すると、Web エージェントは正常に機能しますが、タイムアウトは適用されません。Web エージェントが要求する cookie をユーザのブラウザが受け取らない場合、ユーザはすべての保護対象リソースへのアクセスを拒否されます。

      注: 詳細については、ポリシー サーバ ドキュメントを参照してください。
   b. 必要に応じて、PersistentCookies パラメータに yes を設定して、設定したセッション タイムアウトまで cookie を存続させるようにします。

      このパラメータを no のままにしておくと、cookie は 1 つのブラウザ セッションに限って存続することになります。
   c. CookieDomain パラメータについて、cookie ドメインは Web エージェントがインストールされたシステムのローカル cookie ドメインであることを確認します。たとえば、.mycompany.com が該当します。必要ならば、ドメインを変更します。この値は、大文字と小文字が区別されます。
   d. 以下に示した構文を使用して、CookieProvider パラメータを cookie プロバイダ ドメインに設定します。

      http://*server.domain:port*/siteminderagent/SmMakeCookie.ccc

      ここで、*server.domain:port* は cookie プロバイダとして動作している Web エージェントがある Web サーバの完全修飾ドメイン名 (myserver.mysite.com など) です。cookie プロバイダ名には、拡張子 .ccc が必要です。
   e. cookie プロバイダが適切な関連する MIME タイプ (.ccc) で設定されていることを確認します。
   f. 永続的または一時的な cookie を有効にした場合は、(IP アドレスを比較するために) 適切なタイプの IP チェックを有効にします。
4. オプションで、シングル サインオンの他の設定を変更します。

5. Web エージェント設定ファイルを修正することによってパラメータを編集した場合は、Web サーバを再起動して変更を反映させます。

基本認証用の cookie が必要

以下のパラメータを使用して、SiteMinder が cookie を必要とするかどうかを制御できます。

RequireCookies

SiteMinder が cookie を必要とするかどうかを指定します。SiteMinder では以下を実行するために cookie を使用します。

- シングル サインオン環境を保護する
- セッション タイムアウトを追跡する
- アイドル タイムアウトを追跡する

重要: cookie を要求するように Web エージェントを設定する場合、ユーザ側では、Web ブラウザに HTTP cookie を受け取るように設定することが必要です。ブラウザが cookie を受け取らない場合は、エージェントからエラーメッセージが返され、すべての保護されたリソースに対するユーザのアクセスが拒否されます。

デフォルト: yes

RequireCookies は、ポリシー サーバの設定時に基本認証が設定された場合に限って役立つ特別な設定です。この設定は、基本認証ヘッダを含め、HTTP リクエストの処理を成功させるために、SMSESSION または SMCHALLENGE cookie のどちらかを要求するよう、エージェントに指示します。

Web エージェントが cookie を要求していなくてもユーザの Web ブラウザが cookie を受け入れている場合、Web エージェントは正常に機能します。ただし、ユーザが予期せずに認証情報を要求されたり、Web エージェントがタイムアウトを厳密に実行できなかったりすることがあります。

cookie を必要とするには、RequireCookies パラメータを yes を設定します。

永続的 cookie の設定

複数のブラウザセッションでシングル サインオンを使用するには、永続的な cookie を使用します。永続的な cookie を設定すると、ユーザが SiteMinder セッションの有効期限が切れる前にブラウザ セッションを終了し、新しいブラウザ セッションを開始しても、シングル サインオンはそのまま有効になっています。

Web エージェント 5.x QMR1 以降では、永続的な cookie は、クライアント システムのハードディスクに書き込まれ、最大セッション タイムアウトの設定値 + 7 日間は有効です。通常は、有効期限を過ぎると永続的な cookie は Web ブラウザの cookie ファイルから削除されますが、ブラウザによってその処理方法は異なります。

永続的な cookie を設定する方法

1. PersistentCookies パラメータを yes を設定します。

   SMSESSION cookie は永続的になります。

2. TransientIDCookies パラメータを no に設定します。

   SMIDENTITY cookie は永続的になります。

cookie ドメインの指定

CookieDomain パラメータは、Web エージェントをインストールした Web サーバの cookie ドメイン（netegrity.com など）を定義します。cookie ドメインは、Web エージェントのインストール時に指定します。

必要に応じて、ドメインを変更できます。この値は、大文字と小文字が区別されます。このパラメータを設定するときは、以下の点に注意してください。

- CookieDomain を none に設定した場合、Web エージェントは、Web エージェントをホストしている Web サーバに対してのみ cookie を生成します。これは、サーバ専用の cookie です。たとえば、myserver.netegrity.com です。
- ローカル設定ファイル内で CookieDomain を空白のままにするか二重引用符（""）に設定した場合、Web エージェントは HTTP_HOST ヘッダから cookie ドメインを取り出し、CookieDomainScope パラメータに基づいて値を決定します。

  CookieDomainScope パラメータが 0（デフォルト）に設定されている場合、エージェントはサーバ専用の cookie を作成することなく、そのホストに最も特定された cookie ドメインを選択します。これは、cookie ドメイン myserver.netegrity.com に対応するドメインが netegrity.com であり、myserver.metals.ne.com に対応するドメインが .metals.ne.com であることを意味します。CookieDomainScope パラメータが 2 に設定されている場合、cookie ドメインはそれぞれ .netegrity.com と .ne.com になります。
- CookieDomain パラメータを特定のドメイン（.netegrity.com など）に設定した場合は、それが Web エージェントによって使用されるドメインになります。

cookie プロバイダの指定

cookie プロバイダを使用するには、以下のパラメータを使用してそのロケーションを指定する必要があります。

CookieProvider

cookie プロバイダとして動作している Web エージェントがある Web サーバの URL を(完全修飾ドメイン名を使用して)を指定します。cookie プロバイダ名には、拡張子 .ccc が必要です。

- IIS、Sun Java System、Domino Web サーバでは、以下の URL 構文を使用します。

  http://*server.domain:port*/siteminderagent/SmMakeCookie.ccc

- Apache および Apache ベースの Web サーバでは、以下の URL 構文を使用します。

  http://*server.domain:port*/SmMakeCookie.ccc

このパラメータは以下のパラメータに影響します。

- CCCExt
- SessionUpdatePeriod

デフォルト: デフォルトなし

例: (IIS、Sun Java System、および Domino Web サーバ)
http://server1.myorg.com:80/siteminderagent/SmMakeCookie.ccc

例: (Apache および Apache ベースの Web サーバ)
http://server1.myorg.com:80/SmMakeCookie.ccc

cookie プロバイダを指定する方法

1. CookieProvider パラメータに、cookie プロバイダとして指定する Web サーバの URL を設定します。
2. CCCExt パラメータの値が .ccc に設定されていることを確認します。
3. IgnoreExt パラメータの値に .ccc 拡張子を追加します。
4. (任意)セッション更新期間を変更します。

   これで、cookie プロバイダが指定されます。

詳細情報

各認証情報コレクタ用の MIME タイプの設定 (P. 265)

セッション更新期間の変更

以下のパラメータを使用して、Web エージェントが cookie プロバイダにリクエストをリダイレクトして新しい cookie を設定する間隔を指定することができます。

SessionUpdatePeriod

新しい cookie を設定する目的で、Web エージェントが要求を cookie プロバイダにリダイレクトする頻度（秒単位）を指定します。マスタ cookie を更新すると、SiteMinder セッションのアイドル タイムアウトが原因でその cookie が期限切れになる確率を低下させることができます。

デフォルト: 60

セッション更新期間を変更するには、以下の手順に従います。

1. CookieProvider パラメータが定義されていることを確認します。
2. 目的の間隔を反映するように、SessionUpdatePeriod パラメータの秒数を変更します。

   セッション更新期間が変更されます。

使い捨てのセッション cookie の有効化

一度だけ使用されるセッション cookie を SiteMinder に作成させることによって、環境のセキュリティを強化することができます。使い捨てのセッション cookie を使用することにより、以下のアイテムへのアクセス権のあるユーザが誰でもセッション cookie をコピーしてそれを再使用し、リソースへの不正なアクセスを取得するの防ぐことができます。

- Web サーバ ログ
- SiteMinder Web エージェント ログ
- ドメイン間に置かれている、侵害される可能性のあるプロキシ サーバ(クロスドメイン シングル サインオンの場合)

以下のパラメータを設定することにより、SiteMinder が使い捨てのセッション cookie と複数使用のセッション cookie のどちらを使用するかを制御できます。

StoreSessioninServer

使い捨てのセッション cookie を使用するかどうか指定します。StoreSessioninServer パラメータの値が yes の場合は、使い捨てのセッション cookie が作成され、セッション サーバに格納されます。cookie プロバイダおよび Web エージェントは、セッション サーバの cookie にアクセスします。

cookie プロバイダおよび Web エージェントは、URL 内のセッション cookie を、セッション サーバ上に格納された使い捨てのセッション cookie に対応する GUID に置き換えます。

StoreSessioninServer パラメータの値が no の場合、セッション cookie は URL で直接渡されます。

デフォルト: No

使い捨てのセッション cookie を有効にする方法

1. 環境が以下の条件を満たすようにします。
   - SiteMinder 6.0 SP5 QMR1 以上を使用するために、Web エージェントと cookie プロバイダをアップグレードする。
   - Web エージェントおよび cookie プロバイダ内で DefaultAgentName パラメータの値を使用する。
   - ポリシー サーバが有効なセッション ストアで設定されている。
2. Web エージェントおよび cookie プロバイダで、StoreSessioninServer パラメータの値を yes に設定します。

セッション cookie ドメインの検証

以下のパラメータを使用して SiteMinder にセッション cookie のドメインを検証させることにより、不正なユーザがハイジャックし、SiteMinder セッション cookie を再利用しようとするリスクを減らすことができます。

TrackSessionDomain

セッション cookie の中にセッション cookie の対象ドメインを暗号化して格納するように Web エージェントに指示します。後続のリクエストでセッション cookie が提示されると、Web エージェントは、セッション cookie 内にある対象ドメインを、要求されたリソースのドメインと比較します。ドメインが一致*しない* 場合、Web エージェントはリクエストを拒否します。

たとえば、このパラメータの値が yes に設定されているときに、operations.example.com での使用を目的とするセッション cookie が finance.example.com で提示された場合、Web エージェントはその cookie を拒否します。

デフォルト: no

SiteMinder にセッション cookie のドメインを検証させるためには、TrackSessionDomain パラメータの値を yes に設定します。

セッション cookie の作成または更新の防止

Microsoft Outlook Web Access など、一部の Web アプリケーションでは、ユーザがアプリケーションをアクティブに使用していない場合でも、HTTP リクエストがバックグラウンドで行われます。たとえば、ユーザがサーバ上で新しい電子メールをアクティブに確認していない場合でも、Web Access アプリケーションは HTTP リクエストを行います。

ユーザがアイドル状態だったとしても、セッションが期限切れにならないように、これらのリクエストによって SMSESSION cookie が更新されることがあります。セッションが通常どおり期限切れになるように、これらのバックグラウンド リクエストの際に Web エージェントがセッション cookie を作成または更新できないようにすることができます。

SMSESSION cookie の作成または更新を阻止するには、以下の手順に従います。

1. 以下のどちらか、または両方のパラメータを設定します。

   OverlookSessionForMethods

   Web エージェントがこのパラメータ内に列挙されたメソッドに対してすべての HTTP リクエストのリクエスト メソッドを比較するかどうかを指定します。一致した場合、Web エージェントは SMSESSION cookie の作成も更新も行いません。さらに、cookie プロバイダ（設定されている場合）はそのリクエストに対して更新されません。

   デフォルト: デフォルトなし

   OverlookSessionForUrls

   Web エージェントが、すべての HTTP リクエストの URLs を、このパラメータに示されている URLs と比較するかどうかを指定します。一致した場合、Web エージェントは SMSESSION cookie の作成も更新も行いません。さらに、cookie プロバイダ（設定されている場合）はそのリクエストに対して更新されません。

   デフォルト: デフォルトなし

   例: /MyDocuments/index.html のような相対 URL を使用します。絶対 URL（http://fqdn.host/MyDocuments/index.html）は使用しません。

注: 前述のパラメータの両方を設定すると、URL の前にメソッドが処理されます。

メソッドと URI に基づいたセッション cookie の作成または更新の防止

Microsoft Outlook Web Access など、一部の Web アプリケーションでは、ユーザがアプリケーションをアクティブに使用していない場合でも、HTTP リクエストがバックグラウンドで行われます。たとえば、ユーザがサーバ上で新しい電子メールをアクティブに確認していない場合でも、Web Access アプリケーションは HTTP リクエストを行います。

ユーザがアイドル状態だったとしても、セッションが期限切れにならないようにこれらのリクエストによって SMSESSION cookie が更新されます。セッションが一般的に期限切れになるように、これらのバックグラウンド リクエストの際に Web エージェントがセッション cookie を作成または更新できないようにすることができます。

メソッドと URI に基づいた作成または更新を防ぐ方法

1. 以下のパラメータをすべて設定します。

   OverlookSessionForMethods

   Web エージェントがこのパラメータ内に列挙されたメソッドに対してすべての HTTP リクエストのリクエスト メソッドを比較するかどうかを指定します。一致した場合、Web エージェントは SMSESSION cookie の作成も更新も行いません。さらに、cookie プロバイダ（設定されている場合）はそのリクエストに対して更新されません。

   デフォルト: デフォルトなし

   OverlookSessionForMethodUri

   Web エージェントが、すべての HTTP リクエストの URI を、このパラメータに示されている URI と比較するかどうかを指定します。一致した場合、Web エージェントは SMSESSION cookie の作成も更新も行いません。さらに、cookie プロバイダ（設定されている場合）はそのリクエストに対して更新されません。

   デフォルト: デフォルトなし

   注: メソッドは URI の前に処理されます。

安全な cookie の設定

以下のパラメータを使用して、安全な（HTTPS）接続上の保護されている Web サーバとリ要求ブラウザの間でのみセッション cookie が送信されるように指定することができます。

UseSecureCookies

安全な（HTTPS）接続を使用して、Web サーバに cookie を送信します。このパラメータを使用することで、ブラウザと Web サーバの間のセキュリティを向上させることができます。

この設定が有効な場合、シングル サインオン環境のユーザは、SSL Web サーバから非 SSL Web サーバに移動するときに再認証する必要があります。セキュア cookie は、従来の HTTP 接続を介して渡すことはできません。

デフォルト： No

SSL 接続経由で cookie を送信するには、UseSecureCookies パラメータを yes に設定します。

複数ドメインにわたる安全な cookie の設定

UseSecureCookies パラメータの設定により、それらの間の接続が安全な（HTTPS）場合に*ローカル* cookie を要求ブラウザ セッションに返すためにのみ Web エージェントが設定されます。Web エージェントが cookie プロバイダとしても設定されている場合、UseSecureCookies は他の cookie ドメインのリソースへのアクセスに対するリダイレクトされた要求には適用されません。

cookie プロバイダとして機能する Web エージェントが、安全な cookie を使用するようにも設定されている場合に、別の cookie ドメインの Web エージェントにのみ cookie を返すように設定するには、UseSecureCookies を有効にし、以下のパラメータも設定する必要があります。

UseSecureCPCookies

UseSecureCPCookies が Yes に設定されていると、cookie プロバイダは、セキュア cookie の使用も設定されている（つまり、UseSecureCookies も有効である）別の cookie ドメイン内の Web エージェントにのみ cookie を送信します。

この設定と UseSecureCookies が両方とも有効な場合、複数ドメインのシングル サインオン環境内のユーザは、SSL の Web サーバから別の cookie ドメインの非 SSL の Web サーバに移動するときに、再認証する必要があります。セキュア cookie は、従来の HTTP 接続を介して渡すことはできません。

デフォルト: No

複数のドメインの SSL 接続に cookie を送るには、cookie プロバイダで UseSecureCookies と UseSecureCPCookies を yes に設定します。

識別 cookie の制御

TransientIDCookies パラメータは、エージェント ID cookie（SMIDENTITY）が一時的か永続的かを指定します。

永続的 cookie は、クライアント システムのハード ディスクに書き込まれます。Web エージェント 5.x QMR1 より前のバージョンでは、永続的な cookie は 7 日間にわたって有効でした。Web エージェント 5.x QMR1 以降では、永続的な cookie は、設定済みの最大セッションタイムアウト + 7 日間にわたって有効です（最大セッション タイムアウトの設定には 管理 UI を使用します）。通常、有効期限を過ぎると永続的な cookie は Web ブラウザの cookie ファイルから削除されます。ただし、永続的な cookie の処理方法は、ブラウザによって異なります。デフォルトでは、Web エージェントは永続的な cookie を使用しません。Web エージェントは、過渡的な cookie を使用します。

複数のブラウザセッションでシングル サインオンを使用するには、永続的な cookie を使用します。永続的な cookie を設定すると、ユーザは、SiteMinder セッションの有効期限が切れる前にブラウザ セッションを終了できます。新しいブラウザ セッションを開始しても、シングル サインオン機能は有効のままです。

永続的な cookie がハードディスクに書き込まれるのに対し、過渡的な cookie はハードディスクに書き込まれることはなく、設定済みのセッションタイムアウトの対象になりません。過渡的な cookie は、使用中の cookie フォルダ内にとどまります。

ID cookie を永続的なものにしたい場合、TransientIDCookies を no に設定します。ID cookie を一時的なものにしたい場合、デフォルト値である yes のままにしておきます。

対応する IP チェック機能を確実に設定してください。

セッション猶予期間の変更

通常、Web ページは数多くのリソースで構成され、そのすべてのリソースが Web エージェントによって潜在的に保護されています。1 つのリクエストに関連付けられている各リソースに対して、1 つのセッション cookie が生成されます。1 つのユーザ リクエストに対して複数のセッション cookie を生成するオーバーヘッドを取り除くには、以下のパラメータを設定します。

SessionGracePeriod

SiteMinder セッション（SMSESSION） cookie が再生成されない秒数を指定します。以下の条件が *すべて* 満たされる場合、cookie は再生成されません。

- URL SMSESSION cookie が存在しない。
- 現在の時刻と受け取った SMSESSION cookie の最終アクセス時刻の差が SessionGracePeriod 以下である。
- 現在の時刻と受け取った cookie がアイドルになった時刻の間隔が 2 つの猶予期間を超えている。たとえば猶予期間が 25 分でアイドル タイムアウトが 60 分である場合、セッションがアイドルになる前に残っている時間が 2 つの猶予期間（50 分）に満たないので、SiteMinder は 10 分後にセッション cookie を再生成します。

デフォルト: 30

セッション猶予期間を変更するには、以下の手順に従います。

1. SessionGracePeriod パラメータの値を変更します。
2. ステップ 1 で SessionGracePeriod パラメータの値を増加した場合は、管理 UI を使用して、すべてのレルムで以下の値の両方が SessionGracePeriod パラメータの値を超えて *いない* ことを確認します。
   - セッション タイムアウト値
   - アイドル タイムアウト値

   セッション猶予期間が変更されます。

   注: セッション タイムアウトは、レルムの設定の一部なので、管理 UI を使用して設定します。セッション タイムアウトの設定方法の詳細については、ポリシー サーバのマニュアルを参照してください。

保存された認証情報のタイムアウトの設定

ユーザが認証情報を保存することを選択した場合、ポリシー サーバは、そのユーザの認証情報を保管する永続的な cookie を作成することを Web エージェントに指示します。この cookie を使用することで、Web エージェントは、ユーザに認証情報を再要求する代わりに、cookie に保存されている認証情報に基づいてユーザを認証します。永続的な cookie の保存期間は、以下のパラメータを使用して制御できます。

SaveCredsTimeout

ユーザ認証情報が含まれている永続的な cookie が保存される時間数を指定します。この時間中に、Web エージェントは、cookie 内に保存されたデータでユーザを認証します。この時間を過ぎると、cookie は削除され、Web エージェントは再度ユーザ認証を試みます。

デフォルト: 720（30 日）

保存された認証情報のタイムアウトを設定するには、SaveCredsTimeout パラメータに目的の時間数を入力します。

注: 詳細については、ポリシー サーバドキュメントを参照してください。

複数レルム間でタイムアウトを適用する方法

ユーザ セッション タイムアウトは、ユーザが最初にログオンしたレルムによって制御されます。シングル サインオンによってユーザが新しいレルムに入った場合、新しいレルムに関するタイムアウト値は、引き続き、最初のレルムに初期のログインをした際に確立されたセッションによって制御されます。別のレルムに対する別のタイムアウト値があり、各レルムにそれぞれのタイムアウト値を使用させる場合は、元のレルムのタイムアウトを無視できます。

既にタイムアウトになったユーザは、他のレルムにログインする際に、認証情報を再度要求されます。たとえば、Realm1 の Idle Timeout が 15 分であり、Realm2 の Idle Timeout が 30 分である場合、ユーザが Realm1 で 20 分のアイドル時間を過ごした後、Realm2 にログインしようとすると、認証情報を要求されます。

元のレルムのタイムアウトを無視するには、Web エージェントとレルムを以下の手順で説明するように設定します。

1. EnforceRealmTimeouts パラメータの値を yes に設定します。
2. 管理 UI を使用して以下のタスクを実行します。
   a. 元のタイムアウトに代わるレルム（SSO 機能がユーザのアクセスを許可するレルム）ごとに、以下を実行します。
      - 最大タイムアウト値を無視するには、WebAgent-OnAuthAccept-Session-Max-Timeout レスポンス属性を使用して、レスポンスを作成します。
      - アイドル タイムアウト値を無視するには、WebAgent-OnAuthAccept-Session-Idle-Timeout レスポンス属性を使用して、レスポンスを作成します。
   b. 前のレスポンスをそれぞれ OnAuthAccept ルールにバインドします。

   注: 作成するレスポンスの詳細については、「ポリシー サーバ構成ガイド」を参照してください。

セッション タイムアウト後のユーザのリダイレクト

ユーザが 管理 UI でレルムを設定すると、セッション タイムアウトが設定されます。ユーザの SiteMinder セッションがタイムアウトしたとき、Web エージェントは以下の処理のいずれかを行います。

- ユーザに認証情報を再要求する
- ユーザを別の URL へリダイレクトする

リダイレクト URL が指定されている場合、ユーザにはその宛先ページが表示されます。ページが保護されていない場合は、そのページへの直接アクセス権がユーザに付与されます。ページが保護されている場合は、ページへのアクセス権の付与に先立って、ユーザに対して認証情報が要求されます。リダイレクト URL が指定されていない場合、Web エージェントはセッション タイムアウト後にユーザに認証情報を再要求します。

セッションがタイムアウトしたユーザをカスタマイズされた Web ページの URL にリダイレクトできます。カスタマイズされた Web ページでは、セッションが終了した理由とセッションを再確立する方法が説明されます。たとえば、「You have been logged out automatically as a security precaution. Please login again to continue.」などのメッセージが表示されるカスタム Web ページを作成できます。

セッションがタイムアウトになった後、ユーザが別ページにリダイレクトされなければ、SiteMinder は再度ユーザに認証を要求します。再認証が要求されている理由が理解できないため、これによってユーザが混乱する可能性があります。

セッション タイムアウトの後にユーザを別の URL にリダイレクトする方法

1. エージェント設定オブジェクトまたはローカル設定ファイルに以下のパラメータを追加します。

   IdleTimeoutURL

   セッションのアイドル タイムアウトが発生したときに、Web エージェントがユーザをリダイレクトする先の URL を指定します。

   例: http://example.mycompany.com/sessionidletimeoutpage.html

   注: IdleTimeoutURL は、非永続セッションにのみ使用してください。永続セッションに対して設定した場合は無効になります。

   MaxTimeoutURL

   セッションの最大タイムアウトが発生したときに、Web エージェントがユーザをリダイレクトする先の URL を指定します。

   例: http://example.mycompany.com/maxtimeoutpage.html

デフォルト: デフォルトなし

2. 前のパラメータごとに 1 つずつ URL を入力します。すべてのパラメータに対して同じ URL を使用することも、それぞれ異なる URL を使用することもできます。

   IdleTimeoutURL と MaxTimeoutURL が設定されている場合に、(ポリシー サーバで設定されている) セッションのアイドル タイムアウト値と最大タイムアウト値に同時に達すると、タイムアウトが発生したときにユーザは MaxTimeoutURL に指定された URL にリダイレクトされます。

検証期間と期限切れになった cookie URL での悪用からのセッション cookie の保護

SiteMinder では、以下のアイテムにアクセスできる管理者やその他のユーザによって SiteMinder セッション cookie が侵害される可能性を大幅に減らすことができる時間ベースのセッション cookie パラメータが使用されます。

- Web サーバ ログ
- SiteMinder Web エージェント ログ
- クロスドメイン シングル サインオンの場合にドメイン間に置かれている、侵害される可能性のあるプロキシ サーバ

これらの時間ベースのセッション cookie パラメータは、セッション cookie に「生成日」の概念を付加します。リダイレクトの結果としてセッション cookie（URL セッション cookie）を受け取るエージェントは、cookie の生成日名と値のペアを探し、この値を設定パラメータ CookieValidationPeriod に設定されている値と比較します。生成日の値に CookieValidationPeriod パラメータの値を加えた値が現在の時刻に達しない場合、cookie は拒否されます。

悪用からセッション cookie を保護するには、以下のパラメータを設定します。

CookieValidationPeriod

エージェントがセッション cookie 受け取る期間を（秒単位で）指定します。この期間を過ぎると、セッション cookie は受け取られません。このフィールドが使用されていないか、ゼロに設定されている場合、アイドル タイムアウトおよび最大セッション タイムアウト値に達すると、セッション cookie は期限切れになります。

デフォルト: 空白.

ExpiredCookieURL

（省略可）セッション cookie の期限切れ後にエージェントがユーザをリダイレクトする先の URL を指定します。セッションの作成日も CookieValidationPeriod も設定されていない場合、エージェントはこの設定を無視し、cookie を通常どおり処理します（後方互換性）。

SDK サードパーティ cookie のサポート

組織で SiteMinder 以外の Web エージェントを使用する場合は、以下のパラメータを使用して、シングル サインオンがサポートされるようにそれらの Web エージェントを設定できます。

AcceptTPCookie

Web エージェントがサードパーティ(SiteMinder 以外)の Web エージェントによって作成されたセッション(SMSESSION) cookie を受け取ることを可能にします。サードパーティ エージェントは、SiteMinderSDK を使用して、SMSESSION cookie を生成したり読み取ったりします。

デフォルト: デフォルトなし

注: 詳細については、「SiteMinder Developer's Guides」を参照してください。

Web エージェントが SiteMinder 以外の Web エージェントによって作成されたセッション cookie を受け入れることができるようにするには、AcceptTPCookie パラメータを yes に設定します。

保護されていないリソースにおける cookie プロバイダの無視

Web エージェントはデフォルトではすべての要求を cookie プロバイダに転送します。保護なしリソースがある場合は、以下のパラメータを使用してネットワーク トラフィックを削減することができます。

IgnoreCPForNotprotected

保護されていないリソースの要求に関して、cookie プロバイダがクエリを行わないようにします。このパラメータを no に設定すると、Web エージェントによってすべての要求が cookie プロバイダに送られます。従来の(非フレームワーク)エージェントでは、このパラメータの値が Web エージェントのログ ファイルに表示されるように cookie プロバイダを設定する必要があります。

デフォルト: No

保護なしリソースが要求される場合に、Web エージェントが cookie プロバイダに問い合わせないようにするには、IgnoreCPForNotprotected パラメータの値を yes に設定します。

POST 要求における cookie プロバイダの無視（フレームワーク エージェントのみ）

以下のパラメータにより、フレームワーク エージェントもある環境で従来のエージェントを cookie プロバイダとして使用できます。

LegacyCookieProvider

フレームワーク エージェントが cookie プロバイダに POST 要求を送るかどうかを制御します。フレームワーク エージェントが cookie プロバイダとして動作する従来のエージェントに POST 要求を送ると、リダイレクトされた要求は代わりに GET になって失敗します。no に設定すると、フレームワーク エージェントは cookie プロバイダに POST 要求を送ります。yes に設定すると、フレームワーク エージェントは cookie プロバイダに POST 要求を送りません。

集中的なエージェント設定を使用している場合は、このパラメータをエージェント設定オブジェクトに追加する必要があります。このパラメータは、すでにローカル設定ファイル内に存在します。

デフォルト: なし（送信された POST 要求）

従来のエージェントをフレームワーク エージェントのある cookie プロバイダとして使用するには、LegacyCookieProvider パラメータを yes に設定します。

cookie ドメインの強制

完全修飾ドメイン名を使用すると、cookie が正しく動作することを保証できます。以下のいずれかの条件を満たす URL リクエスト内のホスト名に cookie ドメインを追加するようにエージェントに強制することができます。

- 要求によってドメインが指定されない
- 要求に IP アドレスのみが含まれる

エージェントに cookie ドメインの追加を強制する方法

1. ForceCookieDomain パラメータに yes を設定します。
2. ForceFQHost パラメータに yes を設定します。

   必要に応じて Web エージェントはホスト名にその cookie ドメインを追加します。

cookie ドメイン解決の実装

自動的なドメイン解決を実装するには、CookieDomain パラメータをコメント アウトするか、そのパラメータを none に設定して、発行元のサーバ上でのみ有効な cookie を作成するよう Web エージェントに指示します。

さらに、CookieDomainScope パラメータに値を追加して、cookie ドメインを定義することもできます。有効範囲には、ドメイン名を構成するセクションの数を、ピリオドで区切って指定します (ドメインは必ず「.」で始まります)。

CookieDomainScope の値が 0 である場合、特定のホストに対して最も具体的な有効範囲を使用するようエージェントに指示します。1 という値 (たとえば、.com という cookie ドメインを生じさせます) は、HTTP 仕様により許可されていません。2 という値は、最も一般的な有効範囲を使用するようエージェントに指示します。

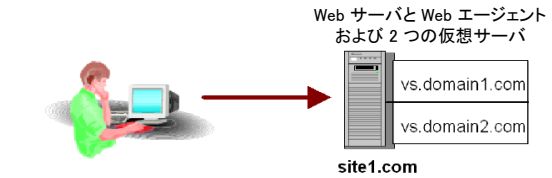
以下の表に、ドメイン名と CookieDomainScope の値をいくつか示します。

ドメイン名	cookie ドメインの有効範囲の値	cookie ドメイン
server.myorg.com	2	.myorg.com
server.division.myorg.com	3	.division.myorg.com
	2	.myorg.com
server.subdivision.division.myorg.com	4	.subdivision.division.myorg.com
	3	.division.myorg.com
	2	.myorg.com

たとえば、division.myorg.com ドメインの有効範囲は 3 です。デフォルトでは、Web エージェントは有効範囲として 2 を想定しています。cookie ドメインの有効範囲を 1 にすることはできません。

cookie ドメインの自動解決

リクエストの受信ホスト ヘッダに基づいて受信要求の cookie ドメインを判別するように、Web エージェントを設定できます。エージェントはホスト ヘッダから cookie ドメインを判断できるので、静的な cookie ドメイン値を設定する必要はありません。このような柔軟性があるので、以下の図に示すように、さまざまな cookie ドメインにある仮想サーバを、Web エージェントで保護するように設定することができます。



Web エージェントは以下のリクエストに対応：

- http://w1.site1.com/
- http://w2.vs.domain1.com/
- http://w3.vs.domain2.com/

完全修飾ドメイン名の強制

cookie が正しく動作するには、完全修飾ドメイン名を使用する必要があります。Web エージェントでは、ユーザが受信リクエストに入力する URL のフォーマットに関係なく、完全修飾ドメイン名を HTTP リクエストに使用することができます。受信ホスト ヘッダに基づいて cookie ドメインを解決するエージェントの機能と共に完全修飾ドメイン名を使用するように強制すると、ネットワーク上に多くのドメインを持つサイトを SiteMinder を使って管理しやすくなります。管理者は、以下のパラメータを使用して、URL のフォームが有効なサイトにユーザがアクセスできるようにして、引き続き cookie をサポートできます。

ForceFQHost

完全修飾ドメイン名を使用するように Web エージェントに強制します。このパラメータは設定されたドメイン ネーム システム(DNS)サービスを使用して、URL リクエストの中に存在しているホスト名に対して、cookie ドメインを強制的に追加します。エージェントではなく DNS サービスを使用します。Web エージェントは、URL の一部が含まれたリクエストを受信すると、元の URI に指定されている同じ転送先リソースにそのリクエストをリダイレクトします。リダイレクト要求では、完全修飾ホスト名が使用されます。完全修飾ホスト名は、設定されている DNS サービスを使用して Web エージェントが決定するものです。このパラメータを ForceCookieDomain パラメータと組み合わせて使用すると、機能を追加できます。

デフォルト: No

例: Web エージェントが http://host1/page.html から要求を受け取ると、それは http://host1.myorg.com/page.html で応答します。Web エージェントが http://123.113.12.1/page.html などの要求を受け取ると、それは http://host1.myorg.com/page.html で応答します。

注: これらの例のように変換されるのは、適切な DNS 検索テーブルが設定されている場合のみです。部分的なドメインが入力された場合、DNS 検索がそのドメインを解決できるかどうかによって、結果は異なります。解決の結果が無効なホストである場合、エラーになります。 多くの場合、そのような要求は Web サーバに届くことすらできません。

完全修飾ドメイン名を使用するように Web エージェントを設定するには、ForceFQHost パラメータを yes に設定します。

cookie ドメインの変更

以下の状況では cookie ドメインを指定する必要があります。

- 別の cookie ドメインの仮想サーバを設定する場合、Web エージェントはユーザ要求内の受信ホスト ヘッダ値を使用して cookie ドメインを決定するため、(空の値の) cookie ドメインを指定すべきではありません。
- フォーム認証方式を設定する場合、認証方式のサーバ名の cookie ドメインを指定します。たとえば、cookie ドメインが .myorg.com である場合、サーバ名は server1.myorg.com のようになります。小文字値を使用することをお勧めします。

  注: 詳細については、ポリシー サーバ ドキュメントを参照してください。

cookie ドメインを変更するには、以下のパラメータを設定します。

CookieDomain

Web エージェントのインストール時に指定した Web エージェントの cookie ドメインを定義します。この場合、完全修飾ドメイン名を指定する必要があります。つまり、ドメインには、少なくとも 2 つのピリオドが含まれている必要があります。たとえば、.myorg.com という cookie ドメインは以下のサーバと一致します。

- w1.myorg.com
- w2.myorg.com
- w3.sales.myorg.com

このドメイン内のすべての Web サーバは、ユーザのブラウザとの間で cookie を送受信できます。同一 cookie ドメイン内のサーバは、cookie を使用してユーザの認証情報を確認します。

デフォルト: 空白

例: .mycompany.com

注: この値では、大文字と小文字が区別されます。

シングル サインオンと併用する場合の SecureUrls の設定

シングル サインオン ネットワーク内に SecureUrls 機能をサポートする Web エージェントと SecureUrls 機能をサポートしない別のエージェントが存在する場合、ユーザが保護されたシングル サインオン リソースを要求すると内部サーバ エラー メッセージが表示されることがあります。

SecureUrls がサポートされる Web エージェントのログには、以下のように、サーバ エラーの理由が表示されます。

エラー: 要求を処理できません。SecureUrls が無効です。

注: シングル サインオン環境内のすべての Web エージェントについて、SecureUrls パラメータに同じ値を設定する必要があります。SiteMinder は、SecureUrls パラメータの値が異なる Web エージェント間の相互運用性をサポートしていません。

完全ログオフの仕組み

完全ログオフを使用すると、Web 開発者は、ユーザ セッションからユーザを完全にログオフさせることができます。これにより、ユーザは Web ブラウザを終了せずにセッションを終了できるようになり、無許可のユーザは開いているセッションを不正に制御できなくなるため、リソースが保護されます。

完全ログオフでは以下の手順が使用されます。

1. ユーザがボタンをクリックするとログ オフします。
2. Web エージェントは、作成されたカスタマイズ ログオフ ページにユーザをリダイレクトします。
3. Web エージェントは、ユーザのブラウザからセッション cookie と認証 cookie を削除します。
4. Web エージェントは、ローカル cookie ドメインと cookie プロバイダ ドメインからもセッション cookie を削除します。このドメインは、シングル サインオン環境用に指定したものです。
5. Web エージェントはポリシー サーバを呼び出して、セッション情報を削除するように指示します。

   ユーザは完全にログオフされます。

完全ログオフの設定

完全ログオフでは、以下のパラメータで作成するカスタム ログオフ ページが使用されます。

LogOffUri

完全なログ オフを有効にし、正常にログ オフした後にユーザに表示される Web サーバ上のカスタム Web ページの場所を指定します。ブラウザ キャッシュ内に格納できないようにこのページを設定する必要があります。設定しなかった場合、ブラウザは、ユーザをログ オフせずに、そのキャッシュからログオフ ページを表示する場合があります。これは、不正なユーザにセッションの支配権を握る機会を与える場合があります。

注: CookiePath パラメータが設定されているときは、LogOffUri パラメータの値が同じ cookie パスを指している必要があります。たとえば、CookiePath パラメータの値が example.com に設定されている場合、LogOffUri は example.com/logoff.html を指している必要があります。

デフォルト: デフォルトなし

制限: 完全修飾 URL は*使用しないでください*。相対 URI を使用する必要があります。

例: /Web pages/logoff.html

完全ログオフの設定方法

1. ユーザのログオフ用のカスタム HTTP アプリケーションを作成します。たとえば、ユーザを指定した URL にリダイレクトするための終了ボタンまたはサインオフ ボタンを追加します。
2. HTML ログオフ ページが、ブラウザのキャッシュからではなく、確実に Web サーバからロードされるようにするには、ログオフ ページをブラウザ内にキャッシュできないように設定します。たとえば、HTML ページの場合は、ページに次のようなメタタグを追加します。

```
<META HTTP-EQUIV="Pragma" CONTENT="no-cache">
<META HTTP-EQUIV="Expires" CONTENT="-1">
```

重要: メタ タグは一部の Web ブラウザで機能しない場合があります。その場合、Cache-Control HTTP ヘッダを使用します。

3. 以下の手順で LogOffURI パラメータを設定します。
   a. 必要に応じて、ポンド記号(#)を削除します。

b. ユーザをログオフするカスタム HTTP ファイルの URI を入力します。URL の絶対パスは入力しないでください。

完全ログオフが設定されます。

詳細情報:

エージェント cookie の cookie パスの指定 (P. 123)

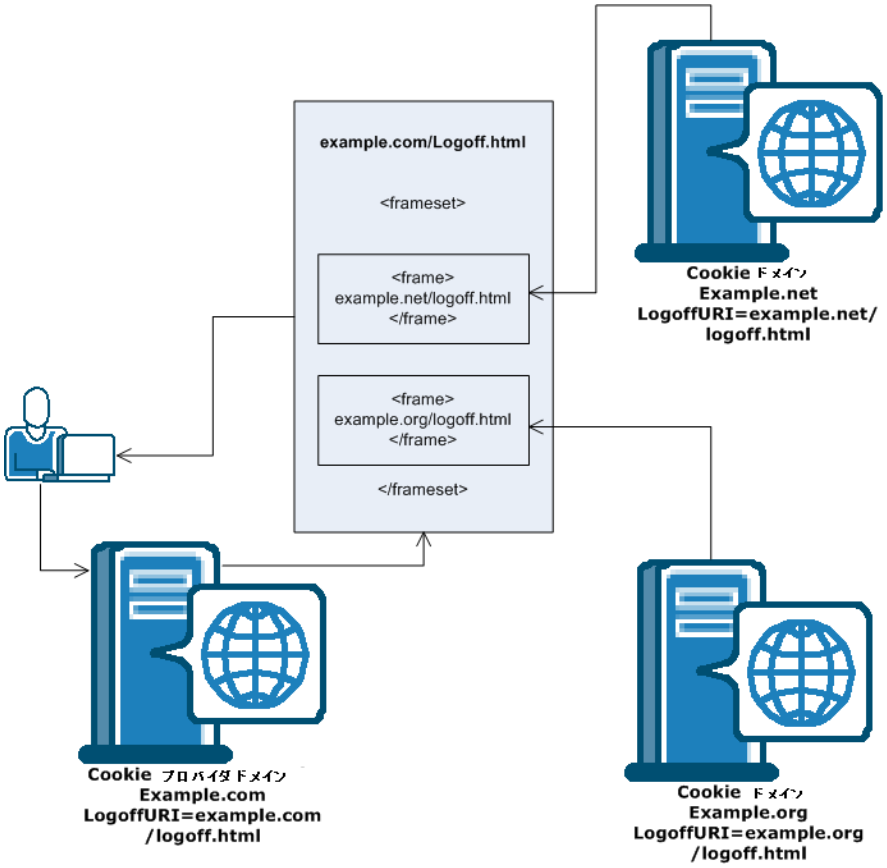
シングル サインオンでの完全ログオフの設定方法

シングル サインオン環境では、セッション cookie はローカル cookie ドメインと、Web エージェントに関連付けられた cookie プロバイダ ドメインからのみ削除されます。複数の cookie ドメインにわたるシングル サインオンの場合、SiteMinder の完全ログオフ機能では、ユーザが訪問したすべての cookie ドメインにおいてユーザを自動的にログオフすることはしません。

複数の cookie ドメインにわたるログオフを設定するには、以下の手順を使用します。

1. SSO 環境内の他の cookie ドメイン用に別のフレーム(または iframes)を含む一元化されたログオフ ページを作成します。これらのフレームは、1x1 ピクセルのような小さいものにできます。
2. 手順 1 で作成されたログオフ ページの各フレームについて、関連する cookie ドメインのログオフ URI へのハイパーリンクを追加します。たとえば、ほかに 2 つの cookie ドメイン(example.org と example.net)がある場合、以下の手順に従います。
   - 1 つのフレームに、example.org のログオフ URI へのハイパーリンクを追加します。
   - 別のフレームに、example.net のログオフ URI へのハイパーリンクを追加します。
3. cookie プロバイダ ドメインのログオフ URI が一元化されたログオフ ページを参照するように設定します。Web サーバがこのログオフ ページをロードすると、一元化されたログオフ ページのフレームが他の cookie ドメインから各ログオフ ページを呼び出します。ユーザがすべての cookie ドメインから一度にログオフされます。

以下の図は、一元化されたログオフ ページを使用する例にを示しています。



注: ハイパーリンクは、<frame> タグではなく <iframe> タグ内に配置することもできます。

IIS 6.0 エージェントと SharePoint Portal Server 2003 の統合

IIS 6.0 Web エージェントは Microsoft's SharePoint Portal Server 2003 と連携して、そのサーバに保管されているリソースに対するシングル サインオンを実現できます。

注: SharePoint セキュリティのコンテキストでは、ユーザ ストアは Active Directory (AD) である必要があります。

Web エージェントと SharePoint サーバを統合するには、以下の手順に従います。

1. 以下の前提条件を両方とも満たします。
   - r12.0 SP3 (またはそれ以降) の Web Agent for IIS 6.0 をインストールします。
   - Microsoft サポート オンライン「文書番号 824330: FIX IIS 6.0 が返さない AUTH_TYPE の統合セキュリティ」に記載されているホットフィックスを適用します。
2. デフォルトの仮想サーバの web.config ファイルに以下を追加します。このファイルは、たとえば、c:¥inetpub¥wwwroot2¥web.config にあります。
3. </system.web> 属性と </configuration> 属性の間に以下のエントリを挿入します。

   ```
   <appSettings>
   <add key="SPS-EnforceIISAnonymousSetting" value="false"/>
   </appSettings>
   ```

4. Web エージェントが SharePoint Portal Server リソースに対してユーザ セキュリティ コンテキストを提供できるように、セッション サーバを有効にします。

   注: 詳細については、ポリシー サーバのマニュアルを参照してください。

   Web エージェントが SharePoint Portal Server 2003 と統合されます。

エージェント cookie の cookie パスの指定

Web エージェントが cookie を作成する場合、Web エージェントは自動的に cookie パスとしてルート(/)ディレクトリを使用します。cookie のドメインとパスの属性は要求の URL と比較されます。cookie がドメインとパスに対して有効な場合、クライアントはサーバに cookie を送ります。cookie パスがルート値を使用する場合、クライアントはドメインのすべての要求を備えたサーバに cookie を送ります。

指定された一連のパスに SiteMinder cookie を設定すると、保護されていないリソースに cookie が送信されるときに生じる Web トラフィックを除去できます。たとえば、cookie パスを /mypackage に設定すると、クライアントはドメインの特定のパッケージ内の要求に対してのみ、cookie を送ります。

エージェント cookie の cookie パスを指定する方法

1. エージェント設定オブジェクトまたはローカル エージェント設定ファイルを開きます。
2. 以下のパラメータ内で cookie プロバイダの cookie パスを設定します。

   MasterCookiePath

   cookie プロバイダによって作成されたプライマリドメイン セッション cookie のパスを指定します。たとえば、このパラメータが /siteminderagent に設定されている場合、cookie プロバイダが作成するすべてのセッション cookie のパスに /siteminderagent が含まれます。cookie プロバイダ エージェントにこのパラメータが設定されていない場合は、デフォルト値が使用されます。

   デフォルト: /(ルート)
3. 以下のパラメータ内でセカンダリ エージェントの cookie パスを設定します。

   CookiePath

   以下のセカンダリ エージェント ブラウザ cookie の cookie パスを指定します。

   - xxSESSION
   - xxIDENTITY
   - xxDOMINODATA
   - xxCHALLENGE(SSL_CHALLENGE_DONE を含む)
   - xxDATA
   - xxSAVEDSESSION

たとえば、このパラメータを /BasicAuth に設定すると、前述のリスト内のセカンダリ エージェントはすべて、パスとして /BasicAuth を使用して作成されます。指定しない場合は、デフォルト値が使用されます。

4.x のエージェントとの後方互換性を維持するため、CookiePath は認証情報 cookie（xxxxCRED など）には追加*されません*。

以下の cookie は*常に*ルート パス（/）を使用します。

- ONDENIEDREDIR
- TRYNO

CookiePathScope パラメータが 0 より大きい場合は、CookiePath パラメータの設定が上書きされます。

デフォルト: /（ルート）

4. （オプション）CookiePath 値を使用する代わりに Web エージェントに URL から cookie パスを抽出させたい場合、以下のパラメータを 0 を超える数に設定します。

   CookiePathScope

   以下のセカンダリ エージェント cookie の cookie パスの範囲を指定します。

   - xxSESSION
   - xxIDENTITY
   - xxDOMINODATA
   - xxCHALLENGE（SSL_CHALLENGE_DONE を含む）
   - xxDATA
   - xxSAVEDSESSION

   CookiePathScope が 0 より大きい場合は、CookiePath パラメータの設定が上書きされます。

   デフォルト: 0

CookiePathScope 設定の機能

以下の表は、CookiePathScope パラメータの値が以下の設定でどのように機能するかを示しています。

- http://*fqdn*/path1/path2/path3/path4/index.html のような URL
- /BasicA の CookiePath パラメータ値

CookiePath 値	CookiePathScope 値	使用されるパス
/BasicA	0	/BasicA
/BasicA	1	/Path1
/BasicA	2	/Path1/Path2
/BasicA	3	/Path1/Path2/Path3
/BasicA	4	/Path1/Path2/Path3/Path4
/BasicA	5	/Path1/Path2/Path3/Path4
/BasicA	99	/Path1/Path2/Path3/Path4
/または「未定義」	0	/
/または「未定義」	1	/Path1
/または「未定義」	2	/Path1/Path2
/または「未定義」	3	/Path1/Path2/Path3
/または「未定義」	4	/Path1/Path2/Path3/Path4
/または「未定義」	5	/Path1/Path2/Path3/Path4
/または「未定義」	99	/Path1/Path2/Path3/Path4

これらの設定はさらに単純な SSO にも影響します。たとえば、CookiePathScope の値が 1 以上に設定されている場合、パスが/BasicA のセッション cookie が /BasicB/Index.html リクエストで有効にならないので、ユーザは /BasicA/Index.html と/BasicB/Index.html の両方の認証情報を要求されます。

第 10 章: Web アプリケーションの保護

このセクションには、以下のトピックが含まれています。

Web アプリケーション開発用メカニズム

SiteMinder には、Web アプリケーション開発用の以下のメカニズムが用意されています。

- 設定可能なレスポンス属性

  レスポンス属性は、ポリシー サーバによって Web エージェントに送信され、HTTP ヘッダまたは cookie に格納されます。管理者は、管理 UI を使用してこれらの属性を設定できます。これらの属性は、Web エージェントで使用できます。

- デフォルトの HTTP ヘッダ

  SiteMinder には、エージェントから Web アプリケーションに渡されるデフォルトの HTTP ヘッダ セットがあります。

- カスタム エラー処理

  SiteMinder では、標準の HTTP エラー内の HTML テキストをカスタマイズするか、カスタム エラー ページを使用して、アプリケーションに関するエラー情報を作成できます。

Web エージェントでのレスポンス属性の機能

SiteMinder のレスポンス属性は、アプリケーションに対して、ユーザ データの収集方法や、その情報を適用してユーザごとにパーソナライズしたコンテンツを表示する方法を指示します。

SiteMinder は、設定可能なレスポンス属性を用意しています。これらの属性は、アプリケーションにデータを渡し、ユーザの操作をカスタマイズするための手段です。

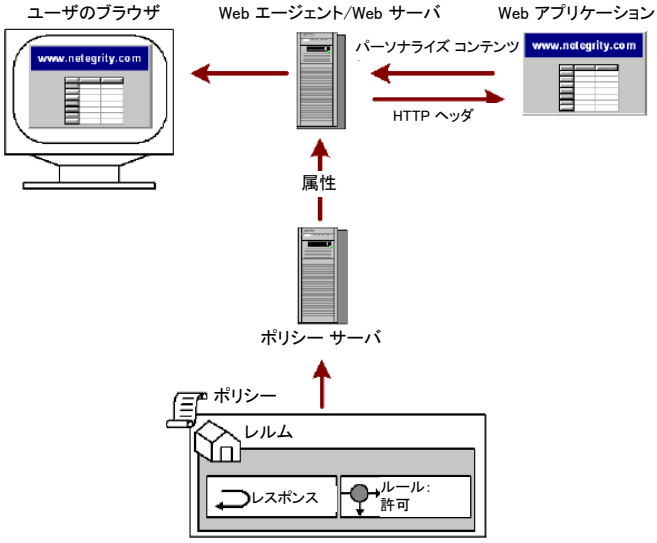
管理 UI を使用してレスポンスを設定し、ポリシー内の特定のルールにそのレスポンスを関連付けます。リクエストが、設定済みのレスポンスに関連付けられているルールをトリガした場合、ポリシー サーバはそのレスポンス データをエージェントに送信し、エージェントはその情報を解釈し、Web アプリケーションが利用できるようにします。

レスポンスを設定した後、そのレスポンスをエージェントのアクションに関連付けます。HTTP ヘッダと cookie のレスポンス属性を、GET と POST の各アクションに関連付けることができます。これらの属性を、認証イベントまたは許可イベントに結び付けることもできます。ユーザがそれらのイベントのどちらかを受け付けまたは拒否した場合に、ポリシー サーバは 1 つのレスポンスを送信することができます。

注: レスポンス属性を設定する場合、Web サーバがエージェントのレスポンスに使用できる最大バッファ サイズは 32 KB であることに注意してください。レスポンスについて、バッファ サイズ以外の制限事項はありません。

ヘッダ属性や cookie 属性以外のレスポンス属性を使用できるのは、認証イベントまたは許可イベントが発生した場合 *のみ*です。それぞれのイベントでユーザが受け入れられた場合でも、拒否された場合でもかまいません。たとえば、1 つのルールに対応させる 1 つの許可**イベント** アクションを選択し、次に 1 つの **WebAgent-OnReject-Redirect** レスポンス属性を設定することができます。許可プロセスにおいてユーザが SiteMinder によって拒否された場合、エージェントはそのユーザを他のページへリダイレクトして、そのページにそのユーザが拒否された理由を示すメッセージを表示することができます。

以下の図は、レスポンス属性がどのようにポリシー サーバから Web サーバに送信されるかを示したものです。



レスポンス属性のメンテナンスを簡素化するには、イベントの種類ごとに別々のレスポンス属性を定義します。たとえば、OnAccept イベントに対して 1 つのレスポンス属性を定義し、OnReject イベントに対して別のレスポンス属性を定義します。レスポンス属性を個別に定義することで、属性値の変更が必要なときの属性検索が容易になります。

HTTP ヘッダと cookie 変数

WebAgent-HTTP-Header-Variable および WebAgent-HTTP-Cookie-Variable 属性を使用すると、Web エージェントは、名前/値ペアのスタティック リストまたはダイナミック リストをアプリケーションに渡すことができます。名前/値ペアはリソースを要求するユーザに固有であるため、アプリケーションはユーザが参照するコンテンツをカスタマイズできます。

たとえば、管理者が WebAgent-HTTP-Header-Variable レスポンス属性にユーザのフルネームを格納するように設定するとします。ユーザが保護対象リソースへのアクセスを許可されると、Web エージェントはユーザのフルネームを Web アプリケーションに渡します。すると、アプリケーションによってユーザの名前が表示されます。これは、顧客との関係構築に役立ちます。

Web アプリケーション環境で、HTTP-Header-Variable レスポンス属性は HTTP_*attribute_name* 変数として表示されます。ここで、*attribute_name* は、HTTP 変数の名前（たとえば、USERFULLNAME）です。名前の一部としてアンダースコア（_）を使用する必要はありません。アンダースコアは、一部のアプリケーション サーバで問題を引き起こすからです。

注: 属性名の一部に使用されているダッシュ（-）をアンダースコア（_）に変換すること、およびすべてのアルファベットが、サーバによって大文字に変換されることがあります。

レスポンス属性のキャッシュ

レスポンス属性をキャッシュに格納するか、またはダイナミック データを含む属性を期限切れにして、強制的にポリシー サーバに問い合わせて情報を更新するように、SiteMinder エージェントに指示することができます。スタティック レスポンス属性を設定すると、ポリシー サーバにより、値のキャッシングが認められます。スタティック値は不変であるため、再計算の必要はありません。ユーザ属性、DN 属性、またはアクティブ属性を設定した場合は、その値をキャッシュするか、データが最新であるようにするために一定の間隔で再計算させるか、いずれかを選択することができます。

認証されたユーザ名をアプリケーションに渡す方法

REMOTE_USER CGI 環境変数は、Web サーバによって認証されたユーザの名前を保持します。Web エージェントが Web サーバにインストールされると、SiteMinder では Web サーバのネイティブ認証を置換します。そのため、REMOTE_USER は通常空白です。

REMOTE_USER 変数を必要とするアプリケーションがある場合、必ず REMOTE_USER 変数に値が設定されるようにする必要があります。

ただし、REMOTE_USER に値を設定できないか、それを必要としない Web サーバについては、Web エージェントのデフォルト ヘッダ HTTP_SM_USER がユーザ名をアプリケーションに渡す方法として使用されます。

詳細情報

REMOTE_USER 変数を設定するように Web エージェントを設定する (P. 134)

IIS Web エージェントで REMOTE_USER 変数を取り込む方法

IIS Web サーバが REMOTE_USER ヘッダを取り込むには、Web サーバで基本認証を有効にする必要があります。基本認証は、IIS 管理コンソール設定の[ディレクトリ セキュリティ]で設定されます。

基本認証が有効になっている場合、ユーザが SiteMinder で保護されたリソースを要求すると、Web エージェントは、パスワードを指定せずにユーザ名を指定することによって、IIS Web サーバの HTTP_Authorization ヘッダを設定しようとします。HTTP_Authorization ヘッダがあることは、IIS サーバの基本認証がその他すべての認証要求より優先されることを意味します。そのため、IIS Web サーバは、ユーザが独自の認証要求に応答しようとしていると見なします。SiteMinder Web エージェントが要求のユーザ コンテキストを設定する場合など、ISAPI フィルタが設定されていない限り、IIS Web サーバは不完全な HTTP_Authorization ヘッダによって渡されたユーザ名を認証しようとします。

Web エージェントは ISAPI フィルタとして動作するため、要求のユーザ コンテキストを設定して、REMOTE_USER ヘッダの値を指定できます。エージェントは、Yes に設定されている SetRemoteUser パラメータおよび 1 つ以上の以下の Web エージェント パラメータの構成に基づいて、REMOTE_USER ヘッダを取り込みます。

- DefaultUsername および DefaultPassword - これら両方のパラメータにより、Web エージェントが大部分のアクティビティで使用する(特権)プロキシ ユーザ アカウントが制御されます。
- ForceIISProxyUser - 通常の動作を無効にして、Web エージェントに、IIS Web サーバをプロキシ サーバとして実行するように強制します。
- UseAnonAccess - Web エージェントに対して、要求のユーザ コンテキストをまったく提供せずに、既存のユーザ コンテキストを未変更のままにするように指示します。
- 認証済みユーザ セキュリティ コンテキストで実行 - Web エージェントは、IIS Web サーバに対して、永続的なセッションに格納された認証情報を使用するように指示するため、通常の動作を無効にします。

SetRemoteUser パラメータおよび UseAnonAccess パラメータを一緒に使用する場合は、注意してください。

以下の表に、これらのパラメータがどのようにして連携するかを示します。

設定	動作
SetRemoteUser=yes および UseAnonAccess=yes	Web エージェントはユーザ セキュリティ コンテキストを渡さないため、REMOTE_USER 変数は設定できません。 ユーザ セキュリティ コンテキストがない場合は、IIS Web サーバは、エージェントが変更した HTTP_Authorization ヘッダの認証情報を使用する必要がありますが、この認証情報に含まれているのはユーザ名のみであるため、不完全です。
SetRemoteUser=yes および UseAnonAccess=no	Web エージェントは、DefaultUserName、DefaultPassword、または ForceIISProxyUser などのその他のパラメータの設定方法に応じて、一部のタイプのユーザ コンテキストを渡すことができます。 Web エージェントがセキュリティ コンテキストを渡さない場合、IIS Web サーバは、Web エージェントによって提供された認証情報を優先して不完全な HTTP_Authorization ヘッダを無視します。

REMOTE_USER 変数を設定するように Web エージェントを設定する

REMOTE_USER 変数を設定するように Web エージェントを設定する

- REMOTE_USER を SiteMinder のログイン ユーザ名の値に設定するには、Web エージェントの SetRemoteUser パラメータを yes に設定します。

  このパラメータのデフォルト値は no で、これは REMOTE_USER 変数を空白のままにするものです。

  注: SiteMinder Web エージェント 5.x QMR 2 以前では、SetRemoteUser パラメータが影響するのは IIS Web サーバのみでした。Apache および Oracle iPlanet エージェントでは、REMOTE_USER は必ず SiteMinder のログイン ユーザ名に設定されます。5.x QMR 2 より前のエージェントをインストールするユーザ、またはそのようなエージェントからのアップグレードを行うユーザは、REMOTE_USER がデフォルトで有効にならないことに注意してください。

- REMOTE_USER 変数を、ユーザのログイン認証情報ではなく、特定のユーザアカウントに基づいて設定するには、以下の手順に従います。
  - SetRemoteUser パラメータを yes に設定して、これを有効にします。
  - RemoteUserVar パラメータを設定します。このパラメータは、エージェントに対し、HTTP-WebAgent-Header-Variable レスポンス属性の値に基づいて 変数の値を設定するように指示するものです。このパラメータは、レガシー アプリケーションと統合するために使用します。

    RemoteUserVar パラメータを設定するには、レスポンス変数の名前のみを入力します。たとえば、「user=ajohnson」のような HTTP-WebAgent-Header-Variable を返すには、RemoteUserVar パラメータを値「user」に設定します。
  - ヘッダ変数を OnAuthAccept ルールにバインドします。既存の HTTP ヘッダ変数レスポンスを使用せずに、新規作成してください。

    注: 詳細については、ポリシー サーバドキュメントを参照してください。
- デフォルトに戻して、REMOTE_USER を空白のままにするには、SetRemoteUser パラメータを no に戻します。

注: SetRemoteUser または RemoteUserVar を設定する前に、セキュリティ上の因果関係を必ず考慮してください。

SiteMinder のデフォルトの HTTP ヘッダ

SiteMinder のデフォルトの HTTP ヘッダは、アプリケーションに対して、ユーザデータの収集方法や、その情報を適用してユーザごとにパーソナライズしたコンテンツを表示する方法を指示します。

Web アプリケーション環境の一部として、SiteMinder エージェントは、デフォルトの HTTP ヘッダを Web サーバに送信します。Web サーバは、そのヘッダ情報を Web アプリケーションが使用できるようにします。これらのヘッダを使用して関数を組み込んで、Web アプリケーションのコンテンツのパーソナライゼーションを行うことができます。ヘッダには、ユーザ名やユーザが実行を許可されているアクションのタイプなどの情報を格納することができます。

エージェントは、ヘッダが Web アプリケーションから呼び出されたかどうかにかかわらず、それらのヘッダを送信します（無条件に）。しかし、いくつかのヘッダを無効にして、ヘッダのスペースを空けることもできます。

Web エージェントが使用できる SiteMinder のデフォルトの HTTP ヘッダは、以下のとおりです。

HTTP_SM_AUTHDIRNAME

ポリシー サーバがユーザを認証する対象となるディレクトリの名前を示します。管理者は、管理 UI を使用してこのディレクトリを指定します。

HTTP_SM_AUTHDIRNAMESPACE

ポリシー サーバがユーザを認証する対象となるディレクトリ ネームスペースを指定します。管理者は、管理 UI を使用してこのネームスペースを指定します。

HTTP_SM_AUTHDIROID

ポリシー サーバ データベースのディレクトリ オブジェクト識別子（OID）を示します。

HTTP_SM_AUTHDIRSERVER

ポリシー サーバがユーザを認証する対象となるディレクトリ サーバを示します。管理者は、管理 UI を使用してこのディレクトリ サーバを指定します。

HTTP_SM_AUTHREASON

認証が失敗した後または 2 回目の認証要求の後に Web エージェントがユーザに返すコードを示します。

HTTP_SM_AUTHTYPE

ポリシー サーバがユーザの ID の確認に使用する認証方式のタイプを示します。

HTTP_SM_DOMINOCN

Domino LDAP ディレクトリを使用してユーザを認証する場合の、ユーザの Domino 正規名を指定します。

例: HTTP_SM_DOMINOCN="CN=jsmith/O=netegrity"

HTTP_SM_REALM

リソースが存在する SiteMinder のレルムを示します。

HTTP_SM_REALMOID

リソースが存在するレルムを識別するレルム オブジェクトの ID を示します。この ID は、サード パーティ製のアプリケーションでポリシー サーバを呼び出す場合に使用します。

HTTP_SM_SDOMAIN

エージェントのローカル cookie ドメインを示します。

HTTP_SM_SERVERIDENTITYSPEC

ポリシー サーバの識別チケットを示します。このチケットがユーザ ID を追跡します。Web エージェントは、ユーザに合わせてコンテンツをパーソナライズできるように、このチケットを使用して匿名認証方式で保護されたコンテンツにアクセスします。

HTTP_SM_SERVERSESSIONID

ユーザ セッションを識別する一意の文字列を示します。

HTTP_SM_SERVERSESSIONSPEC

ユーザ セッション情報を含むチケットを示します。この情報をデコードできるのはポリシー サーバのみです。

HTTP_SM_SESSIONDRIFT

Web エージェントがキャッシュ内の情報を使用して、セッションをアクティブにしておくことのできる時間の長さを示します。この時間を過ぎると、ポリシー サーバとのセッションが検証されます。このヘッダを設定する場合、ポリシー サーバ上のセッション サーバを有効にしておくこと、およびセッション検証期間を設定しておくことが必要です。

HTTP_SM_TIMETOEXPIRE

SiteMinder セッションの残り時間を示します。

HTTP_SM_TRANSACTIONID

各ユーザ リクエストに対してエージェントが生成した一意の ID を示します。

HTTP_SM_UNIVERSALID

ポリシー サーバが生成したユニバーサル ユーザ ID を指定します。この ID は顧客に固有であり、アプリケーションに対してユーザを識別しますが、ユーザ ログインとは異なります。

HTTP_SM_USER

認証されたユーザのログイン名を示します。証明書ベースの認証などで、ユーザがログイン時にユーザ名を入力しなかった場合、この変数は設定されません。

HTTP_SM_USERDN

ポリシー サーバによって判別される、認証済みユーザの識別名を指定します。

匿名認証方式では、このヘッダによって GUID（グローバルな固有識別子）が返されます。

HTTP_SM_USERMSG

認証の試行後にエージェントがユーザに提示するテキストを指定します。認証方式によっては、認証要求時のテキストや認証に失敗した理由が表示されます。

デフォルトの HTTP ヘッダ変数の無効化

システムプラットフォームの多くは、HTTP ヘッダの限界値が 4096 バイトになっています。この限界値を超えないようにしてカスタム レスポンス変数用のスペースを確保する場合、SiteMinder の一部のデフォルト HTTP ヘッダ変数を無効にすることができます。

デフォルトの変数は以下のカテゴリにグループ化されています。

注: 個々の変数を無効にすることはできません。いくつかの変数のカテゴリのみ無効にできます。

- 認証ソース変数
  - SM_AUTHDIRNAME
  - SM_AUTHDIRSERVER
  - SM_AUTHDIRNAMESPACE
  - SM_AUTHDIROID
- ユーザ セッション変数
  - SM_SERVERSESSIONID
  - SM_SERVERSESSIONSPEC
  - SM_SERVERIDENTITYSPEC
  - SM_SESSIONDRIFT
  - SM_TIMETOEXPIRE
- ユーザ名変数
  - SM_USER
  - SM_USERDN
  - SM_DOMINOCN

HTTP ヘッダ変数はデフォルトで使用されていますが、それらを無効にするには、以下のいずれかのタスクを実行します。

- 認証ソース変数を無効にするには、DisableAuthSrcVars パラメータの値を yes に設定します。
- ユーザ セッション変数を無効にするには、DisableSessionVars パラメータの値を yes に設定します。
- ユーザ名変数を無効にするには、DisableUserNameVars パラメータの値を yes に設定します。

注: ユーザが Identity Manager、またはこのカテゴリの変数を使用する可能性のあるアプリケーションを使用している場合は、このパラメータの値を no（有効）に設定するようにしてください。

SiteMinder のデフォルトの HTTP ヘッダを使用するアプリケーションの例

ここでは、SiteMinder 環境ヘッダを呼び出す各種アプリケーションの例を紹介します。アプリケーションにおける、デフォルト ヘッダの使用方法についても紹介します。

HTTP ヘッダの抽出（シェル スクリプトを使用）

デフォルトの HTTP ヘッダを参照するため、環境変数を抽出して表示します。この例では TEST.SH というシェル スクリプトを使用してブラウザに変数を返します。

```
#!/bin/sh
# First put out a valid header
echo Content-Type: text/html
echo
# Then echo out the variables
echo HTTP_SM_USER: $HTTP_SM_USER
echo HTTP_SM_AUTHTYPE: $HTTP_SM_AUTHTYPE
echo HTTP_SM_AUTHDIRNAME: $HTTP_SM_AUTHDIRNAME
echo HTTP_SM_SERVERSESSIONID: $HTTP_SM_SERVERSESSIONID
echo HTTP_SM_SESSIONID: $HTTP_SM_SESSIONID
```

HTTP ヘッダの抽出（NSAPI を使用）

この例では、Oracle iPlanet API（NSAPI）を呼び出してヘッダを取得するアプリケーションを示します。名前/値ペアは、要求の構造のヘッダ ブロックにあります。

ヘッダへのアクセスは、次のように行います。

```
char * user;
user = pblock_findval("HTTP_SM_USER", rq->headers);
```

HTTP ヘッダの抽出（PERL を使用）

PERL（Practical Extraction and Report Language）から環境変数にアクセスする操作は、一般的に行われます。以下のスクリプトは、ブラウザに変数を返すシェルスクリプト TEST.SH の PERL バージョンです。

```
#!/usr/win32/perl117
# Example Test Program To Echo Back Default
# Web Agent Headers from a CGI environment
#
# First put out a valid CGI Header
print “Content-Type: text/html¥n¥n”;
# Now print out the standard HTML document tags
print “<HTML>”;
<HEAD><TITLE> Test Web Agent Headers </TITLE></HEAD>
print “<BODY BGCOLOR=#ffffff>”;
print “<H1> Test Web Agent Headers </H1>”;
print “HTTP_SM_USER: $ENV{‘HTTP_SM_USER‘} <BR>”;
print “HTTP_SM_AUTHTYPE: $ENV{‘HTTP_SM_AUTHTYPE‘} <BR>”;
print “HTTP_SM_SESSIONID: $ENV{‘HTTP_SM_SESSIONID‘} <BR>”;
# Now end the HTML file output
print “</BODY>”;
print “</HTML>”;
```

以下の PERL スクリプトは、SiteMinder 変数だけでなく、すべての環境変数をブラウザに返します。

```
#!/export/home/iplanet/server4/install/perl
print "content-type: text/html¥n¥n";

print "<HTML>¥n";

print "<HEAD>¥n";
print "<TITLE>echo cgi env. vars.</TITLE>¥n";
print "<H2>Echo CGI Environment Variables</H2>¥n";
print "</HEAD>¥n";
print "<BODY>¥n";
print "<HR>¥n";
print "<H3>Environment Variables</H3>¥n";
print "<UL>¥n";
foreach $key (keys %ENV) {
print "<LI>$key = $ENV{$key}¥n";
}
print "</UL>¥n";
print "</BODY>¥n";
```

HTTP ヘッダの抽出（ASP を使用）

デフォルトの HTTP ヘッダを参照するため、ASP スクリプトを使用して環境変数を抽出して表示します。

SiteMinder 変数を返すために使用する ASP スクリプトには、すべての環境変数を取得する ALL_HTTP ヘッダと、SiteMinder 変数を解析するコードが含まれている必要があります。例として、次のスクリプトを使用してください。

```
<HTML>
<HEAD><TITLE> Test Web Agent Headers </TITLE></HEAD>
BODY BGCOLOR=#ffffff
TABLE BORDER=1
<TR><TD VALIGN=TOP><B>Variable</B></TD><TD VALIGN=TOP><B>Value</B></TD></TR>
% For Each key in Request.ServerVariables %
<TR>
<TD><% = key %></TD>
<TD>
<%
if Request.ServerVariables(key) = "" Then
if GetAttribute(key) = "" Then
Response.Write " " ' To force border around table cell
else
Response.Write GetAttribute(key)
end if
else
Response.Write Request.ServerVariables(key)
end if
Response.Write "</TD>"
%>
</TR>
<% Next %>
</TABLE>

<% Function GetAttribute(AttrName)
                Dim AllAttrs
                Dim RealAttrName
                Dim Location
                Dim Result

                AllAttrs = Request.ServerVariables("ALL_HTTP")
                RealAttrName = AttrName

                Location = instr(AllAttrs, RealAttrName & ":")

                if Location <= 0 then
                GetAttribute = ""
                Exit Function
                end if
```

```
                    Result = mid(AllAttrs, Location + Len(RealAttrName) + 1)

                    Location = instr(Result, chr(10))
                    if Location <= 0 then Location = len(Result) + 1

                    GetAttribute = left(Result, Location - 1)
End Function %>
```

ヘッダ変数とエンド ユーザ IP アドレス検証

SiteMinder Web エージェントは、最初のリクエストの後に同じユーザからリクエストを受け取ると、リクエストを送信したユーザの IP アドレスをセッション cookie 内で暗号化されている IP アドレスと比較することによって、後続のリクエストと共に送信されたセッション cookie を検証します。cookie 内のアドレスは、ユーザの初期の要求中にエージェントによって生成されます。

ファイアウォール、ロード バランサ、キャッシュ デバイス、およびプロキシなど、着信ネットワークトラフィックを平準化して管理するために使用されるメカニズムは、ユーザの IP アドレスを変更したり、すべての受信要求が単一の IP アドレスまたは少数の IP アドレスのグループから発信されているかのように見せたりすることがあります。その結果、Web エージェントの IP チェックは無効になります。Web エージェントは、カスタム HTTP ヘッダおよび安全なプロキシ IP アドレスの設定可能なリストを使用して、このようなネットワーク環境で IP チェックを実行できるようになりました。

以下の表に、新しい IP チェック機能の用語をリストします。

用語	定義
HTTP 要求ヘッダ	HTTP 要求の単一の要素を説明する名前/値ペア。
カスタム IP ヘッダ	中間 HTTP ネットワーク アプリケーションまたはハードウェア デバイスがリクエスタの IP アドレスを格納するために使用する、ユーザ定義の HTTP 要求ヘッダ。
IP チェック	最初の要求後に、要求の REMOTE_ADDR を、SMSESSION cookie に格納されている REMOTE_ADDR 値と比較することによって、Web エージェントが要求の認証性をチェックできる機能。この機能は、IP 検証とも呼ばれます。

用語	定義
REMOTE_ADDR	Web サーバにリクエストを送る HTTP クライアントの IP アドレスを表す Web サーバ変数。REMOTE_IP または CLIENT_IP とも呼ばれます。これは、プロキシ サーバ、NAT ファイアウォール、またはその他のネットワーク サービスやデバイスが、リクエスタとターゲット Web サーバの間にある場合のリクエスタ IP アドレスとは異なります。
要求者	HTTP 要求の発信者。通常、ブラウザを使用しているユーザです。
リクエスタの IP アドレス	オリジナルの HTTP 要求を発信しているユーザの IP アドレス。
シングル サインオン	保護された Web サイトに安全にアクセスするためにユーザに求める認証情報入力を、セッション中に 1 回のみで済むようにする機能。
SMSESSION cookie	Web エージェントがシングル サインオン状態を追跡するために使用する HTTP メカニズム。

カスタム ヘッダによる IP アドレスの検証方法

Web エージェントは、REMOTE_ADDR 変数を使用する代わりにカスタム HTTP ヘッダを使用して、ユーザの IP アドレスを判別できるようになりました。プロキシまたはその他のデバイスがカスタム クライアント IP ヘッダを設定し、Web エージェントが受信要求でこのヘッダを検索するように設定されている場合は、エージェントは、クライアント IP の情報源としてこのヘッダを使用します。

カスタム ヘッダを設定するほかに、プロキシ IP アドレスのリストをセットアップすることもできます。 REMOTE_ADDR がプロキシ リストのアドレスと一致する場合は、Web エージェントは、カスタム ヘッダからユーザの IP アドレスを取得します。一致しない場合は、ユーザの IP アドレスは、REMOTE_ADDR から取得されます。

Web エージェントがリクエスタの IP アドレスを解決すると、アドレスは格納され、要求の処理のために使用されます。アドレスを解決できない場合は、IP アドレスは unknown に設定されます。

Web エージェントは、クライアント IP アドレスが解決された場所を記録して、必要なデバッグを容易に行えるようにします。

IP アドレス検証の設定

IP チェックは、以下の 2 つのパラメータを使用して実装されます。

- CustomIpHeader - それぞれのエントリは、Web エージェントがリクエスタの IP アドレスを検索する HTTP ヘッダを指定します。このパラメータに値が設定されていない場合は、デフォルトは空の文字列になります。最大長はないため、値は、有効な HTTP ヘッダ値を含む任意の文字列（たとえば、HTTP_ORIGINAL_IP）にできます。

  この設定は、中央（エージェントの設定オブジェクト内）またはローカル側（設定ファイル内）で定義できます。中央でのエージェント設定の場合は、このパラメータを複数値の設定として指定できます。ローカル設定の場合は、それぞれのエントリを別の行に追加する必要があります。以下に例を示します。

  CustomIpHeader="HTTP_ORIGINAL_IP"

  CustomIpHeader="HTTP_CLIENT_IP"

  注: CustomIpHeader パラメータは、プロキシ アドレス リストを定義せずに使用できますが、プロキシ定義リストを使用するには、少なくとも 1 つのカスタム IP ヘッダを定義する必要があります。

- ProxyDefinition - それぞれのエントリは、リクエスタ IP アドレスを解決するためにカスタム HTTP ヘッダを使用する必要があるプロキシ(キャッシュ デバイスなど)の IP アドレスを指定します。このパラメータに値が設定されていない場合は、デフォルトは空の文字列になります。

  このパラメータには最大長はないため、値は、有効な IP アドレスを含む任意の文字列(たとえば、111.123.23.11)にできます。サーバ名または完全修飾 DNS ホスト名を入力しないでください。

  この設定は、中央(エージェントの設定オブジェクト内)またはローカル側(設定ファイル内)で定義できます。中央でのエージェント設定の場合は、このパラメータを複数値の設定として指定できます。ローカル設定の場合は、それぞれのエントリを別の行に追加する必要があります。以下に例を示します。

  ```
  ProxyDefinition=111.12.1.1
  ProxyDefinition=112.11.2.2
  ```

  注: これらの設定は、TransientIPCheck および PersistentIPCheck パラメータとは無関係です。

注: 従来のすべての Web エージェントでは、これらのパラメータは WebAgent.conf ファイルにあります。フレームワーク エージェントの場合は、これらの値は LocalConfig.conf ファイルにあります。

以前のリリースの Web エージェントによる IP アドレス検証

6.x QMR 2 または 3 以前の Web エージェントの環境では、IP チェックが設定されている場合、シングル サインオンが影響を受ける可能性があります。

v6.x QMR 2 および 5.x QMR 7 より前の Web エージェントは、リクエスタ IP アドレスを解決できないため、これらの Web エージェントによって作成される SMSESSION cookie は、6.x QMR 2 または 3 の Web エージェントによって破棄されることがあります。これには、SDK を使用して SMSESSION cookie を生成するカスタム エージェント、アプリケーション サーバ エージェント、および SMSESSION cookie を使用するシングル サインオン環境内のその他のすべての SiteMinder エージェントが含まれます。

逆に言えば、6.x QMR 2 および 3 の Web エージェントは、リクエスタの IP アドレスを解決できるため、このアドレスは古いエージェントによって解決されたアドレスとは異なります。

HTTP ヘッダの保存

新しいヘッダが生成されたときに、既存の HTTP ヘッダを置き換えずに保存するように、Web エージェントを設定することができます。この機能は、名前は同じであっても値が異なる複数の SiteMinder のレスポンスを生成し、ヘッダへの格納が必要なアプリケーションにおいて有効です。同じ HTTP ヘッダのインスタンスが複数存在する場合、Web サーバはすべての適切なヘッダ値をカンマで区切って 1 つのヘッダを生成して処理します。

デフォルトでは、Web エージェントは、誤ったヘッダ値を使用するアプリケーションへの予防措置としてヘッダを保存しません。Oracle iPlanet、Domino および Apache Web エージェントで HTTP ヘッダを保存するには、PreserveHeaders パラメータを yes に設定します。デフォルト値は no です。

HTTP ヘッダ リソースのキャッシュ方法の制御

以下のパラメータの設定により、Web エージェントでキャッシュ関連のリクエストヘッダを処理する方法を制御できます。

AllowCacheHeaders

Web エージェントで、保護されたリソースに対するリクエストを Web サーバに渡す前に、リクエストから以下のキャッシュ関連 HTTP ヘッダを削除するか動かを指定します。

- if-modified-since
- if-none-match

この設定は、ブラウザでキャッシュされたページをするかどうかに影響しますが、自動許可されたリソース(IgnoreExt パラメータの値によって一致したリソースを含む)には影響しません。自動許可されるリソースのキャッシュは、Web サーバおよびブラウザの設定によって決定されます。

このパラメータには、以下の値を設定できます。

- Yes -- エージェントはキャッシュ関連の HTTP ヘッダを削除しません。セッションを検証するためには SMSESSION cookie が引き続き追跡されます。セッションが期限切れになると、Web エージェントは、キャッシュに格納されているリソースで、if-modified-since HTTP ヘッダに示される時間以降 Web サーバによって変更されていないリソースについて、更新された SMSESSION cookie を 304「変更なし」のレスポンスで送信します。

重要: このパラメータが yes に設定された場合、Web サーバ上のアプリケーションによってパーソナライズされたページで、適切なキャッシュ制御ヘッダを設定されていないページは、ブラウザまたは HTTP に一時的にキャッシュされる可能性があります。これは、予期しない動作を引き起こす可能性があり、ブラウザが機密データをディスクに保存することを可能にします。

- No -- エージェントは、保護されているリソースのリクエストからのみ、キャッシュ関連の HTTP ヘッダを削除します。Web サーバでは、リクエストを無条件として取り扱い、キャッシュ検証操作は実行されません。
- None -- Web エージェントは、保護されているリソースおよび保護されていないリソースについて、キャッシュ関連のヘッダをすべて削除します。

終了したセッションの場合、AllowCacheHeaders パラメータの値にかかわらず、ブラウザはキャッシュされたコンテンツを使用しません。

このパラメータの設定は以下のパラメータに影響します。

- LogOffURI -- LogOffURI パラメータも使用している場合、AllowCacheHeaders パラメータの値を no に設定することをお勧めします。そうしないと、キャッシュされたログオフ ページがユーザに提示され、そのセッションが適切に終了されません。

デフォルト: No

制限: Yes、No、None

保護されているまたはされていないリソースからキャッシュ関連ヘッダをすべて削除するには、AllowCacheHeaders パラメータの値を none に設定します。

注: HTTP 1.1 キャッシュの仕組みの詳細については、RFC 2616 でセクション 13「Caching in HTTP」を参照してください。

ヘッダでの小文字 HTTP の使用（Oracle iPlanet、Apache、Domino Web サーバ）

大文字と小文字を区別するサーバ アプリケーションが存在している場合、エージェントの HTTP ヘッダで大文字または小文字のいずれを使用するかを指定できます。Web エージェントのデフォルトは、小文字のヘッダです。

たとえば、Oracle iPlanet Web サーバの場合、http_sm_user のように、デフォルトで HTTP ヘッダ変数は小文字になります。

注: IIS Web エージェントは、この機能は利用できません。IIS は、すべてのヘッダを強制的に大文字にするためです。

小文字のヘッダを使用するには、LowerCaseHTTP パラメータを yes に設定します。大文字のヘッダ変数が必要な場合、LowerCaseHTTP を no に設定します。

詳細情報:

Oracle iPlanet Web サーバ ログのトランザクション ID の記録 (P. 179)

HTTP ヘッダのエンコード仕様の設定

HTTPHeaderEncodingSpec の設定は、すべての HTTP ヘッダの値、およびすべてのカスタム HTTP-COOKIE レスポンスそれぞれのエンコードに影響を及ぼします。

このパラメータを使用して、Web アプリケーションをサポートし、ローカライズ済みのテキストを特定のエンコードで表示することを期待できます。cookie は、ブラウザとポータルの間で HTTP プロトコルを介してやり取りされるので、HTTP トラフィックが無効と見なす文字が、選択したエンコードによって cookie の中に書き込まれる場合は、RFC-2047 の HTTPWrapSpec を使用する必要があります。

たとえば、RFC-2047 による追加のエンコードを実施しない場合、一部の Shift-JIS 文字は望ましくない結果を招きます。

漢字文字の場合は、SHIFT-JIS のスーパーセットである SECP932 を使用できます。ほとんどの漢字エンコードおよびデコードに SHIFT-JIS を使用できますが、CP932 ではそれより多くの文字セットがカバーされます。

HTTPWrapSpec を使用する場合、データは最初に HTTPHeaderEncodingSpec に従ってエンコードされ、ついで RFC-2047 の仕様に従って追加エンコードされます。

このパラメータの構文は、次のとおりです。

encoding_spec, wrapping_spec

encoding_spec は、UTF-8、Shift-JIS、EUC-J、または ISO-2022 JP のいずれかのエンコード タイプを表すテキスト文字列です。エージェントに使用させたいエンコードタイプを指定します。

wrapping_spec は、ラッピング仕様ですが、RFC-2047 にする必要があります。この変数はオプションですが、ラッピング仕様を記述することを強くお勧めします。ここで選択するエンコードタイプは、HTTP プロトコルと互換性のないバイトコードを生成する可能性があるからです。

2 バイトのエンコード済みデータを含むカスタム HTTP cookie レスポンスを使用している場合、このことが特に当てはまります。たとえば、RFC-2047 で追加エンコードしない場合、一部の Shift-JIS 文字が適切に表示されません。また、ラッピングは、受信アプリケーションがエンコードされたテキストをより適切に解釈できるように、アプリケーションにエンコードのタイプと性質を伝えます。たとえば、このパラメータを Shift-JIS,RFC-2047 に設定することもできます。

RFC-2047 を使用する場合、エージェントは最初に、選択されたエンコード仕様に基づいてデータをエンコードし、ついで RFC-2047 仕様に従ってそのデータを追加エンコードします。

注: **HTTPHeaderEncodingSpec** の設定を空白のままにした場合、デフォルトは UTF-8 であり、ラッピングは行われません。

RFC 2047 への準拠の無効化

デフォルトでは、Web エージェントは RFC 2047 に準拠しています。ただし、ConformToRFC2047 パラメータを no に設定することで、この準拠を無効にできます。

このパラメータが存在しないか、yes に設定されている場合は、Web エージェントは RFC 2047 に準拠しています。

フォームの認証要求に関する SM_AGENT_ATTR_USRMSG レスポンスの使用

SM_AGENTAPI_ATTR_USERMSG レスポンスを使用すると、カスタム SiteMinder 認証方式の開発者は、ユーザへの認証要求の一部、または他の目的で、カスタム テキストを自らのクライアント アプリケーションへ返すことができます。

v5 QMR3 およびそれ以降で、Web エージェントはフォームによる認証要求を行う際に、SM_AGENTAPI_ATTR_USERMSG レスポンスから得られたテキストを SMUSRMSG cookie へ変換できるようになりました。

認証要求が完了した後で SMUSRMSG cookie が削除されることを保証するために、FCC は以下のように、POST 要求が成功した後でその cookie を消費（ブラウザから削除）します。

- SiteMinder ネイティブモードでは、エージェントはログインに成功した後、リダイレクトを行って元のターゲット URL に戻している最中に、その cookie を削除します。
- SiteMinder 4.x 互換モードでは、エージェントは FORMCRED cookie を生成した後、リダイレクトを行ってターゲット URL に戻している最中に、その cookie を削除します。

注: SMUSRMSG cookie は一定の時間にわたってユーザのブラウザ内に保存されます。また、安全ではない HTTP 接続を介して伝送される可能性もあります。その結果、機密データを避ける必要があります。

Web エージェントは、フォームによる認証要求を行う際に、SMUSRMSG cookie の中に配置されたテキストを URL エンコード化します。その結果、それらのテキストは、HTTP 伝送を行う際に安全になりますし、半角スペースや他の有害な文字を除去できます。FCC は、環境からこのテキストを利用できるようになる前に、カスタム FCC 機能でこのテキストをデコードします。

注: URL エンコードは、テキストが SMUSRMSG cookie に置かれていない限り実装されません。

新しい機能を実装するには、カスタム認証方式の開発者は、カスタムフォームをベースとする認証方式を生成する必要があります。Sm_AgentApi_Login() の呼び出しが SM_AGENTAPI_CHALLENGE を返した場合、エージェントは、要求を行っているユーザを、Sm_AgentApi_IsProtected() へのレスポンスとして提供された認証方式 URL へリダイレクトすることにより、そのユーザに認証を要求します。

Web エージェントが、HTML フォーム認証方式テンプレートを使用する認証方式を取り扱う場合、そのエージェントはレスポンス属性 SM_AGENTAPI_ATTR_STATUS_MESSAGE を検索します。この属性が見つかった場合、エージェントは適切な SMUSRMSG cookie を生成し、同時に、認証方式 URL へのリダイレクトを行います。ついで、必要な .FCC ソースファイル内で適切なディレクティブが記述されている場合、FCC はフォームを生成する際に、この cookie を使用できます。

注: 詳細については、ポリシー サーバドキュメントを参照してください。

HTTP ヘッダのレガシー変数の有効化

次のパラメータを使用して、Web エージェントが HTTP ヘッダに関してどの命名規則を使用するかを指定できます。

LegacyVariables

Web エージェントが HTTP ヘッダ名でアンダースコアを使用するかどうかを指定します。一部の Web サーバ（Sun Java System など）では、HTTP ヘッダでアンダースコア文字を使用すると、一部のアプリケーションで問題が発生します。

このパラメータを no に設定すると、以下の例に示されるように、HTTP ヘッダでアンダースコアは使用されません。

SM*HeaderName*

このパラメータを yes に設定すると、以下の例に示されるように、HTTP ヘッダでアンダースコアが使用されます。

SM_*HeaderName*

デフォルト:（従来のエージェント）Yes

デフォルト:（フレームワーク エージェント）No

レガシー変数を有効にし、Web エージェントに HTTP ヘッダ名でアンダースコアを使用させるには、LegacyVariables パラメータの設定値を yes に設定します。

HTTPS ポートの定義

要求をより安全にしておくために Web サーバ（HTTPS）への SSL 接続を使用している場合は、以下のパラメータを使用して HTTPS ポートを指定する必要があります。

HttpsPorts

ユーザが Web サーバへの SSL 接続を使用しているかどうかを Web エージェントがリスンする安全なポートを指定します。このパラメータの値を指定する場合、安全な要求を提供するすべての Web サーバの対象となるすべてのポートを含める必要があります。値を指定しなかった場合、Web エージェントは HTTP スキームをサーバのコンテキストから読み取ります。

サーバが、（HTTPS を HTTP へ変換する）HTTPS アクセラレータの背後にある場合、ブラウザはその要求を SSL 接続として扱います。

デフォルト: 空白

例: 80

例:（複数のポート）80,8080,8083

HTTPS ポートを定義するには、HttpsPorts パラメータの値を SSL を使用するポート番号に設定します。ポート番号が複数ある場合は、カンマで区切ります。

Apache 2.x サーバ上での HttpsPorts パラメータの使用

Apache 2.x Web サーバで HttpsPorts を使用する(たとえば、Apache Web サーバで SSL アクセラレータを使用する場合など)には、Web サーバに追加の設定変更を加えます。SSL アクセラレータ、または HTTP_HOST ヘッダの値を変更するあらゆる中間デバイスを使用する場合は、これらの変更が必要です。

Apache 2.x サーバで HttpsPorts パラメータを使用する方法

1. Apache Web サーバの httpd.conf ファイルを開き、以下の変更を加えます。
   - UseCanonicalName パラメータの値を on に変更します。
   - ServerName パラメータの値を以下のように変更します。

     server_name:*port_number*
2. Web エージェントの以下の設定パラメータを変更します。
   - GetPortFromHeaders パラメータの値を yes に変更します。

Oracle iPlanet Web サーバでの複数の AuthTrans 関数の処理

AuthTrans 関数は、Oracle iPlanet Web サーバを初期化するためのディレクティブです。Oracle iPlanet Web サーバは、obj.conf ファイル内に指定された順番に従って複数の AuthTrans 関数を実行します。Oracle iPlanet サーバは、REQ_PROCEED コマンドが返されるまで、AuthTrans 関数を次々に呼び出します。いったん REQ_PROCEED コマンドが返されると、それ以降の AuthTrans 関数は実行されません。

デフォルトでは、SiteMinder が最初の AuthTrans 関数になり、REQ_PROCEED を返します。他の AuthTrans 関数が実行されるようにするには、EnableOtherAuthTrans パラメータを追加して値を yes に設定する必要があります。

このパラメータのデフォルト値は no です。複数の AuthTrans 関数を有効にするには、EnableOtherAuthTrans パラメータを yes に設定します。

このパラメータを追加することにより、SiteMinder Web エージェントが他の関数と共存できるようになります。

ただし、obj.conf ファイル内で、SiteMinder エージェントの関数を、AuthTrans ディレクティブの最初のエントリにしてください。そのエントリは、次のようになります。

```
AuthTrans fn="SiteMinderAgent"
```

カスタム エラー処理の指定

カスタムエラー処理を使用すると、エラー情報を各自のアプリケーションと関連付けることができます。アプリケーションをユーザに合わせてカスタマイズするには、HTTP 500、HTTP 401、および HTTP 403 のエラー ページに表示される HTML テキストを変更するか、HTTP 401 エラーを除き、カスタム エラー ページまたはアプリケーションを示す URL にユーザをリダイレクトします。

カスタム エラー処理で設定できるエラーには、以下の種類があります。

- サーバ エラー - エージェントは、HTTP 500 Web サーバ エラーが原因で表示されるエラー ページに、ServerErrorFile を使用します。これらのエラー コードは、カスタム エラー ページに対して渡されます。次のものが該当します。
  - Web エージェントが、必要な HTTP ヘッダの値を読み込むことができないために発生するエラー
  - 高度な認証用 cookie が解析されない、もしくはエラーのステータスを含めることができない場合のエラー
  - Web エージェントとポリシー サーバの接続エラー
- アクセス拒否エラー - エージェントは Custom401ErrorFile パラメータの中で指定されたファイルを使用して、標準的な 401 アクセス拒否メッセージを表す、カスタマイズ済みテキストを表示します。リソースにアクセスするための十分な権限がそのユーザにない場合、そのようなエラーが発生します。

  注: 一部の Web サーバは、カスタム テキストに独自のテキストを追加することがあります。したがって、それらのサーバ用のレスポンスページを完全にカスタマイズすることはできません。
- cookie 要求エラー - RequireCookies パラメータが設定済みの場合、Web エージェントは基本認証を実施する間に cookie を設定します。基本認証情報と一緒に、ブラウザからこの cookie が返されなかった場合は、ReqCookieErrorFile パラメータによって指定されたエラー ページが返されます。エージェントはそのユーザがその Web サーバにアクセスすることを拒否します。
- クロスサイト スクリプティング エラー - エージェントは、HTTP 403 クロスサイト スクリプティング エラーが原因で表示されるエラー ページに、CSSErrorFile パラメータの中で指定されたファイルを使用します。クロスサイト スクリプティングは、Web サイトのセキュリティを危うくします。

これらの HTML ファイルまたはアプリケーションを作成したら、Web エージェントに対してカスタム エラー ページまたは URL を指示します。

注: Apache サーバがプロキシ サーバまたはリバース プロキシ サーバとして使用されている場合、Apache エージェントは、カスタム SiteMinder エラー ページではなく、Apache HTTP 標準の 500 エラー ページおよび 403 エラー ページを返します。

カスタム エラー処理の設定

アプリケーションをユーザに合わせてカスタマイズするには、HTTP 500、HTTP 401、および HTTP 403 のエラー ページに表示される HTML テキストを追加するか、カスタム エラー ページまたはアプリケーションを示す URL にユーザをリダイレクトします。

HTTP 500 および 403 エラーのみ: ユーザを URL にリダイレクトするようにエージェントを設定すると、エージェントは、エラー コードを ?SMError=*error_code* という形式で URL に付加します。標準の HTML エラー テキストを追加する場合のみ、先頭と終端を示す Body タグ (<body> </body>) の間に HTML コードを指定できます。完全にカスタムの HTML ページを使用したり、Body タグの外側にあるテキストをカスタマイズしたりすることはできません。

Web エージェントにカスタム エラーページまたは URL を指定するには、テキスト ファイルへのパスを指定するか、エージェント設定ファイル内でその URL を入力します。

カスタム 401 ページに関する注意

- Custom401errorfile パラメータを URL に設定しないでください。
- Custom401errorfile に対応する何らかの値 (使用可能かどうかにかかわりなく) が存在している場合、エージェントは 60 秒ごとに、そのファイルが変更されたかどうかを調べます。しかし、このレスポンスは、性質上、スタティックであることが意図されています。たとえば、「*user_name* denied」というタイプの動的なメッセージを挿入することはできません。

  Custom401errorfile の値が存在する場合、その値の有効性にかかわりなく再チェックがトリガされるので、エージェントを再起動することなく、エラーを訂正できます。その訂正結果は、次のチェックの際に取り出されます。
- カスタマイズ済みのメッセージ ファイルのテキストが、他のエラーによって公開されることはありません。そのファイルのパス名は、起動時、およびエラー発生時にログに記録されます。

- カスタマイズの範囲は、Web サーバによって制限されることがあります。一部の Web サーバは、レスポンスに対して独自のテキストを追加することがあるからです。
- カスタマイズ済みテキスト ファイルのサイズは、システムのファイル サイズの上限のみによって制限を受けます。

エラー処理をセットアップする方法

サーバ エラー、アクセス拒否、cookie 要求、またはクロス サイト スクリプティング（CSS）の各エラーに関して、Web エージェントに適切なカスタム エラー ファイルまたはページを指定するには、ServerErrorFile、Custom401ErrorFile、ReqCookieErrorFile、または CSSErrorFile パラメータ内のファイル パスまたは URL をそれぞれ指定します。エラーファイルはアプリケーション内のどこにあってもかまいません。

重要: カスタム エラー ページとして設定するどの URL も保護されていないリソースです。保護されているので指定されたエラー ファイルを表示できないという誤設定を防ぐために、エラー ページとして設定するどの URL も ignoreURL リストに追加されます。

注: 使用するアプリケーションに HTML タグが含まれた URL が必要な場合は、これらの文字をエンコードして、Web エージェントが要求を遮断しないようにすることができます。HTML 文字のエンコードの詳細については、http://www.cert.org/tech_tips/ を参照してください。

次の例は、エラーファイルの中でのファイルのパスと URL を示しています。この例の中で示す構文は、ローカルのエージェント設定ファイルに関するものです。これらのパラメータは、エージェント設定オブジェクトの中で設定することもできます。

ファイルのパス

```
CSSErrorFile="C:¥error¥error.txt"
ReqCookieErrorFile="C:¥custompages¥error.txt"
ServerErrorFile="C:¥error¥error.txt"
Custom401ErrorFile="C:¥error¥accessdenied.txt"
```

URL:

```
CSSErrorFile="http://www.mycompany.com.error.jsp"
ReqCookieErrorFile="http://www.myorg.com.error.asp"
ServerErrorFile="http://www.mycompany.com.error.jsp"
```

第 11 章: IIS でのユーザ アクセスの管理

このセクションには、以下のトピックが含まれています。

IIS プロキシ ユーザ アカウントの使用(IIS のみ)

SiteMinder によって保護された IIS Web サーバ上のリソースにアクセスしようとしたユーザにそれらのリソースに対する十分な IIS 権限がない場合、Web エージェントがアクセスを拒否することがあります。たとえば、UNIX システム上の LDAP ユーザ ディレクトリに格納されているユーザは、IIS Web サーバを持つ Windows システムにアクセスできないことがあります。

SiteMinder からアクセス権を付与されたユーザは、IIS Web サーバによって十分な権限のあるデフォルトのプロキシ アカウントが得られます。ユーザが有効な Windows セキュリティ コンテキストを持つ場合でも、Web エージェントは DefaultUserName および DefaultPassword パラメータの値を認証情報として使用します。

プロキシ ユーザ アカウントを使用するように IIS Web エージェントを設定する方法

1. ForceIISProxyUser パラメータの値を以下の値の*いずれか*に設定します。
   - IIS サーバ上のアプリケーションへのアクセスがユーザの認証情報自体に基づく場合は、ForceIISProxyUser パラメータの値を yes に設定します。
   - IIS サーバ上のアプリケーションへのアクセスがユーザの代わりに動作する特定のアカウント(プロキシなど)に基づく場合は、ForceIISProxyUser パラメータの値を no に設定します。
2. 以下の Windows 機能のどちらも*使用していない*場合は、手順 3 に進みます。
   - Windows 認証方式
   - Windows ユーザ セキュリティ コンテキスト

3. DefaultUserName パラメータ内にプロキシ ユーザ アカウントのユーザ名を入力します。ドメイン アカウントと、そのドメインの一部ではないローカル マシンを使用している場合は、以下の例に示す構文を使用します。

   DefaultUserName=*Windows_domain¥acct_with_admin_privilege*

   それ以外の場合は、ユーザ名のみを指定します。

4. DefaultPassword パラメータ内に既存の Windows ユーザ アカウントに関連付けられたパスワードを入力します。

   重要: 暗号化できるので、エージェント設定オブジェクト内でこのパラメータを設定することをお勧めします。ローカル設定ファイル内で設定すると、値は暗号化されずにプレーン テキストで格納されます。

   IIS Proxy アカウントが設定されます。

匿名ユーザ アクセスの有効化

ユーザにプロキシ ユーザとしてのアクセス権を付与しない場合は、以下のパラメータを設定します。

UseAnonAccess

プロキシ ユーザの認証情報を使用するのではなく、匿名ユーザとして Web アプリケーションを実行するように IIS Web エージェントに指示します。

デフォルト: No

注: このパラメータは IIS Web エージェントにのみ適用されます。

匿名ユーザ アクセスを有効にするには、UseAnonAccess パラメータを yes に設定します。

IIS 認証での NetBIOS 名または UPN の使用

IIS ネットワークでは、要求されたリソースの場所に関して、ドメイン名とは異なる NetBIOS 名が存在する場合があります。保護されたリソースにアクセスを試みたときに複数のドメイン コントローラが存在すると、ユーザ認証は失敗し、Web サーバ ログに「IIS ログオン エラー」と表示されます。以下のパラメータを使用して、UPN または NetBIOS 名を IIS Web サーバに送信するかどうかを制御できます。

UseNetBIOSforIISAuth

IIS 6.0 Web エージェントが IIS ユーザ認証のために、ユーザ プリンシパル名（UPN）と NetBIOS 名のどちらを IIS 6.0 Web サーバに送信するかを指定します。

注: このパラメータは、Active Directory ユーザ ストアがポリシー サーバに関連付けられている場合のみ有効です。

このパラメータを有効にした場合は、SiteMinder の認証時にポリシー サーバが Active Directory からユーザ DN、UPN、および NetBIOS 名を抽出し、このデータを IIS 6.0 Web エージェントに送り返します。

管理 UI でユーザ ディレクトリに対して[認証済みユーザ セキュリティ コンテキストで実行]オプションを選択したかどうか、および UseNetBIOSforIIAuth パラメータをどのように設定したかに応じて、ユーザのログオン認証情報は以下のように送信されます。

- UseNetBIOSforIISAuth パラメータが no に設定されている場合、IIS 6.0 Web エージェントは UPN 名を送信します。
- UseNetBIOSforIISAuth パラメータが yes に設定されている場合、Web エージェントは NetBIOS 名を送信します。

IIS Web サーバは、Web エージェントから受け取った認証情報を使用してユーザを認証します。

デフォルト: No

Web エージェントで IIS 認証の NetBIOS 名が使用されるようにするには、UseNetBIOSAuth パラメータを yes に設定します。

NT チャレンジ/レスポンス認証を設定する方法（IIS のみ）

IIS Web エージェントでは、NT チャレンジ/レスポンス認証方式をサポートします。ユーザがリソースへのアクセスを要求するときに、IIS Web サーバは NT チャレンジ/レスポンス認証を使用して、ユーザの Internet Explorer ブラウザにチャレンジします。チャレンジとは、ユーザのクライアント システムに格納されているユーザのパスワードに基づいて行う数学的計算のことです。この計算結果は、ブラウザから Web サーバに返されます。Web サーバでは、レスポンスとデータベースのパスワード情報を比較し、同じ計算を実行します。 計算結果が一致すれば、そのユーザはアクセスを許可されます。この処理は、ユーザには見えません。

注: NT チャレンジ/レスポンス認証方式は、Internet Explorer ブラウザのみと連携します。

チャレンジ/レスポンス認証方式を実装するには、次のような 2 種類の方法があります。

- ユーザが保護されたリソースにアクセスしようとするときにユーザにチャレンジする（シングル サインオン環境では、ユーザが初めてリソースを要求したときにのみチャレンジする）
- ユーザに使用している Internet Explorer ブラウザの自動ログオン機能を設定させる。

  自動ログオン機能を使用すると、ユーザはチャレンジを受けずにリソースにアクセスできるようになります。認証処理は引き続き実行されますが、ブラウザとサーバの間の NT チャレンジ/レスポンス処理はユーザには見えません。一般的に、自動ログオンはイントラネットで使用されます。イントラネットでは、セキュリティがそれほど厳重ではないので、ユーザがリソースにシームレスにアクセスできるようにします。自動ログオンは、インターネットを介した通信にはお勧めできません。また、ユーザの Windows アカウントが Web サーバと同じ Windows ドメイン システムになければならないため、通常は使用することもできません。

SiteMinder エージェントは、認証情報コレクタを使用して、NT チャレンジ/レスポンス認証方式のためにユーザの Windows 認証情報を収集します。エージェントでは、NTLM 認証情報を収集する目的で、拡張子 .NTC をサポートしています。

注: このデフォルトの動作を変更したい場合にのみ、NTCEXT を設定する必要があります。

SiteMinder に NT チャレンジ/レスポンス認証を使って処理させるには、以下の手順に従います。

1. 次のタスクによって、IIS Web サーバの NT チャレンジ/レスポンス認証を設定します。
   a. ファイル拡張子 .ntc をマップします。
   b. 仮想ディレクトリを作成および設定し、その仮想ディレクトリが NT チャレンジおよびレスポンス認証情報を求めるようにします。
2. 管理 UI 内で NT チャレンジ/レスポンス認証方式を設定します。
3. NTLM 認証情報コレクタを指定します。
4. 管理 UI を使用して、NT チャレンジ/レスポンス認証に関するポリシーを設定します。
5. （オプション）ユーザに使用している Internet Explorer ブラウザの自動ログオン機能を設定させます。

   IIS の NT チャレンジ/レスポンス認証が設定されます。

ファイル拡張子 .NTC のマップ

IIS Web サーバ上で tNT チャレンジ/レスポンス認証を設定するには、ISAPIWebAgent.dll アプリケーションにファイル拡張子 .NTC をマップする必要があります。

ファイル拡張子 .NTC をマップする方法

1. インターネット サービス マネージャを開きます。
2. 左ペインの[Web サイト]を右クリックし、右ペインの[既定の Web サイト]を右クリックして[プロパティ]を選択します。

   [既定の Web サイトのプロパティ]ダイアログ ボックスが表示されます。
3. [ホーム ディレクトリ]タブをクリックします。
4. [アプリケーションの設定]グループ ボックスで[構成]をクリックします。

   [アプリケーションの構成]ダイアログ ボックスが表示されます。
5. [追加]をクリックします。

   [アプリケーションの拡張子マッピングの追加/編集]ダイアログ ボックスが開きます。

   a. [実行可能ファイル]フィールドで、[参照]をクリックし、次のファイルを見つけます。*web_agent_home*/bin/ISAPIWebAgent.dll。

   b. [開く]をクリックします。

   c. [拡張子]フィールドに「.ntc」と入力します。
6. [OK]を 3 回クリックします。

   [アプリケーションの拡張子マッピングの追加/編集]ダイアログ ボックス、[アプリケーション構成]ダイアログ ボックス、および[既定の Web サイトのプロパティ]ダイアログ ボックスが閉じます。ファイル拡張子 .ntc がマップされます。

Windows 認証方式の仮想ディレクトリの設定（IIS 6.0）

<stmdnr> Windows 認証方式を使用するには、IIS 6.0 Web サーバ上で仮想ディレクトリを設定します。仮想ディレクトリには、認証情報に関して NT チャレンジおよびレスポンスが必要です。

Windows 認証方式用の仮想ディレクトリを設定する方法

1. インターネット インフォメーション サービス（IIS）マネージャを開きます。
2. 左側のペインで、以下のアイテムを展開します。
   - Web サーバ アイコン
   - Web サイト フォルダ
3. 以下の*いずれか*の手順を実行します。
   - Web サイト全体のすべてのリソースを SiteMinder Windows 認証方式で保護するには、「Default Web Site」フォルダを右クリックし、[プロパティ]を選択して手順 4 に進みます。
   - SiteMinder Windows 認証方式で Web サイト全体を保護しない場合は、以下の手順を実行します。
     a. 以下のフォルダを見つけます。

     ```
     ¥siteminderagent¥ntlm
     ```

     b. ntlm フォルダを右クリックし、[プロパティ]を選択して手順 4 に進みます。

   [プロパティ]ダイアログボックスが表示されます。
4. [ディレクトリ セキュリティ]タブをクリックします。
5. [匿名アクセスおよび認証制御]グループ ボックスで[編集]をクリックします。

   [認証方法]ダイアログ ボックスが表示されます。
6. 以下の手順を実行します。
   - [匿名アクセスを有効にする]チェック ボックスをオフにします。
   - [統合 Windows 認証]チェック ボックスをオンにします。
7. [OK]を 2 回クリックします。

   [認証方法]ダイアログ ボックスおよび[プロパティ]ダイアログ ボックスが閉じます。仮想ディレクトリが設定され、認証情報に関して NT チャレンジおよびレスポンスを求めます。

注: これらの変更を反映するため Web サーバを再起動します。

Internet Explorer に対する自動ログオンの設定

Web エージェントがユーザの認証情報を*要求せずに*ユーザを認証する場合は、各ユーザにセキュリティ設定を変更することによって Internet Explorer ブラウザに対して Windows NT の自動ログイン機能を設定させます。

自動ログインの設定方法

1. Internet Explorer ブラウザを起動します。
2. 以下の*いずれか*をクリックします。
   - ［表示］メニュー（Internet Explorer 4.x または 5.x）
   - ［ツール］メニュー（Internet Explorer 6.0）
3. ［インターネット オプション］を選択します。

   ［インターネットオプション］ダイアログ ボックスが開きます。
4. ［セキュリティ］タブをクリックします。
5. 適切なセキュリティ ゾーンをクリックします。［インターネット］、［イントラネット］、［信頼済みサイト］、［制限付きサイト］から選択します。
6. 以下のいずれかをクリックします。
   - ［カスタム］ラジオ ボタン（Internet Explorer 4.x）
   - ［レベルのカスタマイズ］ラジオ ボタン（Internet Explorer 5.x または 6.0）
7. ［設定］をクリックします。
8. ［ユーザ認証］セクションまでスクロール ダウンします。［ログオン］オプションで、4.x の場合は［現在のユーザ名とパスワードで自動的にログオン］ラジオ ボタンを、5.x または 6.0 の場合は［現在のユーザ名とパスワードで自動的にログオンする］ラジオ ボタンをオンにします。
9. ［OK］を 2 回クリックします。

   ［セキュリティ設定］ダイアログ ボックスおよび［インターネット オプション］ダイアログ ボックスが閉じます。ユーザの設定が保存され、自動ログインが設定されます。

チャレンジ/レスポンス認証方式の設定

NT チャレンジ/レスポンス認証を実装するには、管理 UI を使用して認証方式を作成する必要があります。

注: 詳細については、ポリシー サーバドキュメントを参照してください。

NT チャレンジ/レスポンス認証方式を作成する場合は、次のように設定します。

- ［各方式共通セットアップ］グループボックス

  ［認証方式タイプ］: Windows Template

- ［方式タイプのセットアップ］タブ:

  ［サーバ名］: 完全修飾ドメイン名。たとえば、以下のようになります。

  server1.myorg.com

  ［ターゲット］: /siteminderagent/ntlm/smntlm.ntc

  注: このディレクトリは、インストール時にすでに設定された仮想ディレクトリと一致している必要があります。ターゲットである smntlm.ntc は、存在していなくてもかまいません。また、.ntc で終わる名前や、デフォルトの代わりに使用するカスタム MIME タイプでもかまいません。

- ［詳細］タブ:

  ［ライブラリ］: smauthntlm

NTLM 認証情報コレクタの指定

NTLM 認証情報コレクタ（NTC）は、Web エージェント内のアプリケーションです。NTC は、Windows 認証方式によって保護されているリソースに関連する NT 認証情報を収集します。この方式は、Internet Explorer ブラウザからアクセスされる、IIS Web サーバ上のリソースに適用できます。

各認証情報コレクタには、1 つの MIME タイプが関連付けられています。IIS に関しては、以下のパラメータで NTC MIME TYPE が定義されています。

NTCExt

NTLM 認証情報コレクタと関連付けられた MIME タイプを指定します。NTC は、Windows 認証方式によって保護されているリソースに関連する NT 認証情報を収集します。この方式は、Internet Explorer ブラウザからアクセスされる IIS Web サーバ上のリソースに適用できます。

このパラメータに複数の拡張子を持たせることもできます。エージェント設定オブジェクトを使用している場合は、複数値オプションを選択します。ローカル設定ファイルを使用している場合は、各拡張子をカンマで区切ります。

デフォルト： .ntc

使用している環境で前のパラメータによって指定されたデフォルトの拡張子がすでに使用されている場合は、別の MIME タイプを指定できます。

認証情報コレクタをトリガする拡張子を変更するには、他のファイル拡張子を NTCExt パラメータに追加します。

Information Card 認証方式を実装する方法

CA SiteMinder では、Windows CardSpace を実装する Information Card 認証方式（ICAS）をサポートします。保護されているリソースへのアクセスを求めるユーザは認証カードを選択できます。SiteMinder では、ユーザの身元を確認するためにカードに含まれている情報が使用されます。

ICAS の実装では、以下の SiteMinder コンポーネントの設定変更が必要です。

- SiteMinder Web エージェントをホストしているサーバ
- SiteMinder ポリシー サーバ
- smkey データベース

次の手順に従います。

1. Web サーバ上で以下のタスクを実行します。
   a. IIS 6.0 Web サーバ上で SSL 通信を有効にします。

      注: 詳細については、Microsoft のドキュメントを参照するか、または http://support.microsoft.com/ に移動します。

   b. Web サーバ証明書を .pfx ファイルとしてエクスポートします。
   c. SiteMinder InfoCard.fcc テンプレートをカスタマイズします。
2. ポリシー サーバ上で以下のタスクを実行します。
   a. ポリシー サーバに JCE をインストールします。
   b. ポリシー サーバ上の java.security ファイルを更新します。
   c. ポリシー サーバ上の config.properties ファイルを更新します。
   d. smkey データベースがまだない場合は、ポリシー サーバ設定ウィザードで作成します。
   e. Web サーバから smkey データベースに.pfx ファイル証明書を追加します。
   f. ポリシー サーバ内のユーザ ディレクトリの設定
   g. 管理 UI を使用した CardSpace のカスタム認証方式の作成
   h. （オプション）レスポンスで使用するセッション ストア内に、要求を格納します。
   i. （オプション）セッション ストアから要求値を取得できるようにすることにより、パーソナライズを有効にします。

j. （オプション）格納された要求値を取得するようにアクティブなレスポンスを設定します。

smkey データベースへの Web Server 証明書のエクスポート

ICAS の設定の一部として、別の形式で SiteMinder Web エージェントをホストしている Web サーバから SSL 証明書をエクスポートし、それを smkey データベースに配置することが含まれます。

Web Server 証明書をエクスポートする方法

1. インターネット インフォメーション サービス（IIS）マネージャが開きます。
2. Web サーバの展開後、Web サイト フォルダを展開します。
3. ［既定の Web サイト］（または ICAS の設定対象）を右クリックし、［プロパティ］を選択します。

   ［プロパティ］ページが表示されます。
4. ［ディレクトリ セキュリティ］タブをクリックします。
5. ［セキュリティで保護された通信］ペインで、［サーバー証明書］をクリックします。

   Web Server 証明書ウィザードが開きます。
6. ウィザードを使用して、.pfx ファイルに証明書をエクスポートします。

   重要: この手順で選択したパスワードを記憶させます。今後必要となります。
7. ［完了］をクリックしてウィザードを閉じ、［OK］をクリックして［プロパティ］ページを閉じます。
8. IIS マネージャを閉じます。
9. ポリシー サーバ管理者に .pfx 証明書ファイルを配布し、管理者がそれを smkey データベースにインポートするようにします。

   注: 詳細については、「Federation セキュリティ サービス Guide」を参照してください。

Information Card 認証方式のための FCC のテンプレートの設定

SiteMinder Web エージェントには、SiteMinder 内の ICAS を実装するために使用できるフォーム認証情報コレクタ(FCC)のテンプレートが含まれます。

Information Card 認証方式のための FCC のテンプレートを設定する方法

1. テキスト エディタで以下のデフォルトの FCC ファイルを開きます。

   web_agent_home¥samples_default¥forms¥InfoCard.fcc

2. 以下のディレクトリにファイルのコピーを保存します(これにより、後で必要になった場合に備えて、デフォルトの FCC 設定が維持されます)。

   web_agent_home¥samples¥forms¥

3. FCC のファイルのコピーから、以下の情報を記録します。

   重要: この情報は、ポリシー サーバを設定するために必要です。

   - Web エージェントをホストしている IIS Web サーバの完全修飾ドメイン名
   - 手順 2 で保存した FCC ファイルの名前
   - 手順 2 で保存した FCC ファイル内の requiredClaims パラメータ タグの値(引用符なし) 以下の例を参照してください。

     ```
     http://schemas.xmlsoap.org/ws/2005/05/identity/claims/privatepersonalidentifier
     ```

4. (オプション) テキスト エディタを使用して、FCC ファイルのコピーで以下の変更のいずれかを加えます。

   - カスタム ロゴを使用するには、netegrity_logo.gif ファイルを独自の画像に置換し、FCC ファイル内の以下のリンクをそれに応じて更新します。

     ```
     <img alt="Logo" src="/siteminderagent/dmspages/netegrity_logo.gif">
     ```

第 12 章: ユーザ アクティビティの追跡

このセクションには、以下のトピックが含まれています。

監査によるユーザ アクティビティまたはアプリケーション使用状況の追跡

監査によって、Web サイトでのアプリケーションの使用頻度を測定したり、ユーザ アクティビティを追跡したりすることができます。監査は以下のパラメータを使用して制御されます。

EnableAuditing

ユーザ セッション キャッシュに格納される正常に行われたすべてのユーザ許可に関する情報を Web エージェントが記録するかどうかを指定します。ユーザ許可を有効にすると、Web エージェントがポリシー サーバへ問い合わせを行わずにキャッシュの情報を使用しても、ユーザ許可情報は記録されます。ユーザがリソースにアクセスすると、Web エージェントはユーザ名とアクセス情報を固有の Web サーバ ログファイルに記録します。

デフォルト: No

ポリシー サーバと Web エージェントの両方でユーザ アクティビティを監査します。キャッシュに格納されている情報に基づいてユーザにリソースへのアクセスが許可されるたびに、Web エージェントからアカウンティング サービスにメッセージが送信されます。このアクションにより、アカウンティング サービスでは、ポリシー サーバと Web エージェントで正常に行われたユーザ許可を追跡します。Web エージェントが監査メッセージをアカウンティング サービスに送信できなかった場合、リソースへのアクセスは拒否されます。その後、管理 UI から SiteMinder アクティビティ レポートを実行できます。ポリシー サーバからのレポートには、各 SiteMinder セッションのユーザ アクティビティが示されます。

注: 詳細については、ポリシー サーバドキュメントを参照してください。

監査によってユーザ アクティビティまたはアプリケーション使用状況を追跡するには、EnableAuditing パラメータの値を yes に設定します。

トランザクション ID

Web エージェントは、ユーザ許可リクエストが成功するたびに、一意のトランザクション ID を生成します。エージェントは、HTTP ヘッダにその ID を追加します。ID は以下のログにも記録されます。

- 監査ログ
- Web サーバ ログ(サーバがクエリ文字列をログに記録するように設定されている場合)
- ポリシー サーバ ログ

トランザクション ID を使用して、所定のアプリケーションのユーザ アクティビティを追跡できます。

注: 詳細については、ポリシー サーバドキュメントを参照してください。

トランザクション ID は、モック クエリ パラメータとしてログに表示され、既存のクエリ文字列の末尾に追加されます。以下の例に、クエリ文字列(末尾は STATE=MA)に追加されたトランザクション ID(太字)を示します。

```
172.24.12.1, user1, 2/11/0, 15:30:10, W3SVC, MYSERVER, 192.168.100.100, 26844,
47, 101, 400, 123, GET, /realm/index.html,
STATE=MA&SMTRANSACTIONID=0c01a8c0-01f0-38a47152-01ad-02714ae1
```

URL にクエリ パラメータがない場合、エージェントはトランザクション ID を Web サーバ ログ エントリの末尾に追加します。以下に例を示します。

```
172.24.12.1, user1, 2/11/0, 15:30:10, W3SVC, MYSERVER, 192.168.100.100, 26844,
47, 101, 400, 123, GET, /realma/index.html,
SMTRANSACTIONID=0c01a8c0-01f0-38a47152-01ad-02714ae1.
```

注: ユーザがリソースにアクセスすると、Web エージェントは、ユーザ名とアクセス情報をネイティブの Web サーバ ログ ファイルに記録します。

Oracle iPlanet Web サーバ ログのトランザクション ID の記録

Solaris に該当

Oracle iPlanet Web サーバ ログに SiteMinder トランザクション ID を記録できます。

Oracle iPlanet Web サーバ ログに SiteMinder トランザクション ID を記録する方法

1. magnus.conf ファイルを開きます。
2. 以下のヘッダ変数を、Web サーバ初期化時にロギングする HTTP サーバ変数の既存リスト内に追加します。

   %Req->headers.SM_TRANSACTIONID%"

   注: エージェント設定オブジェクトまたはローカル設定ファイル内で LowerCaseHTTP パラメータの値を yes に設定しなかった場合は、ヘッダ変数を大文字で入力する必要があります。

   以下の例では、SMTRANSACTIONID ヘッダ変数を既存のエントリに最後に太字で示しています。ただし、変数のリスト内のどの場所にも配置できます。

   ```
   Init fn="flex-init" access="D:/iPlanet/server4/https-orion/logs/access"
   format.access="%Ses->client.ip% - %Req->vars.auth-user% [%SYSDATE%]
   ¥" %Req->srvhdrs.clf-status% %Req-srvhdrs.content-length% %Req->headers.-
   SM_TRANSACTIONID%"
   ```

3. 変更を適用するため Oracle iPlanet サーバを再起動します。

   トランザクション ID が Oracle iPlanet Web サーバ ログに表示されます。以下の例は、Web サーバ ログのエントリを示しています。ここでは、トランザクション ID を太字で示しています。

   ```
   11.22.33.44 - user1 [21/Nov/2003:16:12:24 -0500] "GET /Anon/index.html HTTP/1.0"
   200 748 3890b4b9-58f8-4a74df53-07f6-0002df88
   ```

詳細情報:

ヘッダでの小文字 HTTP の使用（Oracle iPlanet、Apache、Domino Web サーバ）
(P. 149)

Apache Web サーバ ログへのトランザクション ID の記録

Apache Web サーバ ログの SMTRANSACTIONID ヘッダ変数に SiteMinder トランザクション ID を記録できます。

Apache Web サーバ ログに SiteMinder トランザクション ID を記録する方法

1. httpd.conf ファイルを開きます。
2. LogFormat ディレクティブに SM_TRANSACTIONID ヘッダ変数を追加します。

   以下に例を示します。

   ```
   LogFormat "%h %l %u %t ¥"%r¥" %>s %b ¥"%{SM_TRANSACTIONID}i¥"" common
   ```

   注: httpd.conf ファイルおよび LogFormat ディレクティブの詳細については、Apache Web サーバのマニュアルを参照してください。
3. 変更を適用するにはサーバを再起動します。

   トランザクション ID が Apache Web サーバ ログに記録されます。

IIS 6.0 サーバ ログでのユーザ名およびトランザクション ID の記録

IIS 6.0 サーバ上のリソースを保護する Web エージェントは、SiteMinder トランザクション ID や認証されたユーザ名を IIS サーバ ログにデフォルトでは*記録しません*。これらの Web エージェントは ISAPI 拡張として機能するので、サーバにはすでにトランザクションが記録されています。以下のパラメータを使用して、この情報を追加するように Web エージェントに強制することができます。

AppendIISServerLog

認証されたユーザ名と IIS サーバ ログに対する SiteMinder トランザクション ID を個別の行に追加するように Web エージェントに指示します。

デフォルト: No

SetRemoteUser

いくつかのレガシー アプリケーションが必要とする場合がある REMOTE_USER 変数の値を指定します。

デフォルト: No

IIS サーバ ログにトランザクション ID およびユーザ名を記録する方法

1. AppendIISServerLog パラメータの値を yes に設定します。
2. SetRemoteUser パラメータの値を yes に設定します。

   IIS サーバ ログにはユーザ名およびトランザクション ID が含まれます。

第 13 章: ログの設定

このセクションには、以下のトピックが含まれています。

起動イベントのログ

デバッグを支援するために、起動時のイベントはログに記録されます。各メッセージは、問題の手がかりになる可能性があります。これらのログは、以下の場所に格納されます。

- Windows システムでは、Windows アプリケーション イベント ログに記録されます。
- UNIX システムでは、STDERR へ送信されます。Apache サーバは、STDERR を Apache の error_log ファイルにマップするので、これらのイベントもそのログに記録されます。

エラー ログとトレース ログ

Web エージェントのパフォーマンスを監視したり、Web エージェントとポリシー サーバの通信状態を確認したりする場合は、Web エージェントのロギング機能を使用します。ロギング機能は、パフォーマンスを分析して問題をトラブル シューティングすることを目的として、SiteMinder プロセスの動作に関する正確で包括的な情報を提供します。

ログとは、プログラムを実行している間に発生したイベントからなる記録のことです。1 つのログは、一連のログメッセージによって構成されています。各ログメッセージは、プログラムを実行している間に発生した何らかのイベントについて記述しています。ログメッセージは、ログファイルに書き込まれます。

注: IIS 6.0 Web エージェントは、最初のユーザ要求が送信されてから、ログ ファイルを作成します。Apache 2.0 Web エージェントでは Apache サーバの起動時にログ ファイルが作成されます。

Web エージェントは、以下のログ ファイルを使用します。

エラー ログ

プログラム レベルおよび操作レベルのエラーが含まれます。たとえば、エージェントがポリシー サーバと通信できない場合を示します。このログの詳細出力のレベルはカスタマイズできません。エラー ログに含まれるメッセージのタイプは、以下のとおりです。

エラーメッセージ

プログラム レベルのエラーが含まれます。これらは、プログラムの不適切または異常な動作、あるいはネットワーク障害のような何らかの外部の問題に起因すると考えられる機能障害を示しています。動作レベルのエラーもあります。このエラーのタイプは、ファイルを開く、またはユーザを認証するなどの動作の成功を妨げる障害を意味します。

情報メッセージ

何らかのイベントが発生したことをユーザまたは管理者に知らせることを目的とするメッセージです。つまり、サーバの起動や停止、または何らかのアクションが実施されたことを意味します。

警告メッセージ

通常とは異なる状況やイベント、または潜在的な問題を示唆している状況やイベントについて、ユーザまたは管理者に警告することを目的とするメッセージです。これは必ずしも、何かが誤っていることを意味するとは限りません。

トレース ログ

詳細な警告メッセージと情報メッセージが含まれます。これらは、設定することができます。たとえば、トレース メッセージやフロー状態メッセージが含まれます。また、このファイルには、ヘッダの詳細や cookie 変数などのデータも含まれます。トレース ログに含まれるメッセージは、以下のとおりです。

トレース メッセージ

トレースまたはデバッグ、あるいはその両方の目的で、プログラムの動作に関する詳細な情報を提供します。トレース メッセージは一般的に、通常の動作時は無効にしておきます。情報、警告、およびエラーの各メッセージとは対照的に、トレース メッセージはソース コード内に埋め込まれていて、容易にはローカライズできません。さらに、トレース メッセージには、メッセージ自体の他に、重要なデータが含まれていることがあります。たとえば、現在のユーザやレルムの名前です。

Web エージェントを設定するときに、エラー ログ ファイルとトレース ログ ファイルの両方のロケーションを指定します。エラー ログとトレース ログは、Web エージェントの正常な動作を妨げている可能性がある問題の解決に役立ちます。

注: Windows プラットフォーム上のエージェントで、EnableWebAgent パラメータを yes に設定すると、Web エージェント ログが確実に作成されます。EnableWebAgent を no（デフォルト）のままにして、ログ パラメータを設定した場合は、UNIX プラットフォーム上のエージェントに対するエージェント ログのみが作成されます。

ログ ファイルに表示されるパラメータ値

Web エージェントは、Web エージェントのエラー ログ ファイルに、設定パラメータとその値をリストしますが、トラディショナル エージェントとフレームワーク エージェントとでその方法は異なります。

フレームワーク エージェントでは、エージェント設定オブジェクトまたはローカル設定ファイルに入力されているとおりに、設定パラメータとその値がログ ファイルに正確に記録されます。値が正しくない可能性があるパラメータを含め、すべてのパラメータがログ ファイルに記録されます。

トラディショナル エージェントでは、パラメータ値を記録する前に処理が行われます。パラメータの値が正しい場合は、パラメータとその値がログ ファイルに記録されます。値が正しくないパラメータは、ログ ファイルに記録*されません*。

エラー ロギングのセットアップと有効化

管理者がロギングを有効にして、ログファイルの場所を指定するまでは、エラー ログは利用できません。エラー ロギングを有効にするパラメータや、ログ データを追加するなどのオプションを指定するパラメータは、ローカル設定ファイル、またはポリシー サーバのエージェント設定オブジェクトに定義されます。

IIS 6.0 Web サーバまたは Apache 2.0 Web サーバにインストールされている Web エージェントは、ローカル設定ファイルでローカルに設定されるログ パラメータの動的な設定をサポートしていません。したがって、管理者がパラメータを変更した場合でも、エージェントを再起動するまでは、その変更結果は有効になりません。しかし、ポリシー サーバ上のエージェント設定オブジェクト内でそれらのパラメータを設定した場合、ログに関するそれらの設定は保存され、動的に更新されます。

注: IIS 6.0 Web エージェントは、最初のユーザ要求が送信されてから、ログ ファイルを作成します。Apache 2.0 Web エージェントでは Apache サーバの起動時にログ ファイルが作成されます。

エラー ロギングをセットアップして有効にするには、以下の手順に従います。

1. ログ ファイルがまだない場合は、新しいログ ファイルと関連ディレクトリを作成します。
2. LogFile パラメータの値を yes に設定します。

   注: Web サーバのローカル設定ファイル内でこのパラメータの値を yes に設定すると、ポリシー サーバ上で定義されたあらゆるロギング設定より優先されます。たとえば、LocalConfig.conf ファイル内でこのパラメータの値を yes に設定すると、ポリシー サーバ上の対応するエージェント設定オブジェクトで AllowLocalConfig パラメータの値を no に設定しても、ログ ファイルは生成されます。ポリシー サーバのログ設定をすべて上書きするには、LocalConfig.conf ファイル内の（ファイルの名前やサイズなどを定義する）関連するロギング パラメータも設定してください。
3. ファイル名も含め、エラー ファイルの絶対パスを LogFileName パラメータの中で指定します。以下に例を示します。

   /export/iPlanet/servers/https-jsmith/logs/WebAgent.log
4. （省略可）以下のパラメータを設定します（ポリシー サーバのエージェント設定オブジェクト、またはローカル設定ファイルで設定します）。

   LogAppend

既存のログ ファイルの最後に新しいログ情報を追加します。このパラメータを no に設定した場合、ロギングが有効になるたびに、ログ ファイル全体が上書きされます。

デフォルト: No

LogFileSize

ログ ファイルのサイズ制限(メガバイト単位)を指定します。現在のログ ファイルがこの制限に到達すると、新しいログ ファイルが作成されます。新しいログ ファイルは、以下の命名規則のうちの 1 つを使用します。

- フレームワーク エージェントでは、新しいログ ファイルの名前は、元の名前にシーケンス番号が追加されたものになります。たとえば、サイズ制限に到達すると、myfile.log という名前のログ ファイルは myfile.log.1 に名前が変更されます。
- 従来のエージェントでは、新しいログファイルの名前は、元の名前に日付とタイムスタンプが追加されたものになります。たとえば、サイズ制限に到達すると、myfile.log という名前のログ ファイルは myfile.log.09-18-2003-16-07-07 に名前が変更されます。

古いファイルは手動でアーカイブするか削除する必要があります。

デフォルト: 0(ロールオーバーなし)

例: 80

LogLocalTime

ログがグリニッジ標準時(GMT)とローカル時間のどちらを使用するかを指定します。GMT を使用するには、この設定を no に変更します。このパラメータが存在しない場合は、デフォルト設定が使用されます。

デフォルト: yes

ローカル設定ファイルを使用する場合、設定は以下の例のようになります。

```
LogFile="yes"
LogFileName="/export/iPlanet/servers/https-myserver/logs/errors.log"
LogAppend="no"
LogFileSize="80"
LogLocalTime="yes"
```

エラー ロギングが有効になります。

トランスポート層インターフェース（TLI）ロギングの有効化

Web エージェントとポリシー サーバの間の接続を検討する場合は、トランスポート層インターフェース ロギングを有効にします。

TLI ロギングを有効にする方法

1. Web サーバに以下の環境変数を追加します。

   SM_TLI_LOG_FILE

2. 以下の例に示されるように、変数の値のディレクトリおよびログ ファイル名を指定します。

   directory_name/*log_file_name*.log

3. Web エージェントが有効であることを確認します。
4. Web サーバを再起動します。

   TLI ロギングが有効になります。

保存されるログ ファイルの数の制限

Web エージェントが維持するログ ファイルの数を制限できます。たとえば、Web エージェント ログを格納するシステム上のディスク空き容量を節約したい場合は、以下のパラメータを使用して、ログ ファイルの数を制限できます。

LogFilesToKeep

保持する Web エージェント ログ ファイルの数を指定します。以下の場合に新しいログ ファイルが作成されます。

- Web エージェントが起動したとき。
- ログ ファイルのサイズ制限（LogFileSize パラメータの値で指定）に達したとき。

このパラメータの値を変更しても、保持数を超える既存のログ ファイルは自動的に削除*されません*。たとえば、システムに 500 個のログ ファイルが格納されているときに、それらのファイルのうち 50 個のみを保持することを指定しても、Web エージェントは、残りの 450 個のファイルを削除*しません*。

このパラメータの値を 0 に設定すると、すべてのログ ファイルが保持されます。

デフォルト: 0

保存されるログ ファイルの数を制限する方法

1. 既存のログ ファイルをすべて、システムからアーカイブするか削除します。
2. LogAppend パラメータの値を no に設定します。
3. LogFilesToKeep パラメータの値を維持するログ ファイルの数に変更します。

トレース ロギングをセットアップする方法

トレース ロギングをセットアップするには、以下の手順に従います。

1. トレース ロギングをセットアップおよび有効化します。
2. 以下のリストを確認することによってトレース ログに記録する内容を決定します。
   - トレース ログ コンポーネントとサブコンポーネント
   - トレース メッセージ データ フィールド
   - データ フィールド フィルタ
3. デフォルトのトレース設定ファイルを複製します。
4. 記録するアイテムが含まれるように複製ファイルを変更します。
5. Web エージェントを再起動します。

トレース ロギングの設定

トレース ロギングを使用するには、事前に、トレース ログ ファイルの名前、ロケーション、およびパラメータを指定して、トレース ロギングを設定しておく必要があります。これらの設定によって、ファイル自体のサイズおよび形式を制御します。トレース ロギングを設定したら、トレース ログ ファイルのコンテンツを別個に指定します。これにより、トレース ログ ファイル自体のパラメータを変更することなく、必要に応じて、トレース ログに含める情報のタイプを変更することができます。

トレース ロギングを設定するには、以下の手順に従います。

1. Web サーバ上で WebAgentTrace.conf ファイルを見つけます。ファイルの複製を作成します。
2. エージェント設定オブジェクトまたはローカル設定ファイルを開きます。
3. TraceFile パラメータに yes を設定します。

   注: Web サーバのローカル設定ファイル内でこのパラメータの値を yes に設定すると、ポリシー サーバ上で定義されたあらゆるロギング設定より優先されます。たとえば、LocalConfig.conf ファイル内でこのパラメータの値を yes に設定すると、ポリシー サーバ上の対応するエージェント設定オブジェクトで AllowLocalConfig パラメータの値を no に設定しても、ログ ファイルは生成されます。ポリシー サーバのログ設定をすべて上書きするには、LocalConfig.conf ファイル内の（ファイルの名前やサイズなどを定義する）関連するロギング パラメータも設定してください。
4. TraceFileName パラメータの中で、トレース ログ ファイルの絶対パスを指定します。これは、トレース ログ出力を保持するファイルです。
5. WebAgentTrace.conf ファイルのコピー（手順 1 で作成）のフル パスを以下のパラメータに指定します。

   TraceConfigFile

   監視するコンポーネントとイベントを決定する WebAgentTrace.conf 設定ファイルの場所を指定します。

   デフォルト: デフォルトなし

   例: C:¥Program Files¥ca¥webagent¥config¥WebAgentTrace.conf

   注: このファイルは、Web サーバを再起動するまで使用されません。
6. エージェント設定オブジェクトまたはローカル設定ファイルに以下のパラメータを設定することで、トレース ログ ファイルの情報の形式を定義します。

   TraceAppend

ロギングが有効になるたびにファイル全体を書き直す代わりに、既存のログ ファイルの最後に新しいログ情報を追加します。

デフォルト: No

TraceFormat

トレース ファイルがメッセージを表示する方法を指定します。以下の*いずれか*のオプションを選択します。

- default - 角かっこ[]を使用してフィールドを囲みます。
- fixed - 固定幅のフィールドに使用します。
- delim - 選択した文字を使用してフィールドを区切ります。
- xml - XML-like タグを使用します。Web エージェントには、DTD（Document Type Definition、文書タイプ定義）や、他のスタイルシートは*付属していません*。

デフォルト: default（角かっこ）

TraceDelimiter

トレース ファイル内のフィールドを区切るカスタム文字を指定します。

デフォルト: デフォルトなし

例: |

TraceFileSize

トレース ファイルの最大サイズを指定します(メガバイト単位)。この制限に到達すると、Web エージェントは新しいファイルを作成します。

デフォルト: 0(新しいログ ファイルは作成されません)

例: 20(MB)

LogLocalTime

ログがグリニッジ標準時(GMT)とローカル時間のどちらを使用するかを指定します。GMT を使用するには、この設定を no に変更します。このパラメータが存在しない場合は、デフォルト設定が使用されます。

デフォルト: yes

7. Web エージェントが目的のアクティビティを監視するように、WebAgentTrace.conf ファイルを編集します。

   フレームワーク Web エージェントは、エージェント設定ファイルでローカルに設定されたログ パラメータの動的な設定をサポートしていません。したがって、パラメータを変更しても、Web サーバを再起動するまでは、その変更結果は有効になりません。しかし、ポリシー サーバ上のエージェント設定オブジェクト内でそれらのパラメータを設定した場合、ログに関するそれらの設定は保存され、動的に更新されます。

   注: IIS 6.0 Web エージェントは、最初のユーザ要求が送信されてから、ログ ファイルを作成します。Apache 2.0 Web エージェントでは Apache サーバの起動時にログ ファイルが作成されます。

8. Web エージェントで新しいトレース設定ファイルが使用されるよう、Web サーバを再起動します。

トレース ログ コンポーネントとサブコンポーネント

Web エージェントは、特定の SiteMinder コンポーネントを監視できます。コンポーネントを監視すると、そのコンポーネントに対するすべてのイベントがトレース ログに記録されます。各コンポーネントには 1 つ以上のサブコンポーネントがあり、Web エージェントはこれらも監視できます。Web エージェントに、コンポーネントに対するイベントを必ずしもすべて記録させる必要がない場合は、監視する必要があるサブコンポーネントのみを指定することができます。

たとえば、Web サーバ上の Web エージェントに対するシングル サインオン メッセージのみを記録する場合は、Web エージェント コンポーネントと SSO サブコンポーネントを指定します。

使用可能なコンポーネントとサブコンポーネントは、以下のとおりです。

AgentFramework

すべてのエージェント フレームワーク メッセージを記録します（フレームワーク エージェントにのみ適用されます）。使用可能なサブコンポーネントは、以下のとおりです。

- 管理
- フィルタ
- HighLevelAgent
- LowLevelAgent
- LowLevelAgentWP

AffiliateAgent

4.x のアフィリエイト エージェントに関連する Web エージェント メッセージを記録します。4.x のアフィリエイト エージェントは、Federation セキュリティ サービス（別途購入製品）の一部です（フレームワーク エージェントにのみ適用されます）。使用可能なサブコンポーネントは、以下のとおりです。

- RequestProcessing

SAMLAgent

SAML アフィリエイトエージェントに関連する Web エージェントメッセージです。（フレームワーク エージェントにのみ適用されます）。使用可能なサブコンポーネントは、以下のとおりです。

- RequestProcessing

WebAgent

すべての Web エージェント ログ メッセージを記録します。IIS 6.0 または Apache 2.0 エージェントを*除く*すべてのエージェントに適用されます。使用可能なサブコンポーネントは、以下のとおりです。

- AgentCore
- キャッシュ
- 認証
- レスポンス
- 管理
- SSO
- フィルタ

Agent_Functions

すべてのエージェント API メッセージを記録します。使用可能なサブコンポーネントは、以下のとおりです。

- Init
- UnInit
- IsProtected
- ログイン
- ChangePassword
- 確認
- ログアウト
- 認証
- 監査
- FreeAttributes
- UpdateAttributes
- GetSessionVariables
- SetSessionVariables
- DeleteSessionVariables
- トンネル
- GetConfig
- DoManagement

Agent_Connection_Manager

エージェント API の内部処理に関連するメッセージを記録します。使用可能なサブコンポーネントは、以下のとおりです。

- RequestHandler
- クラスタ
- サーバ
- WaitQueue
- 管理
- 統計

各サブコンポーネントの説明については、WebAgentTrace.conf ファイルを参照してください。

トレース メッセージ データ フィールド

メッセージに含めるデータ フィールドを指定することにより、特定のコンポーネントの各トレース メッセージに何を含めるかを定義することができます。

データ フィールドの構文は、以下のとおりです。

```
data:data_field1,data_field2,data_field3
```

以下の例に、いくつかのデータ フィールドを示します。

```
data:message,date,time,user,agentname,IPAddr
```

メッセージによっては、フィールドに対するデータが存在しないこともあるため、フィールドが空になる場合もあります。たとえば、データ フィールドとして RealmOID を選択した場合、トレース メッセージによって、レルムの OID が表示される場合と表示されない場合があります。

使用可能なデータ フィールドは、以下のとおりです。

Message

実際のトレース メッセージが含まれます。

SrcFile

トレース メッセージのソース ファイルと行番号が含まれます。

Pid

プロセス ID が含まれます。

Tid

スレッド ID が含まれます。

Date

日付が含まれます。

Time

時間が含まれます。

PreciseTime

ミリ秒単位までの時間が含まれます。

Function

トレース メッセージが入っているコード内の関数が含まれます。

User

ユーザの名前が含まれます。

Domain

SiteMinder ドメインが含まれます。

Realm

SiteMinder レルムが含まれます。

AgentName

使用されているエージェント名が含まれます。

TransactionID

トランザクション ID が含まれます。

DomainOID

SiteMinder ドメイン OID が含まれます。

IPAddr

クライアント IP アドレスが含まれます。

RequestIPAddr

エージェントがあるサーバの IP を表示するトレース ファイルが含まれます。

IPPort

クライアント IP ポートが含まれます。

CertSerial

証明書シリアル番号が含まれます。

SubjectDN

証明書の所有者 DN が含まれます。

IssuerDN

証明書の発行者 DN が含まれます。

SessionSpec

SiteMinder セッション仕様が含まれます。

SessionID

SiteMinder セッション ID が含まれます。

UserDN

ユーザ DN が含まれます。

Resource

要求されたリソースが含まれます。

Action

要求されたアクションが含まれます。

RealmOID

レルム OID が含まれます。

ResponseTime

CA Web エージェントまたは SDK エージェントと API アプリケーションに関連する、ポリシー サーバの平均レスポンス時間(ミリ秒単位)が含まれます。

注: ResponseTime をトレース ログに出力するには、WebAgentTrace.conf ファイル、またはポリシー サーバ設定オブジェクト(ACO)に指定されたその他のファイルに、データ フィールド ResponseTime と一緒にコンポーネント Agent_Con_Manager を含めて、ポリシー サーバを再起動してください。Agent_Con_Manager コンポーネント、つまりエージェント API 接続マネージャは、ポリシー サーバからレスポンスを受け取るたびに ResponseTime を計算して、実行の平均を維持します。トレース ログで ResponseTime を見つけるには、[PrintStats]を検索してください。

トレース メッセージ データ フィールド フィルタ

特定の問題に焦点を当てるために、データ フィールドの値に基づいてフィルタを指定することによって、トレース ログの出力を絞り込むことができます。たとえば、index.html ページで問題が発生している場合、トレース設定ファイル内で Resource:==/html と指定することにより、html サフィックス(拡張子)を持つリソースのみをフィルタして抽出することができます。各フィルタは、ファイル内で個別の行を使用して記述する必要があります。

フィルタは、以下の構文を使用します。

data_field:*filter*

使用可能なフィルタのタイプは、以下のとおりです。

- ==(完全一致)
- !=(等しくない)

フィルタは、以下の例に示すように、ブール ロジックを使用します。

Action:!=get(get 以外のすべてのアクション)

Resource:==/html(末尾が /html のすべてのリソース)

トレース ログのコンテンツの決定

WebAgentTrace.conf ファイルによって、トレース ログのコンテンツが決まります。トレース ログに表示するコンポーネントとデータ項目は、Web サーバ上の WebAgentTrace.conf ファイルの設定を変更することで制御できます。ファイル編集時の考慮事項は、以下のとおりです。

- エントリは、大文字と小文字が区別されます。

  コンポーネント、データ フィールド、またはフィルタを指定する場合、その値は、WebAgentTrace.conf ファイルの命令内にあるオプションと正確に一致している必要があります。

- 設定に関する行は、コメント解除してください。そうしないと、それらの行は無視されます。
- 既存の Web エージェントの上に新しい Web エージェントをインストールする前に WebAgentTrace.conf ファイルを変更すると、ファイルは上書きされます。したがって、事前にファイルの名前を変更するか、ファイルのバックアップを作成する必要があります。インストール後、変更を新しいファイルに統合することができます。

トレース ログのコンテンツを決定するには、以下の手順に従います。

1. WebAgentTrace.conf ファイルを開きます。

   注: 元のファイルを複製し、コピーを変更することをお勧めします。これにより、デフォルト設定が維持されます。

2. 以下のステップに従って、コンポーネントとサブコンポーネントを追加します。

   a. エージェントのタイプと一致するセクションを見つけます。たとえば、サーバに Apache 2.0 Web エージェントがインストールされている場合は、以下のような行を探します。

   ```
   # For Apache 2.0, Apache 2.2, IIS 6.0, IIS 7.0 and SunOne Web Agents
   ```

   SiteMinder r12.0 SP3 IIS Web エージェントは、ISAPI フィルタおよび拡張で、IIS 6 および IIS 7 の両方で動作します。IIS 7 のインストールでは、SiteMinder Web エージェントを設定する前に Web サーバ上にロール サービスを追加する必要があります。

   注: 詳細については、「SiteMinder Web エージェント インストール ガイド」を参照してください。

   b. そのセクションで以下の行を見つけます。

   ```
   #components:
   ```

c. 行をコメント解除して(コメント化されている場合)、コロンの後ろに目的のコンポーネント名を追加します。コンポーネントが複数ある場合は、以下の例のように、カンマで区切ります。

```
components:  AgentFramework, HTTPAgent
```

d. (任意)コンポーネント名の後ろに、目的のサブコンポーネントの名前を追加します。以下の例のように、サブコンポーネント名はスラッシュで区切ります。

```
components:  AgentFramework/Administration
```

3. 以下のステップに従って、データ フィールドとフィルタを追加します。

a. 該当するセクションで以下の行を見つけます。

```
#data:
```

b. 行をコメント解除して(コメント化されている場合)、コロンの後ろに目的のデータ フィールドを追加します。データ フィールドが複数ある場合は、以下の例のように、カンマで区切ります。

```
data: Date, Time, Pid, Tid, TransactionID, Function, Message, IPAddr
```

c. (任意)データ フィールドの後ろに、コロン、ブール演算子、および目的の値を付けて、データ フィールドにフィルタを追加します。フィルタに指定する値は、完全に一致する必要があります。以下の例は、特定の IP アドレスのアクティビティをログに記録するフィルタを示しています。

```
data: Date, Time, Pid, Tid, TransactionID, Function, Message,
IPAddr:==127.0.0.1
```

注: 各フィルタは、ファイル内の別々の行に指定する必要があります。

4. 変更を保存し、ファイルを閉じます。
5. 変更を適用するために Web サーバを再起動します。

トレース ログのコンテンツが決定されました。

保存されるトレース ログ ファイルの数の制限

Web エージェントが維持するトレース ログの数を制限できます。たとえば、Web エージェント ログを格納するシステム上のディスク空き容量を節約したい場合は、以下のパラメータを使用して、トレース ログの数を制限できます。

TraceFilesToKeep

保持する Web エージェント トレース ログ ファイルの数を指定します。以下の場合に新しいトレース ログが作成されます。

- Web エージェントが起動したとき。
- トレース ログのサイズ制限（TraceFileSize パラメータの値で指定）に達したとき。

このパラメータの値を変更しても、保持数を超える既存のトレース ログは自動的に削除*されません*。たとえば、システムに 500 個のトレース ログが格納されているときに、それらのファイルのうち 50 個のみを保持することを指定しても、Web エージェントは、残りの 450 個のトレース ログを削除*しません*。

このパラメータの値を 0 に設定すると、すべてのトレース ログが保持されます。

デフォルト: 0

保存されるトレース ログの数を制限する方法

1. 既存のトレース ログをすべて、システムからアーカイブするか削除します。
2. TraceAppend パラメータの値を no に設定します。
3. TraceFilesToKeep パラメータの値を維持するトレース ログの数に変更します。

Agent Connection Manager のトレース ログによる詳細なエージェント接続データの収集

Web エージェントとポリシー サーバの間の接続に関する詳細情報を収集するために、Agent Collection Manager によって収集された情報が含まれているトレース ログ ファイルを作成します。

詳細な Web エージェント接続データを収集する方法

1. エージェント設定オブジェクトまたはローカル設定ファイルを開きます。
2. TraceFile パラメータの値を yes に設定します。

   注: Web サーバのローカル設定ファイル内でこのパラメータの値を yes に設定すると、ポリシー サーバ上で定義されたあらゆるロギング設定より優先されます。たとえば、LocalConfig.conf ファイル内でこのパラメータの値を yes に設定すると、ポリシー サーバ上の対応するエージェント設定オブジェクトで AllowLocalConfig パラメータの値を no に設定しても、ログ ファイルは生成されます。さらに、ポリシー サーバのトレース ログ設定をすべて上書きするには、LocalConfig.conf ファイル内の (ファイルの名前やサイズなどを定義する) 関連するトレース ロギング パラメータを設定してください。
3. TraceFileName パラメータの中で、エージェント接続データのトレース ログ ファイルの絶対パスを指定します。これは、トレース ログ出力を保持するファイルです。
4. 以下のファイルの絶対パスに TraceConfigFile パラメータの値を設定します。

   web_agent_home/config/AgentConMgr.conf

   注: 以下の例のとおり、*web_agent_home* 変数は、Web Agent のインストール場所を示します。

   - Windows インストールのデフォルトの場所: C:¥Program Files¥CA¥webagent

     UNIX インストールのデフォルトの場所 : /opt/ca/webagent
5. 以下のパラメータを設定することによって、エージェント接続データのトレース ログ ファイルの形式を指定します。

   TraceAppend

   ロギングが有効になるたびにファイル全体を書き直す代わりに、既存のログ ファイルの最後に新しいログ情報を追加します。

   デフォルト: No

   TraceDelimiter

トレース ファイル内のフィールドを区切るカスタム文字を指定します。

デフォルト: デフォルトなし

例: |

TraceFileSize

トレース ファイルの最大サイズを指定します(メガバイト単位)。この制限に到達すると、Web エージェントは新しいファイルを作成します。

デフォルト: 0(新しいログ ファイルは作成されません)

例: 20(MB)

TraceFormat

トレース ファイルがメッセージを表示する方法を指定します。以下の*いずれかのオプション*を選択します。

- default - 角かっこ[]を使用してフィールドを囲みます。
- fixed - 固定幅のフィールドに使用します。
- delim - 選択した文字を使用してフィールドを区切ります。
- xml - XML-like タグを使用します。Web エージェントには、DTD (Document Type Definition、文書タイプ定義)や、他のスタイルシートは*付属していません*。

デフォルト: default(角かっこ)

LogLocalTime

ログがグリニッジ標準時(GMT)とローカル時間のどちらを使用するかを指定します。GMT を使用するには、この設定を no に変更します。このパラメータが存在しない場合は、デフォルト設定が使用されます。

デフォルト: yes

6. Web サーバを再起動すると、新しい設定が有効になります。

   Web エージェント接続に関する詳細情報が収集されます。

   注: SiteMinderr12.0 SP3 では、BusyHandleCount と FreeHandleCount の属性は使用されません。

第 14 章: パフォーマンスの調整

このセクションには、以下のトピックが含まれています。

Web エージェント キャッシュ

Web エージェントは、ユーザ セッションとリソース情報をキャッシュメモリに格納します。この手法は、Web エージェントの効率を向上させます。ユーザがアクセスを要求するたびに、Web エージェントがポリシー サーバから情報を取得する必要がなくなるからです。

キャッシュを設定することで、これらの情報の格納方法を管理できます。キャッシュの大きさは、キャッシュ エントリの数で表されます。それぞれのキャッシュの総エントリ数は、指定された最大キャッシュ サイズを超えることはできません。

注: Web エージェントのキャッシュ設定の変更を有効にするには、Web サーバを再起動する必要があります。

キャッシュ管理には、次のガイドラインが適用されます。

- キャッシュが満杯になると、最も直近で利用されていないエントリから新しいエントリに置き換えられます。
- リソース キャッシュの場合、ResourceCacheTimeout パラメータの値に達すると、エントリが削除されます。
- ユーザ セッション キャッシュの場合、レルムごとに設定したセッションのタイムアウト値に基づいてエントリが削除されます。

ポリシーを変更すると、SiteMinder はキャッシュ内のリソース情報を削除します。また、管理 UI を使用して、ユーザ キャッシュとリソース キャッシュを手動で消去することもできます。

注: 詳細については、ポリシー サーバ ドキュメントを参照してください。

リソース エントリをキャッシュに保存しておく時間の制御

以下のパラメータを使用して、リソース エントリをキャッシュに保存しておく時間の長さを変更することができます。

ResourceCacheTimeout

リソース エントリがキャッシュに保存される秒数を指定します。時間間隔の値を超えると、Web エージェントはキャッシュされたエントリを削除します。その後、保護されているリソースにユーザがアクセスしようとすると、Web エージェントはポリシー サーバに問い合わせます。

デフォルト: 600（10 分）

注: このパラメータの値を変更した場合は、変更を適用するために Web サーバを再起動する必要があります。

リソース エントリをキャッシュに保存しておく時間を変更するには、ResourceCacheTimeout パラメータを目的の秒数に設定します。

リソース キャッシュの最大サイズの設定

以下のパラメータを使用して、Web ページなど、Web エージェントが追跡するリソース キャッシュ エントリの最大数を設定することができます。

MaxResourceCacheSize

Web エージェントがそのリソース キャッシュ内で保持するエントリの最大数を指定します。エントリには以下の情報が含まれます。

- リソースが保護されるかどうかに関するポリシー サーバのレスポンス
- レスポンスで返される追加属性

最大値に達すると、新しいリソース レコードが最も古いリソース レコードと置き換わります。

これらをより大きな数値に設定する場合は、十分なシステム メモリがあることを確認してください。

OneView モニタを使用して Web エージェント統計を表示している場合は、ResourceCacheCount に表示される値が MaxResourceCacheSize パラメータで指定された値より大きいことがあります。これはエラーではありません。Web エージェントは、MaxResourceCacheSize パラメータを 1 つのガイドラインとして使用します。また、値は状況により異なります。これは、MaxResourceCacheSize パラメータはリソース キャッシュ内の平均サイズのエントリの最大数を示すためです。実際のキャッシュ エントリは、あらかじめ識別された平均サイズより大きかったり小さかったりする可能性があります。したがって、実際の最大エントリ数は指定された値より多い場合や少ない場合があります。

注: フレームワーク エージェントなど、共有メモリを使用する Web エージェントの場合、キャッシュは MaxResourceCacheSize の値に基づいて一定サイズが事前に割り当てられ、それより増えることはありません。

デフォルト:（Domino Web サーバ）1000

デフォルト:（IIS および Sun Java System Web サーバ）700

デフォルト:（Apache Web サーバ）750

リソース キャッシュの最大サイズを設定するには、以下の手順に従います。

1. MaxResourceCacheSize パラメータの値を、目的のリソース最大数に設定します。
2. フレームワーク エージェントでは、変更を適用するために Web サーバを再起動する必要があります。

リソース キャッシュの最大サイズが変更されます。

リソース キャッシュの無効化

動的な一意の URL を使用するアプリケーションを保護している場合は、リソース キャッシュを無効にすることをお勧めします。アプリケーションによって使用される URL が一意であるため、それらはキャッシュから読み取られません。

リソース キャッシュを無効にするには、MaxResourceCacheSize の値をゼロに変更します。

ユーザ セッション キャッシュの最大サイズの設定

以下のパラメータを使用して、エージェントがセッション キャッシュ内で保持するユーザの最大数を設定することができます。

MaxSessionCacheSize

エージェントがそのセッション キャッシュ内で保持するユーザの最大数を指定します。セッション キャッシュには、認証するユーザのセッション ID が正常に格納されます。それらのユーザが同じセッション中に同じレルム内の別のリソースにアクセスした場合、エージェントはポリシー サーバをコールする代わりにセッション キャッシュの情報を使用します。この最大数に達すると、エージェントは最も古いユーザ レコードを新しいユーザ レコードと置き換えます。

このパラメータの値は、持続期間にリソースにアクセスしてそれを使用する予定のユーザの数に基づいて設定します。これらをより大きな数値に設定する場合は、十分なシステム メモリがあることを確認してください。

デフォルト:（Domino Web サーバ）1000

デフォルト:（IIS および Oracle iPlanet Web サーバ）700

デフォルト:（Apache Web サーバ）750

ユーザ セッション キャッシュの最大サイズを設定する方法

1. MaxSessionCacheSize パラメータの値を必要なユーザの最大数に設定します。
2. フレームワーク エージェントでは、変更を適用するために Web サーバを再起動する必要があります。

   ユーザ セッション キャッシュの最大サイズが変更されました。

匿名ユーザのキャッシング

以下のパラメータを使用して、キャッシュに匿名ユーザの情報を格納するように Web エージェントを設定することができます。

CacheAnonymous

Web エージェントが匿名のユーザ情報をキャッシュするかどうかを指定します。このパラメータは、たとえば以下の状況に対して設定できます。

- Web サイトのユーザのほとんどが匿名ユーザで、それらのユーザのセッション情報を格納したい場合。
- 登録ユーザと匿名ユーザの両方が Web サイトにアクセスする場合。

  匿名ユーザの情報のみでキャッシュが満杯になり、登録ユーザ用の領域がなくなってしまう可能性がある場合は、このパラメータを無効にすることをお勧めします。

デフォルト: No

キャッシュに匿名ユーザの情報を格納するには、CacheAnonymous パラメータの値を yes に設定します。

HOST ヘッダを送信しないテスティング ツールへの対応

SiteMinder Web エージェントでは、HTTP リクエスト内の HOST ヘッダの値を使用して以下の設定を判断します。

- エージェント名
- サーバ名
- サーバの IP アドレス

HTTP バージョン 0.9 および 1.0 では HOST ヘッダを使用しないため、SiteMinder Web エージェントでは HTTP バージョン 1.1 リクエストのみを受け入れます。そのため、HOST ヘッダを送信しないテスト ツールで問題が生じています。Web エージェントでそれらのリクエストが拒否されるからです。

SiteMinder r12.0 SP3 では、HOST ヘッダ値を定義するための新しいエージェント設定パラメータがサポートされます。Web エージェントでは、HOST ヘッダが含まれないすべてのリクエストでこの値を使用します。

HOST ヘッダを送信しないテスティング ツールに対応する方法

1. 以下のいずれかのアイテムを開きます。
   - 中央設定を使用している場合は、エージェント設定オブジェクトを開きます。
   - ローカル設定を使用している場合は、LocalConfig.conf ファイルを開きます。
2. 以下のパラメータを追加します。

   DefaultHostName

   HOST ヘッダに対する値を定義します。HTTP バージョン 0.9 または 1.0 リクエスト(HOST ヘッダなし)を送信するテスト/パフォーマンス ツールを使用するには、エージェント設定オブジェクトまたは LocalConfig.conf ファイルにこのパラメータを追加します。このパラメータが設定されていない場合、Web エージェントでは HTTP 1.1 リクエストのみを受け入れます。

   デフォルト: なし(空白)

   例: webserver.example.com
3. 上記のパラメータの値を適切なホスト名に設定します。前述の例を参照してください。
4. 以下のいずれかのアイテムを保存して閉じます。

- 中央設定を使用している場合は、エージェント設定オブジェクトを保存して閉じます。
- ローカル設定を使用している場合は、LocalConfig.conf ファイルを保存して閉じます。

Web エージェントでは、HOST ヘッダのない HTTP リクエストに対して代わりに DefaultHostName の値を使用します。

第 15 章: プロキシ サーバでの Web エージェントの使用

このセクションには、以下のトピックが含まれています。

プロキシ サーバの背後にあるエージェントの設定

Web エージェントをプロキシ サーバの背後にインストールする場合、以下のパラメータを使用して、プロキシ サーバで動作する Web エージェントを設定できます。

ProxyTrust

プロキシ サーバが実行した許可を信頼するよう、アクセス先サーバに対応する Web エージェントに指示します。アクセス先サーバに対応する Web エージェントがユーザを再度許可する必要がないため、これはより効率的です。

デフォルト: no

ExpireForProxy

クライアントがコンテンツ(ページおよび潜在的なヘッダまたは cookie)をキャッシュしないようにします。このパラメータの値が Yes に設定された場合、Web エージェントは以下の HTTP ヘッダのいずれかを HTTP レスポンスに挿入します。

- Expires
- Cache-control

コンテンツをキャッシュしない場合、後続の要求は引き続き転送されます。

ExpireForProxy パラメータが yes に設定された場合、Web エージェントでは、適切な ProxyHeaders<*suffix_name*> パラメータに指定された文字列を、エージェントが実行したリクエストの種類に基づいて、HTTP レスポンスに挿入します。

HTTP/1.1 リクエストの場合、エージェントは、以下のパラメータの値をヘッダとしてレスポンスに挿入します。

- ProxyHeadersAutoAuth
- ProxyHeadersProtected
- ProxyHeadersUnprotected

HTTP/1.0 リクエストの場合、エージェントは、以下のパラメータの値をヘッダとしてレスポンスに挿入します。

- ProxyHeadersAutoAuth10
- ProxyHeadersProtected10
- ProxyHeadersUnprotected10

デフォルト: No

注: このパラメータ名には 'proxy' という単語が含まれていますが、このパラメータの設定は、Web ブラウザ、またはこのパラメータ設定を使用する SiteMinder エージェントが動作する Web サーバに接続するすべてのクライアントの動作にも影響があります。

ページをキャッシュしないようプロキシに指示するために、Web エージェントはそのページに関する Expires ヘッダを追加します。このヘッダは、過去の日付に設定されています。そのため、HTTP 1.0 仕様で規定されているように、プロキシがそのページをキャッシュすることは防止されます。302 リダイレクトが発生した場合、cache-control: no-cache（キャッシュ制御: キャッシュがありません）ヘッダが代わりに設定されます。これは、コンテンツのキャッシングを防止しますが、マイクロソフト サポートによる説明のとおり、Internet Explorer（IE）ブラウザを使用している場合は、ブラウズ操作に悪影響を及ぼします。

302 リダイレクトに対して cache-control: no-cache を使用する場合、IE の中でインプレース文書の表示を管理する ActiveX コンポーネントは、ファイルを検索する際に、ブラウザのキャッシュを必要とします。このヘッダは、ファイルをキャッシュしないようブラウザに指示するので、この ActiveX コンポーネントはファイルを検索することができず、リクエストを正しく表示することに失敗します。さらに、Web エージェントの ExpireForProxy 設定項目を yes に設定すると、バックエンドサーバはプロキシに対し、リソースをキャッシュしないよう指示します。

プロキシ サーバの背後にあるエージェントを設定する方法

1. ProxyTust パラメータに yes を設定します。
2. ExpireForProxy パラメータに yes を設定します。

3. （任意）Cache-Control および ExpireForProxy（HTTP）ヘッダの値をカスタマイズします。

   プロキシ サーバの背後にあるエージェントが設定されます。

詳細情報:

Cache-Control ヘッダ設定と ExpireForProxy ヘッダ設定のカスタマイズ (P. 216)

Cache-Control ヘッダ設定と ExpireForProxy ヘッダ設定のカスタマイズ

cache-control と ExpireForProxy の各ヘッダをカスタマイズして、アプリケーション ファイル（.doc、.pdf など）のインプレース起動に影響を及ぼすことなく、Web リソースを安全にすることができます。以下のコンテンツ タイプの特定の HTTP ヘッダを別途設定することによって、Web ブラウザまたはプロキシ サーバによってコンテンツがどのようにキャッシュされるかを制御することができます。

- 自動許可されている
- 保護されていない
- 保護されている

重要： RFC 2068 に準拠したこれらの設定の変更の影響がよくわからない場合は、デフォルトの設定を使用することをお勧めします。デフォルトの設定を変更する予定がある場合は、ユーザがアイドル タイムアウトを追跡するためにセッションを確立した後で、保護されていないページにアクセスすると、SiteMinder セッション cookie が更新されることに注意してください。したがって、保護されていないページを、HTTP ヘッダをキャッシュするプロキシのキャッシュ対象とすることは望ましくありません。

プロキシによるキャッシュを防ぐために、以下の特性が設定ヘッダに適用されます。

- エージェントのアクティビティにかかわらず、すべてのリダイレクトで Cache-Control: no-cache ヘッダが設定されます。
- Web サーバは、使用されている HTTP プロトコル（1.0 または 1.1 以上）に基づいて、プロキシ/クライアントに適切なヘッダを送り返します。

cache-control: private や cache-control: max-age=60 のような複数のヘッダを使用することに対応して、あらゆるパラメータは、複数の値を表す文字列を使用して設定する必要があります。

次に、新しい設定について説明します。

1. ProxyHeadersDefaultTime - デフォルトは 60 秒です。
2. ProxyHeadersTimeoutPercentage - デフォルトは 10% です。
3. 以下の Cache-Control ヘッダを使用できます。

   ProxyHeadersAutoAuth

Web エージェント設定の ExpireForProxy パラメータが yes に設定されている場合、Web エージェントがクライアントへの HTTP レスポンスに挿入する HTTP 1.1 ヘッダの値を指定します。このヘッダの値によって、自動許可されたリソースがキャッシュされるかどうか、またはキャッシュされる期間が決定します。

デフォルト: Expires: Thu, 01 Dec 1994 16:00:00 GMT

例(推奨される設定): "Cache-control: max-age=60"

ProxyHeadersAutoAuth10

Web エージェント設定の ExpireForProxy パラメータが yes に設定されている場合、Web エージェントがクライアントへの HTTP レスポンスに挿入する HTTP 1.0 ヘッダの値を指定します。このヘッダの値によって、自動許可されたリソースがキャッシュされるかどうか、またはキャッシュされる期間が決定します。

デフォルト: Expires: Thu, 01 Dec 1994 16:00:00 GMT

例(推奨される設定): "Expires: Thu, 01 Dec 1994 16:00:00 GMT"

ProxyHeadersProtected

Web エージェント設定の ExpireForProxy パラメータが yes に設定されている場合、Web エージェントがクライアントへの HTTP レスポンスに挿入する HTTP 1.1 ヘッダの値を指定します。このヘッダの値によって、保護されているリソースがキャッシュされるかどうか、またはキャッシュされる期間が決定します。

デフォルト: Expires: Thu, 01 Dec 1994 16:00:00 GMT

Cache-Control: no-cache

例(推奨される設定): "Cache-Control: private"

ProxyHeadersProtected="Cache-Control: max-age=60"

ProxyHeadersProtected10

Web エージェント設定の ExpireForProxy パラメータが yes に設定されている場合、Web エージェントがクライアントへの HTTP レスポンスに挿入する HTTP 1.0 ヘッダの値を指定します。このヘッダの値によって、保護されているリソースがキャッシュされるかどうか、またはキャッシュされる期間が決定します。

デフォルト: Expires: Thu, 01 Dec 1994 16:00:00 GMT

Cache-Control: no-cache

例(推奨される設定): "Expires: Thu, 01 Dec 1994 16:00:00 GMT"

ProxyHeadersUnprotected

Web エージェント設定の ExpireForProxy パラメータが yes に設定されている場合、Web エージェントがクライアントへの HTTP レスポンスに挿入する HTTP 1.1 ヘッダの値を指定します。このヘッダの値によって、保護されていないリソースがキャッシュされるかどうか、またはキャッシュされる期間が決定します。

デフォルト: Expires: Thu, 01 Dec 1994 16:00:00 GMT

Cache-Control: no-cache

例(推奨される設定): ProxyHeadersUnprotected="Cache-Control: private"

ProxyHeadersUnprotected="Cache-Control: max-age=60"

ProxyHeadersUnprotected10

Web エージェント設定の ExpireForProxy パラメータが yes に設定されている場合、Web エージェントがクライアントへの HTTP レスポンスに挿入する HTTP 1.0 ヘッダの値を指定します。このヘッダの値によって、保護されていないリソースがキャッシュされるかどうか、またはキャッシュされる期間が決定します。

デフォルト: Expires: Thu, 01 Dec 1994 16:00:00 GMT

Cache-Control: no-cache

例(推奨される設定): "Expires: Thu, 01 Dec 1994 16:00:00 GMT"

複数のヘッダを設定する場合（たとえば、保護されていない HTTP/1.1 コンテンツに対して、cache-control ヘッダを推奨設定値にする場合）、以下のことに注意してください。

- 設定パラメータを複数の箇所で記述する*必要があります*。また、それらの値をカンマ（,）またはプラス記号（+）で区切ることはできません。
- これらの設定パラメータの値は HTTP レスポンス ヘッダの値なので、RFC 2616（HTTP/1.1 用）、RFC 1945（HTTP/1.0 用）、および RFC 822 に準拠する必要があります。HTTP/1.1 と HTTP/1.0 の両方が HTTP ヘッダの形式を RFC 822 メッセージの形式として指定します。つまり、「Name: Value」となります（Name の後にコロン、スペース、値が続きます）。

保護されていないリソースにユーザがアクセスした場合に、適切なキャッシュ期限ヘッダを設定するよう Web エージェントが設定されていない場合、Web エージェントはデフォルトではそれらのヘッダを設定しません。したがって、Web ブラウザまたはプロキシ サーバが SMSESSION cookie をキャッシュすることが許可されます。このキャッシュされた cookie は、ユーザが他のセッション（別のユーザ コンテキスト）を開始すると、Web ブラウザまたはプロキシ サーバによって再利用されるため、許可されていないインパーソネーション（偽装）が発生します。

詳細情報:

プロキシ サーバの背後にあるエージェントの設定 (P. 213)

プロキシ ヘッダの使用に関する注意事項

- Web エージェントがどのプロキシ ヘッダも送信しないように設定するには、ProxyHeadersUnprotected の値を空白にします。以下に例を示します。

  ProxyHeadersUnprotected=""

  注: 二重引用符（"）を表示するには、引用符（'）を使用します。Web エージェントは、引用符を自動的に二重引用符へ変換します。

- %% または %d という値（同じものとして取り扱われます）は、ProxyHeaders 行の中で記述することができます。この値は、IdleTimeout および SessionTimeout のうち小さいものに対して、ProxyHeadersTimeoutPercentage をかけた値へ置き換えられます。タイムアウトが設定されていない場合、ProxyHeadersDefaultTime が使用されます。

- 標準的なヘッダ（1.1 およびそれ以降）と、HTTP 1.0 ヘッダーのそれぞれに関して、バックエンド サーバへのリクエストを想定して、これらの値が正しく設定されていることを確認してください。
- ExpireForProxy="YES" は、クエリ文字列の中で SMSESSION cookie を渡す、cookie プロバイダによるリダイレクトを期限切れにします。

セキュリティの考慮事項

ブラウザセッションは、ログアウトの後も持続します。そのため、SMSESSION cookie を削除した場合も、ユーザが同じブラウザセッションを使用して、既にキャッシュされたファイルを表示する作業を防止することはありません。この問題が発生する原因は、以下のとおりです。プロキシ サーバはログアウト要求を意識せず、保護されたコンテンツ/保護されていないコンテンツのすべてが、タイムアウトになる（cache-control: max-age=60）までは、cache-control: private ユーザ用にそれらをキャッシュの中にとどめます。したがって、1 つのページ内でそのようなリクエストを実行した場合、有効な SMSESSION cookie が返されます。セキュリティを保証する唯一の方法は、キープ アライブを無効にすること、またはそのブラウザを閉じることです。

さらに、ローカルブラウザキャッシュは、private/max-age の組み合わせから影響を受けます。そのキャッシュは、複数のセッションにわたってローカルキャッシュを参照するからです。この理由で、保護されたリソースに関連する max-age の時間は、できるだけ短くする必要があります。

allowcacheheaders="FALSE" 設定ヘッダが使用されている（デフォルト）状況で、if-modified-since と if-none-match の各要求ヘッダを利用する場合、プロキシ サーバがこれらのヘッダを参照することは妨げられません。したがって、プロキシ サーバによるリクエストに対して、これらの参照されたヘッダが有効になります。

以下をインストールすることにより、この課題を回避することもできます。

- プロキシ サーバ上に Web エージェント。
- リクエストからこれらのヘッダを削除する他のフィルタ。

HTTP 1.0 と、HTTP 1.1 またはそれ以降は、キャッシングを行うプロキシに対して命令を指定する目的で、互いに異なるヘッダを使用します。そのため、接続のタイプに基づいて最適な取り扱いを保証するために、これらのバージョンを 1 つの方法で設定する必要があります。

第 16 章: リバース プロキシ サーバの設定

このセクションには、以下のトピックが含まれています。

リバース プロキシ ソリューションのタイプ

SiteMinder は、以下のリバース プロキシ ソリューションをサポートしています。

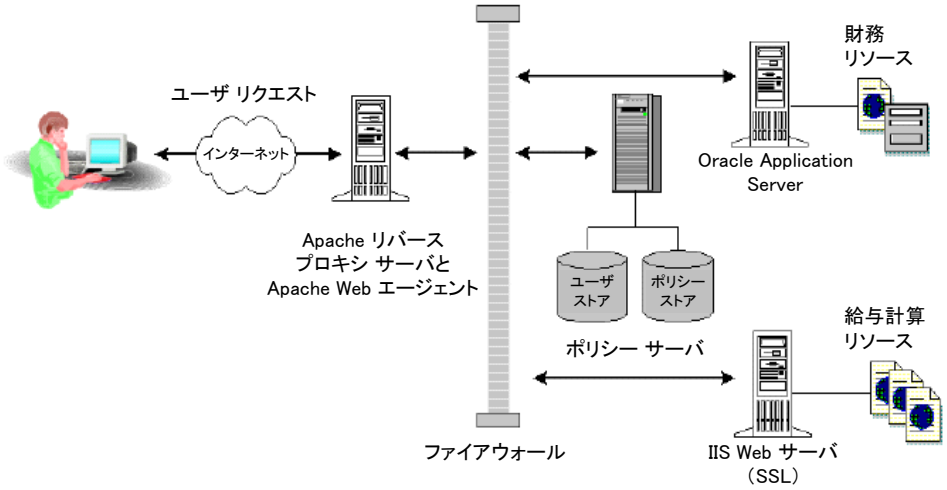
- Apache ベースのリバース プロキシ エージェント
- Oracle iPlanet ベースのリバース プロキシ エージェント
- SiteMinder セキュア プロキシ サーバ

SiteMinder でのリバース プロキシ サーバの機能

Apache または Oracle iPlanet Web サーバがリバース プロキシ サーバとして機能するように設定することができます。リバース プロキシ サーバは、企業の代理として、組織の内部ネットワークに要求を転送する機能を持つプロキシ サーバです。このプロキシ サーバにより、クライアントはバックエンド サーバ上にあるリソースにアクセスできるようになります。バックエンド サーバとは、ファイアウォールの背後にあるサーバのことです。

バックエンド サーバへのゲートウェイとして Apache または Oracle iPlanet リバース プロキシ サーバを使用する環境であれば、SiteMinder Web エージェントはこれらのリソースを保護できます。したがって、リソースが、バックエンドの SiteMinder Web エージェントによって保護されていないリソースも保護できるようになります。また、リソースはイントラネットや許可されたインターネットのユーザに対しても安全です。

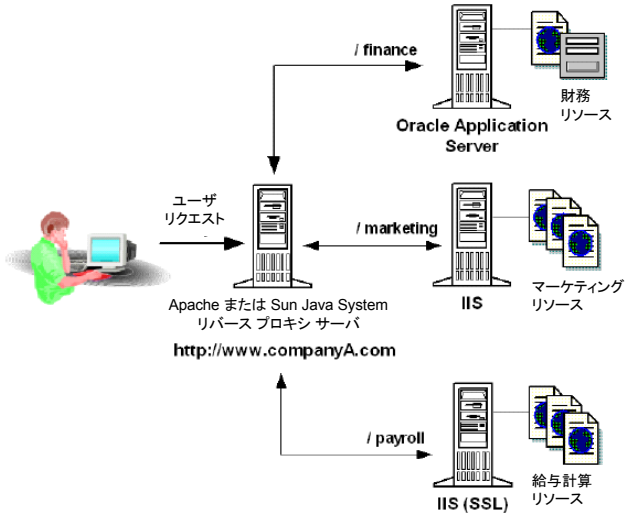
以下の図に、リバース プロキシ サーバを使用したネットワークを示します。



リバース プロキシ サーバを使用すると次の利点があります。

- cookie ドメイン内のユーザは、シームレスにバックエンド サーバ上のリソースにアクセスできます。他のドメインのユーザは、リバース プロキシ サーバの認証(通常はファイアウォールの認証も)を受けないと、それらの同じバックエンド サーバにアクセスできません。
- ユーザは、同じドメイン名を使用して、いくつかのバックエンド サーバ上でホストされている別のリソースにアクセスできます。
- リバース プロキシ エージェントは、Apache または Oracle iPlanet Web サーバに対して他の SiteMinder Web エージェントと同じ機能をサポートします。
- SiteMinder エージェントを利用できないサーバ上にリソースが存在している場合、Apache または Oracle iPlanet リバース プロキシ サーバをそのサーバの手前に配置することができます。これにより、Apache または Oracle iPlanet サーバの Web エージェントによってバックエンドのすべてのリソースが保護されます。

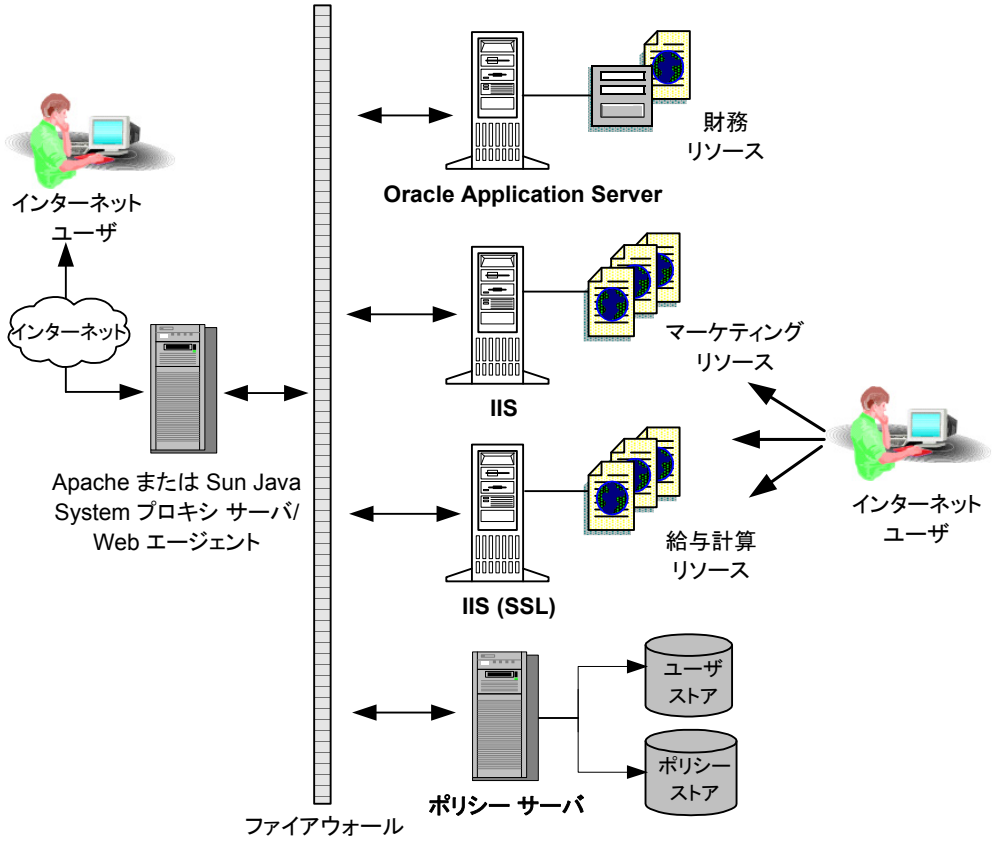
以下の図は、リバース プロキシ サーバがユーザからの要求をバックエンド サーバ上の適切なリソースにどのように転送するかを示したものです。



SiteMinder リバース プロキシ展開の考慮事項

通常は、Apache または Oracle iPlanet リバース プロキシ エージェントを展開する場合、Apache または Oracle iPlanet Web エージェントと保護対象リソースをホストするサーバとの間にファイアウォールが存在します。また、ポリシー サーバもファイアウォールの背後に配置する必要があります。

以下の図に、SiteMinder リバース プロキシの展開を示します。



SiteMinder リバース プロキシ エージェントを展開する際は、以下の点を考慮してください。

- ポリシーがレスポンス属性を返すように設定されている場合、それらのレスポンス変数は、リバース プロキシ サーバと保護対象リソースが存在するバックエンドの Web サーバの両方に送信されます。保護対象リソースへの要求が発生すると、ポリシー サーバはまず、レスポンス属性(CGI または HTTP 変数)を Apache または Oracle iPlanet サーバ上のエージェントに送信します。次に、エージェントは受信したレスポンス属性をリクエスト内に挿入し、バックエンド サーバに送信します。
- バックエンド サーバや保護対象アプリケーションに独自の認証機能が備わっている場合、それらの認証機能は無効にしておく必要があります。バックエンド認証を無効にすると、SiteMinder の認証が優先されるようになります。

  重要: リバース プロキシのキャッシュを設定すると、SMSESSION cookie を含め、すべての cookie がキャッシュに格納されます。詳細については、Apache または Oracle iPlanet Web サーバのドキュメントを参照してください。

詳細情報

HTTPS ポートの定義 (P. 155)

Apache リバース プロキシ サーバを設定する方法

SiteMinder で Apache ベースのリバース プロキシ サーバを設定するには、以下の手順に従います。

1. Apache リバース プロキシ サーバの *背後に*ある任意の Apache Web エージェントに関して、以下のパラメータの設定を更新します。
   a. ProxyAgent の値を yes に設定し、このエージェントがリバース プロキシ エージェントとして機能することを指定します。
   b. ProxyTimeout パラメータを、秒単位で何らかの値に設定します。

      リバース プロキシはこの値を使用して、背後に展開された Web エージェントへの要求をタイム アウトにします。
   c. リストから以下の値をすべて削除することにより、BadURLChars パラメータを編集します。

      %
   d. （オプション）ProxyTrust パラメータを有効にします。

      このパラメータを yes に設定すると、プロキシ エージェントの背後にある Web エージェントに対して、プロキシ エージェントから送信されるセッション情報を信頼し、改めて検証しないよう指示します。このパラメータを有効にすると通信の効率が向上します。これは、プロキシ エージェントからプロキシ サーバの呼び出しが 1 回のみになるためです。プロキシの背後にあるエージェントは、ポリシー サーバにアクセスする必要がありません。
   e. SSL 用にセットアップされたポートを Apache サーバに示すために、httpsports パラメータを設定します。
2. Apache Web サーバの httpd.conf ファイルに以下のディレクティブを追加します。

   ProxyPass

   リモート サーバからローカル サーバへのマッピングを許可します。このディレクティブの値は以下の形式を使用します。

   /local_virtual_path partial_URL_of_remote_server

   例: ProxyPass /realma/ http://server.example.org/realma/

   ProxyPassReverse

   HTTP リダイレクト レスポンス上での Apache サーバによるロケーション ヘッダの調整を許可します。このディレクティブの値は以下の形式を使用します。

/local_virtual_path partial_URL_of_remote_server

例: ProxyPassReverse /realma/ http://server.example.org/realma/

注: ディレクティブの詳細については、使用している Web サーバのマニュアルを参照してください。

3. Apache Web サーバを再起動します。

Sun Java System 6.0 リバース プロキシ サーバの設定

SiteMinder で Oracle iPlanet 6.0 Web サーバをリバース プロキシとして使用することができます。

注: SiteMinder エージェント設定ウィザードは、Oracle iPlanet（以前の Sun Java System）Web サーバ上のデフォルトの obj.conf ファイルのみを変更します。SiteMinder で他のインスタンスまたはリバース プロキシ展開を保護するには、デフォルトの obj.conf ファイルから、対応する <instance_name>-obj.conf ファイルに SiteMinder 設定をコピーします。たとえば、Web サーバのインストール時に obj.conf ファイルが作成されましたが、その後 my_server.example.com という名前のサーバ インスタンスを追加したとします。SiteMinder で my_server.example.com 上のリソースを保護するには、obj.conf ファイルから my_server.example.com-obj.conf ファイルに、ウィザードによって追加された SiteMinder 設定をコピーします。

Oracle iPlanet Web サーバをリバース プロキシとして設定する方法

1. 以下のディレクトリに移動します。

   web_server_installation_directory/plugins

2. 以下のサブディレクトリを作成します。

   ¥passthrough¥bin

3. 以下のいずれかのファイルを passthrough¥bin ディレクトリに追加します。

   - passthrough.dll（Windows）
   - libpassthrough.so（UNIX）

   注: Sun Java System Web Server 6.1 のリバース プロキシのアドオンを Sun Microsystems からダウンロードします。

4. 以下のいずれかのセクションを *web_server_installation_directory*/config/magnus.conf ファイルに追加します。

   Windows

   ```
   Init fn="load-modules"
   shlib="web_server_installation_directory/plugins/passthrough/bin/passthrough.dll"
   funcs="init-passthrough,auth-passthrough,check-passthrough,service-passthrough" NativeThread="no"

   Init fn="init-passthrough"
   ```

   UNIX

   ```
   Init fn="load-modules"
   shlib="web_server_installation_directory/plugins/passthrough/bin/libpassthrough.so"
   funcs="init-passthrough,auth-passthrough,check-passthrough,service-passthrough" NativeThread="no"

   Init fn="init-passthrough"
   ```

   注: すべての行は Init で始まる*必要があります*。続くどの設定も*同じ*行に入力する必要があります。

5. 以下のセクションを config/*instance_name*-obj.conf ファイルの先頭に追加することにより、パススルー オブジェクトを作成します。

   ```
   <Object name="passthrough">
   ObjectType fn="force-type" type="magnus-internal/passthrough"
   Service type="magnus-internal/passthrough" fn="service-passthrough"
   servers="http://server_name:port"
   Error reason="Bad Gateway" fn="send-error" uri="$docroot/badgateway.html"
   </Object>
   ```

6. 以下のようなセクションを server_install_directory/config/obj.conf ファイル内のデフォルト オブジェクトの先頭に追加することにより、転送する URI を設定します。

   ```
   NameTrans fn="assign-name" from="(/uri|/uri/*)" name="passthrough"
   ```

   URI は、リモート サーバ上で展開された Web アプリケーションのコンテキスト ルートです。また、パススルーは、以下の例に示される、obj.conf ファイルの <Object> の名前に相当します。

   ```
   <Object name="default">
   ...
   NameTrans fn="assign-name" from="(/webapp1|/webapp1/*)" name="passthrough"
   ...
   </Object>
   ```

名前の値は、手順 5 で使用したオブジェクト名の値と一致させる必要があります。

7. Web サーバを再起動します。

   リバース プロキシが設定されます。

Sun Java System 7.0 リバース プロキシ サーバの設定

SiteMinder で Oracle iPlanet 7.0 Web サーバをリバース プロキシとして使用することができます。

注: SiteMinder エージェント設定ウィザードは、Oracle iPlanet（以前の Sun Java System）Web サーバ上のデフォルトの obj.conf ファイルのみを変更します。SiteMinder で他のインスタンスまたはリバース プロキシ展開を保護するには、デフォルトの obj.conf ファイルから、対応する <instance_name>-obj.conf ファイルに SiteMinder 設定をコピーします。たとえば、Web サーバのインストール時に obj.conf ファイルが作成されましたが、その後 my_server.example.com という名前のサーバ インスタンスを追加したとします。SiteMinder で my_server.example.com 上のリソースを保護するには、obj.conf ファイルから my_server.example.com-obj.conf ファイルに、ウィザードによって追加された SiteMinder 設定をコピーします。

Oracle iPlanet Web サーバをリバース プロキシとして設定する方法

1. 以下のディレクティブを *instance_name*-obj.conf ファイルに追加します。

   NameTrans

   以下の形式を使用して、ローカルおよびリモートの仮想パスを指定します。

   NameTrans fn="map" from="*local_virtual_path*" name="reverse-proxy-/local_virtual_path" to="*remote_virtual_path*"

   例: NameTrans fn="map" from="/realma" name="reverse-proxy-/reamla" to="http://realma"

2. 以下のディレクティブを obj.conf ファイルの末尾に追加します。

   オブジェクト名

   以下の形式を使用して、NameTrans ディレクティブ内で使用されるローカルの仮想パスの名前とリモートの仮想パスの URL を指定します。

   <Object name="reverse-proxy-/*local_virtual_path*">

   Route fn="set-origin-server" server="http://*remote_server_URL*:*port*"

   </Object>

   例: <Object name="reverse-proxy-/reamla">

   Route fn="set-origin-server" server="http://server.example.org:*port*"

   </Object>

   Object ppath

   クライアントからサーバに与えられた部分的なパスを指定します。

   例: <Object ppath="http:*">

   Service fn="proxy-retrieve" method="*"

   </Object>

3. Web サーバを再起動します。

   リバース プロキシが設定されます。

SiteMinder セキュア プロキシサーバ

より高度なリバース プロキシ ソリューションを必要とする場合は、SiteMinder セキュア プロキシ サーバを使用できます。これには、Apache または Oracle iPlanet ベースの SiteMinder リバース プロキシ エージェントに比べて以下のような利点があります。

- Web サーバが組み込まれ完全にサポートされています。これには、SSL アクセラレータ カードのサポートや、キーと証明書の管理を行う GUI ツールなどが含まれます。
- 複数のセッション方式のサポート（cookie ベースと cookie なし）
- 以下のような、柔軟なプロキシ ルールのサポート。
  - URL に加えて、HTTP ヘッダと SiteMinder レスポンスに基づくルールをサポートしています。
  - 複雑なルールを簡単に取り扱うことができます。

セキュア プロキシ サーバにより、従来の SiteMinder アーキテクチャに新しいレイヤが導入されました。このレイヤはすべての要求を企業内の宛先サーバに転送またはリダイレクトします。

セキュア プロキシ サーバが要求を処理する際、ユーザが要求した URL は SM_PROXYREQUEST という HTTP ヘッダ変数内に保持されます。セキュア プロキシ サーバが要求をプロキシする前の、ユーザが要求したオリジナルの URL を必要とする他のアプリケーションは、このヘッダを使用できます。

セキュア プロキシ サーバによる SiteMinder 処理用の SM_PROXYREQUEST HTTP ヘッダ

セキュア プロキシ サーバにより、従来の SiteMinder アーキテクチャに新しいレイヤが導入されました。このレイヤはすべての要求を企業内の宛先サーバに転送またはリダイレクトします。

セキュア プロキシ サーバが要求を処理する際、ユーザが要求した URL は SM_PROXYREQUEST という HTTP ヘッダ変数内に保持されます。セキュア プロキシ サーバが要求をプロキシする前の、ユーザが要求したオリジナルの URL を必要とする他のアプリケーションは、このヘッダを使用できます。

第 17 章: 着信 URL の処理の制御

このセクションには、以下のトピックが含まれています。

URL 内のクエリ データのデコード

ポリシー サーバを呼び出す前に、Web エージェントの Base64 アルゴリズムが URL のクエリ データをデコードするように設定する(それによってポリシー サーバは適切なリソースを参照します)には、以下のパラメータを使用します。

DecodeQueryData

ポリシー サーバをコールする前に、Web エージェントが URL 内のクエリ データをデコードするかどうかを指定します。環境内で以下のタスクのいずれかを実行する必要がある場合は、このパラメータを yes に設定します。

- 正しい文字列に対してルール ファイラが機能していることを保証する必要がある場合
- クエリ文字列内のデータに対して書き込みルールが機能していることを保証する必要がある場合

デフォルト: No

ポリシー サーバをコールする前に、Web エージェントが URL のクエリ データをデコードするように設定するには、DecodeQueryData パラメータの値を yes に設定します。

URL 内のクエリ データの無視

IgnoreQueryData パラメータは、Web エージェントが URL を取り扱う方法に影響を及ぼします。Web エージェントが URL 全体をキャッシュに格納せず、ルール処理のためにクエリ文字列と一緒に URI をポリシー サーバに送信するように設定すると、パフォーマンスが向上します。この場合は、以下のパラメータを使用します。

IgnoreQueryData

Web エージェントが URL 全体（クエリ文字列を含む）をキャッシュに保管し、ルール処理のために URI 全体をポリシー サーバに送信するかどうかを指定します。 完全な URL 文字列には、以下の例に示すように、URI、フック（?）、およびクエリ データが含まれます。

URI?*query_data*

デフォルトでは、リクエストの対象となった URL がキャッシュに保管されます。後続のリクエストでは、一致する URL がキャッシュで検索されます。リクエスト内で URI が同一でもクエリ データが異なると、一致は失敗します。クエリ データを無視すると、パフォーマンスが向上します。

IgnoreQueryData パラメータが yes の場合は、以下の処理が発生します。

- URL はフックの箇所で切り捨てられます。 URI だけがキャッシュされ、ポリシー サーバへ送信されます。クエリ データは、リダイレクトの適正な状態を維持するために、他の場所で維持されます。
- フック（?）の前にある部分のみがポリシー サーバに送信されて、ルールの処理が行われます。
- 以下の例に示す 2 つの URI は、同一のリソースとして処理されます。

  /myapp?data=1

  /myapp?data=2

IgnoreQueryData パラメータが no の場合は、以下の処理が発生します。

- その場合、URL 全体がキャッシュされます。
- URI 全体がポリシー サーバに送信されてルールの処理が行われます。
- 以下の例に示す URI は、異なるリソースとして処理されます。

  /myapp?data=1

/myapp?data=2

デフォルト: No

Web エージェントが、処理のためにポリシー サーバに URI のみを送信するようにするには、IgnoreQueryData パラメータの値を yes に設定します。

重要: URL クエリ データに依存するポリシーがある場合は、この設定を有効にしないでください。

リダイレクト URL のクエリ文字列暗号化

Web エージェントは、FCC および SCC、パスワード サービス アプリケーション（CGI または JSP）、または cookie プロバイダなどの認証情報コレクタと通信する場合、リダイレクト URL 内に平文で示されるプロトコル パラメータを使用します。

Web エージェントは、リダイレクト URL 内のすべての SiteMinder クエリ パラメータを暗号化して、エージェント対話の保護を強化できるようになりました。Web エージェントが暗号化するのは、SiteMinder のコンポーネント間で送信されるデータのみで、SiteMinder 以外のアプリケーションへのリダイレクトは対象外です。

クエリ文字列の暗号化が有効な場合、Web エージェントは、ブラウザに 302 リダイレクト レスポンスを返すときにクエリ データを暗号化します。302 レスポンスにより、ユーザは別の SiteMinder リソースにリダイレクトされます。

すべてのクエリ パラメータは smquerydata という単一のクエリ パラメータにまとめられます。SecureUrls パラメータが有効な場合、SiteMinder は、必要に応じて、smquerydata パラメータが暗号化されていないすべてのリクエストに対するアクセスを拒否します。

以下のパラメータのいずれかが有効な場合、SecureUrls 機能はサポートされません。

FCCCompatMode

4.x の Web エージェントまたはサード パーティのアプリケーションによって保護されているリソースに対してフォームを提供するよう FCC/NTC を有効にします。

注: SMUSRMSG は、FccCompatMode が yes に設定されている場合のみ、カスタム認証方式でサポートされます。
デフォルト:（従来のエージェント）Yes

デフォルト:（フレームワーク エージェント）No

重要: このパラメータを no に設定すると、Netscape ブラウザのバージョン 4.x のサポートが削除されます。

LegacyEncoding

Web エージェントで、レガシー URL 内のすべてのドル記号（$）文字を強制的にハイフン（-）に置換します。これにより、MSR、パスワード サービス、および DMS に対する下位互換性も保証されます。このパラメータを no に設定すると、Web エージェントは文字列の SM を -SM- に変換します。このパラメータを yes に設定すると、Web エージェントはドル記号（$）文字を変換しません。

デフォルト:（フレームワーク エージェント） No

デフォルト:（従来のエージェント） Yes

SecureUrls パラメータが yes に設定されている場合は、前述のパラメータの値が yes に設定されていても、Web エージェントはそれらの値を無視します。この場合、これらのパラメータの値は、以下の例に示すように、設定オブジェクトまたは設定ファイル内での設定にかかわらず、エージェント ログでは no になります。

```
[12/Jul/2005:05:23:57-975-1-0] SecureUrls: 'YES'
[12/Jul/2005:05:23:57-975-1-0] FccCompatMode: 'NO'
[12/Jul/2005:05:23:57-975-1-0] LegacyEncoding: 'NO'
```

リダイレクト URL のクエリ文字列暗号化と認証情報コレクタ

認証情報コレクタを使用して、リダイレクト URL のクエリ文字列を暗号化する場合、認証情報コレクタは、クエリ データの暗号化に使用するキーを提供します。

フォーム認証方式の場合、クエリ文字列ディレクティブ smquerydata は FCC テンプレートの一部です。FCC を提供する Web エージェントは、FCC がポスティングされると、このディレクティブを使用して、暗号化されたクエリ データを目的の Web エージェントに送信します。

使用されるディレクティブは、以下のとおりです。

```
<INPUT type='hidden' name='smquerydata' value='$$smquerydata$$'>
```

注: カスタム FCC を使用している場合、他の FCC ディレクティブ（TARGET など）と共に、smquerydata ディレクティブをカスタム FCC に追加する必要があります。

SiteMinder r12.0 SP3 Web エージェントは、SecureUrls パラメータが有効である場合、この機能をサポートする他の Web エージェントから提供される認証情報コレクタとのみ連携できます。この機能は 5.x QMR 7 で導入されました。

リダイレクト URL のクエリ文字列暗号化と FCC ベースのパスワード サービス

リダイレクト URL のクエリ文字列を暗号化する場合は、FCC ベースのパスワード サービスのみを使用できます。CGI ベースおよび JSP ベースのパスワード サービスは、暗号化クエリ パラメータでは使用できません。SecureUrls パラメータを no に設定した場合、3 つのパスワード サービス バージョンをどれでも使用できます。

注: CGI および JSP パスワード サービスは 5.x QMR 7 以降はいずれ廃止予定の機能とされていますが、現在もサポートされています。

リダイレクト URL 内のクエリ文字列パラメータの暗号化

以下のパラメータにより、Web エージェントはリダイレクト URL 内のすべての SiteMinder クエリ パラメータを暗号化できます。

SecureURLs

Web エージェントがリダイレクト URL 内の SiteMinder クエリ パラメータを暗号化するかどうかを指定します。この設定を使用して、高度な認証方式であるパスワード サービスによって保護されている要求されたリソースのセキュリティを強化したり、要求が cookie プロバイダを呼び出すときのセキュリティを強化したりすることができます。

重要: Web エージェントは、SiteMinder コンポーネント間で送信されたデータを暗号化するだけです。リダイレクトのために SiteMinder 以外のアプリケーションに送信されるデータは暗号化されません。

以下の SiteMinder 認証情報コレクタおよびアプリケーションは SecureUrls 機能をサポートします。

- HTML フォーム認証
- 証明書およびフォーム認証
- SSL 認証
- 証明書またはフォーム認証
- NTLM 認証
- ACE 認証
- SafeWord 認証
- ユーザによる自己登録
- cookie プロバイダによるマルチドメイン シングル サインオン
- FCC ベースのパスワード サービス(CGI または JSP ベースではない)

デフォルト: No

リダイレクト URL 内のクエリ文字列パラメータを暗号化する方法

1. SecureURLs パラメータの値を yes に設定します。
2. シングル サインオン環境内のリダイレクト URL 内のクエリ文字列パラメータを暗号化する場合、シングル サインオン環境内のすべての Web エージェントの SecureURL パラメータが同じ値に設定されていることを確認します。
3. カスタム FCC を使用している場合、他の FCC ディレクティブ（TARGET など）と共に、smquerydata ディレクティブをカスタム FCC に追加します。

   クエリ文字列パラメータは SiteMinder リダイレクト URL 内で暗号化されます。

URI への無制限のアクセスの許可

SiteMinder で保護しない URI がある場合は、以下のパラメータを設定することによって、それらの URI への無制限のアクセスを無視して許可するように Web エージェントに命令できます。

IgnoreUrl

保護されていない URL 内の URI を指定します。URI と関連付けられたリソースにアクセスしようとしたユーザは、認証を要求されません。Web エージェントは、スラッシュが 3 つ登場した後で、それ以降の URI 部分を無視します。たとえば、このパラメータを以下の値に設定したとします。

```
http://www.example.com/directory
```

Web エージェントは以下の URI を無視します。

```
directory
```

指定された URI が別のドメインにあっても、出現場所にかかわらず Web エージェントはそれを無視します。たとえば、Web エージェントは、以下の URL のすべてに事前に表示された URI を無視します。

```
http://www.example.com/directory
http://www.example.net/directory
http://www.example.org/directory
```

注: この値では、大文字と小文字が区別されます。

デフォルト: デフォルトなし

例:（ローカル設定ファイル内の複数の URI）

IgnoreUrl="http://www.example.com/directory"

IgnoreUrl="http://www.example.com/directory2"

例:（ドメインを指定せずに URI のみを使用）

IgnoreUrl="/*resource*/"

URI への無制限のアクセスを許可するには、以下のタスクのいずれかを実行します。

- 中央設定では、無視する URI を持つ完全修飾ドメイン名をエージェント設定オブジェクトに追加します。URI が複数ある場合は、パラメータに複数値の設定を使用します。
- ローカル設定では、ローカル設定ファイルで完全修飾ドメイン名と URI ごとに個別の行を追加します。

指定された URI を使用するリソースは、Web エージェントによって無視され、それらのリソースへのアクセスが自動的に付与されます。

URL の最大サイズの設定

以下のパラメータを使用して、Web エージェントが処理できる最大 URL サイズを増加することができます。

MaxUrlSize

Web エージェントが処理できる URL の最大サイズ（バイト単位）を指定します。Web サーバによって、URL の長さ制限は異なるため、このパラメータを設定する前に Web サーバ ベンダーのマニュアルを確認してください。

デフォルト: 4096 B

最大 URL サイズを変更するには、MaxUrlSize パラメータに指定されたバイトの数を変更します。

第 18 章: URL 監視によるセキュリティの適用

このセクションには、以下のトピックが含まれています。

URL 監視の概要

Web エージェントには、Web サイトの正常な運用を中断させようとしたり、サイトのセキュリティ機構を回避して情報に不正にアクセスしようとしたりするユーザからの攻撃に対する防備機能が備わっています。

Web エージェントでは、リソースリクエストの URL を監視して、対象となるリソースのセキュリティポリシーを実行します。SiteMinder Web エージェントが URL を解釈および解析する方法は、リソースが置かれている Web サーバの方法と異なります。このような違いがあるため、パフォーマンスが微妙に異なり、セキュリティの問題が発生する可能性があります。また、場合によっては無許可のユーザがリソースにアクセスできることもあります。Web サイトの設計および SiteMinder Web エージェントの設定では、これらの問題を考慮する必要があります。

保護されていないリソースのファイル拡張子を無視することによるオーバーヘッドの削減

以下のパラメータを使用して、特定のタイプのリソースの要求を無視するように Web エージェントに指示することにより、SiteMinder のオーバーヘッドを縮小できます。

IgnoreExt

Web エージェントが SiteMinder ポリシーを確認せずに Web サーバに要求を渡すリソースのタイプを指定します。SiteMinder ポリシーによって保護されるレルムにアイテムが存在する場合でも、Web エージェントはこのパラメータによって指定されたそのアイテムへのアクセスを許可します。

以下の条件のどちらかを満たすリソースに対する要求を無視することができます。

- リソースが、Web エージェントに対して無視するよう指定した拡張子で終わっている場合。
- 保護されているリソースを表す URI にピリオド(.)が 1 つだけ含まれている場合。

  たとえば、要求されたリソースの URI が /my.dir/ である場合、Web エージェントは要求を直接 Web サーバへ渡します。

デフォルト: .class、.gif、.jpg、.jpeg、.png、.fcc、.scc、.sfcc、.ccc、.ntc

重要: IgnoreExt パラメータを設定する場合は注意してください。セキュリティの問題には、検討が必要なものがいくつかあります。

デフォルトでは、エージェントは、スラッシュ(/)で区切られた複数のピリオドを含むリソースに対する要求を*無視しません*。Web エージェントは、以下の例に示された手順に従って、リソースの要求を処理します。

1. 拡張子 .gif が IgnoreExt パラメータに追加されます。拡張子が .gif のリソースの要求は、Web エージェントによって無視されます。
2. 要求は以下の URI に対して行われます。

   /dir1/app.pl/file1.gif,
3. 一部の Web サーバが file1.gif リソースにサービスを提供する代わりにアプリケーションとして /dir1/app.pl を実行するので、Web エージェントはポリシーサーバに対して /dir1/app.pl/file1.gif をチェックします。

Web サーバに問い合わせずに /dir1/app.pl/file1.gif へのアクセスを付与することにより、セキュリティ違反が発生した可能性があります。

保護されていないリソースのファイル拡張子を無視することによってオーバーヘッドを削減するには、IgnoreExt パラメータの値に無視するリソースの拡張子を追加します。

期間や拡張のないリソースを保護する方法

サーブレットなど、期間のない URL があります。拡張のない URL もあります。これらの状況は両方ともセキュリティリスクをもたらします。以下の手順で、これらのリスクを実証します。

1. 使用している環境には、保護されたリソースである、/mydir/servlets というディレクトリが含まれています。
2. Web エージェントは拡張子が .gif のリソースに対する要求を無視するように設定されています。
3. 不正なユーザが、以下の例に示されるように URL の最後に拡張子 .gif と共に架空のファイルの名前を追加します。

   /mydir/servlets/file.gif
4. Web エージェントは拡張子 .gif を無視し、不正なユーザに /mydir/servelets ディレクトリへのアクセス権を与えます。

セキュリティのリスク回避が最優先事項である場合、エージェントがどの拡張子も無視できないようにしてください。その際、次のような結果が生じることを考慮してください。

- Web エージェントがページ上にあるイメージの URL をすべて評価するので、パフォーマンスが低下することがあります。
- 以前は認証が不要だったリソースに関してユーザが認証を要求されることがあるため、Web サイトの動作が変わることがあります。

期間がない URL を保護するには、以下のオプションがあります。

- OverrideIgnoreExtFilter 機能を使用するよう、エージェントを設定します。
- 保護されているリソースに、Web エージェントが無視するよう設定されている拡張子が付いていないことを確認します。

拡張子のないリソースの保護

不正なユーザが拡張子のないリソースへのアクセスを取得するのを防ぐために、以下のパラメータを使用することができます。

OverrideIgnoreExtFilter

Web エージェントがすべての URI と比較する目的で使用するために、複数の文字列から成る 1 つのリストを指定します。これは、通常は拡張子が Web エージェントによって無視されるリソース、または拡張子のないファイルやアプリケーションの保護に役立ちます。URI がリスト内の文字列の 1 つと一致する場合、Web エージェントはポリシー サーバへの問い合わせを行い、そのリソースが保護されているかどうかを決定します。

パスを厳密に指定するのではなく一般的な文字列を指定することをお勧めします。一群のリソースを保護するために部分的な文字列を含めることもできます。たとえば、指定された文字列が /servlet/ の場合、以下のリソースが保護されます。

- /dira/app1/servlet/app
- /dirb/servlet/app1
- /dirc/mydir/servlet/app2

デフォルト: デフォルトなし

拡張子のないリソースを保護するには、OverrideIgnoreExtFilter パラメータの値に保護するリソース(期間のない)の文字列を追加します。エージェント設定オブジェクトを使用している場合は、文字列を追加するために複数値オプションを使用します。ローカル設定ファイルを使用している場合は、各文字列を個別の行に追加します。

アプリケーションのセキュリティ保護

無許可のユーザは、Web エージェントが無視するように設定されている拡張子を含む虚偽のファイル名を URL の最後に追加することができます。 追加すると、無許可のユーザがそのリソースにアクセスできるようになります。そのような試行へのアクセスを Web エージェントに拒否させるには、以下のパラメータを使用します。

SecureApps

エージェントが、権限のないユーザからの URL を許可することを防ぎます。Web エージェントが、特定の拡張子で終わるファイルに対するリクエストを無視するように設定されている場合は、偽の URL を作成してリソースにアクセスしようとする攻撃を受ける可能性があります。

たとえば、以下の URL を持つリソースがあるとします。

```
/scripts/myapp
```

以下の例のような偽の URL を作成して、アクセス権を取得しようとする攻撃を受ける可能性があります。

```
/scripts/myapp/junk.jpg
```

SecureApps パラメータの値が no の場合に、Web エージェントが .jpg ファイルのリクエストを無視するように設定されていると、/scripts/myapp/junk.jpg のリクエストは自動的に許可されます。

SecureApps パラメータの値が yes の場合は、Web エージェントは、リソースが正当であるか、URL が偽であるかの検出を試みます。

デフォルト: No

アプリケーションを保護するには、SecureApps パラメータの値を yes に設定します。

複雑な URI の処理

DisableDotDotRule パラメータは、Web エージェントが、スラッシュ (/) で区切られた 2 つのドットを含む URI を自動的に許可するかどうかを判別します。

DisableDotDotRule が yes に設定されている場合、エージェントは二重ドット ルールを適用しません。たとえば、以下の URI を考えます。

- /dir1/app.pl/file1.gif

  Web エージェントは、IgnoreExt パラメータを使用して、リソースを自動許可するかどうかを判別します。

- /dir1/okay.button.gif

  エージェントはこの URI を無視できます。これは、2 つのドットはスラッシュ (/) で区切られていないためです。二重ドット ルールは、この場合は適用されません。

DisableDotDotRule が no に設定されていると (デフォルト)、Web エージェントは二重ドット ルールを*適用します*。Web エージェントは以下の URI に対する要求の認証を要求し、要求をポリシー サーバに渡します。

- /dir1/app.pl/file1.gif

  この URI は二重ドット ルールに当てはまります。これは、2 つのドットはスラッシュで区切られているためです。

  Web サーバは、/dir1/app.pl をターゲット リソースと見なし、/file1.gif を追加のパス情報と見なすことができます (一般に、このパス情報は、CGI ヘッダで PATH_INFO として表示可能です)。

- /dir1/okay.button.gif

  エージェントはこの URI を無視することができます。これは、二重ドット ルールは実施されているのに、2 つのドットがスラッシュ (/) で区切られていないため、ルールが適用されないからです。

重要: 許可されていないアクセスが発生することのないように、IgnoreExt パラメータおよび DisableDotDotRule パラメータは併用しないでください。たとえば、/dir1/app.pl の保護が必要であるにもかかわらず、DisableDotDotRule パラメータを yes に設定すると、エージェントは URI /dir1/app.pl/file1.gif を無視します。これは、二重ドット ルールを無効にし、.gif を IgnoreExt パラメータに含めたためです。その結果、許可されていないユーザが保護されたアプリケーション /dir1/app.pl にアクセスできるようになります。

無効な URL 文字の指定

URL リクエストの一部として使用できない一連の文字を指定することができます。それらの文字は、エージェントにより、無効な URL 文字として扱われます。このリスト内で指定されている文字または文字列を含む URL リクエストは、Web エージェントによって拒否されます。URL のうち、「?」文字より前の部分に対して、このチェックが実施されます。悪意のある Web クライアントがそのような文字を使用して SiteMinder ルールを回避する場合があるので、Web エージェントはそのような文字が含まれる URL リクエストを拒否します。

Web エージェントが無効な URL 文字を含んでいる URL リクエストを拒否する場合、Web サーバは以下のメッセージのいずれかで応答します。

- 内部サーバ エラー
- Web ページが見つからないエラー（404）

エージェントが要求をどのように処理するかについては、Web エージェント ログを確認してください。

以下のパラメータを使用して文字を指定します。

BadUrlChars

URL リクエストに使用できない文字シーケンスを指定します。Web エージェントは、このパラメータよって指定された文字シーケンスに対して、「?」文字の前にある URL 内の文字を確認します。指定された文字のいずれかが見つかった場合、Web エージェントは要求を拒否します。

以下の文字を指定できます。

- 円記号（¥）
- ダブルスラッシュ（//）
- ピリオドとスラッシュ（./）
- スラッシュとピリオド（/.）
- スラッシュとアスタリスク（/*）
- アスタリスクとピリオド（*.）
- ティルダ（~）
- %2d
- %20
- %00-%1f

- %7f-%ff
- %25

複数の値はカンマで区切ります。スペースは*使用しない*でください。

無効な URL 文字は、その前に疑問符(?)が付いている場合にのみ、CGI パラメータの中で使用できます。

デフォルト: <、>、&、;

制限:

- 実際に文字を指定するか、その文字の URL エンコード形式を入力することができます。たとえば、文字 a を入力するか、または、そのエンコード値である %61 を入力できます。
- 最大 4096 文字まで指定できます(区切り文字として使用するカンマを含みます)。
- また、文字の範囲をハイフンで区切って指定することもできます。構文は *starting_character-ending_character* です。たとえば、一連の文字として「a-z」と入力できます。
- 引用符(")を URL エンコード値 %22 でを指定します。ASCII は使用できません。

無効な URL 文字を指定するには、BadURLChars パラメータの値をブロックする文字が含まれるように編集します。

注: Apache 2.0 リバース プロキシ サーバおよび Outlook Web Access(OWA)を設定する際は、必ず BadURLChars パラメータをオフにしてください。 OWA では、電子メール件名中の文字が、BadURLChars パラメータでリストされていたとしても、無制限に許可されてしまうからです。

詳細情報

リバース プロキシ ソリューションのタイプ (P. 221)

IIS 6.0 サーバと BadURLChars 設定

IIS 6.0 Web サーバの SiteMinder Web エージェントは、ISAPI 拡張として機能します。HTTP リクエストを受信すると、最初に IIS 6.0 Web サーバは Web エージェントに転送する前に必ずリクエストを処理します。

IIS 6.0 Web サーバは、URL の無効な文字をフィルタリングしてから、リクエストを Web エージェントに渡すことができます。サーバは URI を、HTML ページまたは CGI アプリケーションのような物理 Web リソースにマッピングします。そのため、マッピング処理中に URI が変更された場合、Web エージェントは元の URI の特定の文字を表示できないことがあります。Web エージェントは、Web サーバから渡されたリソースに対してのみ動作します。このことは BadURLChars パラメータに文字を含める際に考慮する必要があります。

注: 一部の文字をフィルタリングした後、IIS 6.0 Web サーバは、リクエストを Web エージェントに渡す代わりに、エラー ページを返すことがあります。

無効なフォーム文字の指定

クロスサイト スクリプティング攻撃では一般に以下の文字が使用されます。

- 左山形かっこと右山形かっこ(< と >)
- アンパサンド(&)
- 引用符(")

認証チャレンジ中にユーザにフォームを提示するためにスクリプティング くコードを使用する場合、前述の文字が HTML フォームに送信される前に、これらの文字をリテラル HTML としてエンコードするよう以下のパラメータを使用して Web エージェントを設定します。

BadFormChars

フォーム上で出力として使用する前に、Web エージェントがリテラル HTML 文字としてエンコードする文字を指定します。ディレクティブの置換文字列のみが未処理の HTML としてエンコードされます。フォーム テンプレート(login.fcc テンプレートなど)のソース行は変更されません。ソース行を変更せずに維持することで、スクリプティング コードを含む動的なデータがフォーム内のデータとしてブラウザに戻されることを防ぎます。

デフォルト: 無効(リテラル エンコーディングなし)

例: <、>、&、%22

制限:

- 次の文字のみが許可されます: <、>、&、%22
- 実際に文字を指定するか、その文字の URL エンコード形式を入力することができます。たとえば、文字 a を入力するか、または、そのエンコード値である %61 を入力できます。
- 最大 4096 文字まで指定できます(区切り文字として使用するカンマを含みます)。
- また、文字の範囲をハイフンで区切って指定することもできます。構文は *starting_character-ending_character* です。たとえば、一連の文字として「a-z」と入力できます。
- 引用符(")を URL エンコード値 %22 でを指定します。ASCII は使用できません。

BadFormChars パラメータを設定する方法

1. BadFormChars パラメータを先頭の # 文字を削除することによって有効にします。

   前述の文字がすべてを含まれた BadFormChars パラメータが有効になります。

2. (任意)使用しないすべての文字をリストから削除します。残りの文字がカンマで互いに区切られていることを確認します。

無効なクエリ文字の指定

URL のクエリ文字列部分に特定の文字を禁止するには、以下のパラメータを設定します。

BadQueryChars

Web エージェントによって、URL のクエリ文字列部分('?' の後)で禁止される文字を指定します。

デフォルト: 空(クエリ文字列に禁止される文字はありません)

制限:

- 実際に文字を指定するか、その文字の URL エンコード形式を入力することができます。たとえば、文字 a を入力するか、または、そのエンコード値である %61 を入力できます。
- 最大 4096 文字まで指定できます(区切り文字として使用するカンマを含みます)。
- また、文字の範囲をハイフンで区切って指定することもできます。構文は *starting_character-ending_character* です。たとえば、一連の文字として「a-z」と入力できます。
- 引用符(")を URL エンコード値 %22 でを指定します。ASCII は使用できません。

例: %25 は、クエリ内の URL エンコード文字をブロックします。

Web エージェントは、URL のクエリ文字列内の文字を BadQueryChars パラメータに定義された文字の ASCII 値と比較することにより、クエリ文字列に禁止された文字が含まれているかどうかを検索します。たとえば、以下のように処理されます。

1. BadQueryChars パラメータに、以下に示すパーセント記号（%）の URL エンコード値が含まれているとします。

   %25

2. Web エージェントは、以下のクエリ文字列が含まれる HTTP リクエストを受信します。

   xxx=%0d

3. Web エージェントでは前述の例にある URL を検査しますが、URL エンコード値はデコードしません。たとえば、Web エージェントによって前述の例（手順 2）がキャリッジ リターンではなくリテラル文字列 %0d として解釈されます。

4. Web エージェントは BadQueryChars パラメータの値を検索し、それらを ASCII 値に変換します。たとえば、手順 1 の %25 はパーセント記号（%）に変換されます。

5. Web エージェントは、URL 内の各文字を、BadQueryChars パラメータからデコードされた ASCII 値と比較します。

6. その結果、ASCII パーセント記号（%）が以下の両方の場所に存在するため、Web エージェントによってリクエストがブロックされます。

   - URL のクエリ文字列
   - BadQueryChars パラメータのデコードされた（ASCII）値

クエリ文字列から特定の文字をブロックするには、BadQueryChars パラメータの値にブロックする文字を含めます。

第 19 章: 攻撃の防止

このセクションには、以下のトピックが含まれています。

クロスサイト スクリプティングからの Web サイトの保護

クロスサイト スクリプティング（CSS）攻撃が発生するのは、ブラウザからの入力テキスト（通常は、ポストされたデータ、または URL 内のクエリ パラメータから得られたデータ）がブラウザによって表示される場合です。このとき、有効で実行可能なスクリプトを形成することが可能な文字を、フィルタ処理なしでブラウザ内で表示してしまうことが問題です。

疑いを抱いていないユーザに対して、攻撃用 URL を提示できます。ユーザがその URL をクリックした場合、アプリケーションはブラウザに対して表示を返すことがあります。その表示の中には、入力文字と共に、クエリ文字列の中に含まれている無効な文字に関するエラー メッセージがあります。ブラウザ上でこれらのパラメータに関する表示が実施された場合、望まれていなかったスクリプトがそのブラウザ上で実行される事態が発生する可能性があります。

たとえば、検索エンジンの Web ページでユーザが「news」と入力した場合、アプリケーションは通常、空白のフィールド、または以下のような応答を返すことがあります。

news の検索結果は以下のとおりです:

攻撃用 URL への応答として、ブラウザは次のような応答を受け取ることがあります。

news<script>BadProgram</script>

二重引用符が（"）が ASCII 文字として入力された場合、BadCSSChars パラメータはそれを解釈しません。二重引用符を無効なクロスサイト スクリプティング文字として含める場合は、ASCII 文字と同等の 16 進数値である %22 を入力します。以下に例を示します。

BadCSSChars="%22"

クロスサイト スクリプティングをチェックするための Web エージェントの設定

URL の中で、実行可能なスクリプトの一部になる可能性のある文字をチェックするよう Web エージェントに指示するには、CSSChecking パラメータを yes に設定します。このパラメータを有効にすることにより、Web エージェントはクエリ文字列を含め URL 全体をスキャンします。そして、次のようなデフォルト文字セットのエスケープバージョンと、エスケープされていないバージョンの両方を探します。

- 左山形かっこと右山形かっこ(< および >)
- 引用符(')

CSS のデフォルト文字セットの上書き

デフォルトのクロスサイト スクリプティング文字セットを上書きするには、BadCSSChars パラメータに関して、必要な文字セットを入力します。希望する文字すべてからなる文字列全体を入力してください。たとえば、BadCSSChars パラメータを <,> に設定した場合、Web エージェントは左山形かっこと右山形かっこのみを検索します。

文字セットに関する問題を検出すると、Web エージェントはアクセス拒否メッセージをユーザに返し、エージェント エラー ログに以下のメッセージを記録します。

> Caught Possible Cross Site Scripting Violation in URL. Exiting with HTTP 403 ACCESS FORBIDDEN.

一部のアプリケーションは、Web サーバ プラットフォームにかかわりなく、クエリ文字列の中で引用符を使用することを必要とします。たとえば、iNotes Web Access のような特定の Domino アプリケーションは、引用符を使用することを必要とします。

クエリ文字列の中で引用符を必要とするアプリケーションを使用するには、BadCssChars パラメータから引用符を削除してください。

このパラメータに何も変更を加えないままにした場合、Web エージェントはデフォルトの文字セットをチェックします。

注: クロスサイト スクリプティングの詳細については、「CERT Advisory」を参照してください。

HTTP 専用属性を備えた cookie での情報の保護

クロスサイト スクリプティング攻撃に対する保護に役立つように、Web エージェントに、次のパラメータを使用して、作成するすべての cookie に HTTP 専用属性を設定させることができます。

UseHTTPOnlyCookies

Web エージェントが作成する cookie で HTTP のみの属性を設定するように Web エージェントに指示します。Web エージェントが、ユーザのブラウザにこの属性を持つ cookie を返すと、その cookie の内容はスクリプトで読み取ることができなくなります。これは、cookie を最初に設定した Web サイトのスクリプトにも当てはまります。これにより、cookie 内の機密情報を、権限のないサード パーティがスクリプトを使用して読み取ることを防ぐことができます。

デフォルト: No

cookie 内の情報を保護するには、UseHTTPOnlyCookies パラメータの値を yes に設定します。

セキュリティ侵害を防止するための IP アドレスの比較

無許可のシステムでは、パケットを監視して cookie を不正に入手し、その cookie を使って別のシステムにアクセスすることができます。無許可のシステムによるセキュリティ侵害を防止するために、永続的な cookie と過渡的な cookie を使用する IP チェックを有効または無効にすることができます。

IP チェック機能により、Web エージェントは、最後のリクエストから受信した cookie に保存されている IP アドレスと現在のリクエストの IP アドレスとを比較して、その 2 つが一致するかどうかを確認できます。一致しない場合、Web エージェントはリクエストを拒否します。

IP チェックを実装する目的で使用される 2 つのパラメータは、PersistentIPCheck と TransientIPCheck です。これらを次のように設定します。

- PersistentCookies を有効にした場合、PersistentIPCheck を yes に設定します。
- PersistentCookies を有効にしなかった場合、TransientIPCheck を yes に設定します。

SiteMinder ID cookie は、IP チェックの影響を受けません。

DNS DOS 攻撃の防止

IP アドレスに誤りのある有効な HTTP リクエストを攻撃者が送信した場合、Web エージェントはその IP アドレスを完全修飾ドメイン名に解決しようとします。HTTP リクエストのボリュームが十分に大きい場合、サービス妨害状態が Web エージェントと場合によっては DNS サーバに影響を与える可能性があります。以下のパラメータは、Web エージェントが DNS 参照を実行するかどうかを制御します。

DisableDNSLookups

Web エージェントが DNS の検索を実行することを防ぎます。

DNS DOS 攻撃を防ぐには、DisableDNSLookups パラメータの値を yes に設定します。

重要: このパラメータの値を yes に設定したときは、cookie ベースの機能が適切に動作するように、完全修飾ドメイン名を使用する必要があります。

第 20 章: フォームによるユーザの認証

このセクションには、以下のトピックが含まれています。

認証情報コレクタが要求を処理する方法

Web エージェントと認証情報コレクタによって保護されているリソースに対してユーザがリクエストを行った場合、SiteMinder はそのリクエストを以下のように処理します。

注: このプロセスは、FCC、SFCC、SCC、および NTC のコレクタのみに当てはまります。シングル サインオンを目的とする cookie プロバイダには当てはまりません。

1. ユーザは、保護されたリソースへのリクエストを行います。
2. そのリソースを保護している Web エージェントはポリシー サーバへの問い合わせを行い、フォーム、高度な SSL、または Windows の認証方式により、そのリソースが保護されていることを検出します。
3. その Web エージェントは、そのユーザを適切な認証情報コレクタへリダイレクトします。また、その認証情報コレクタを表す URL に、ターゲット リソースと自らの暗号化済みエージェント名を含むクエリ データを追加します。
4. 以下のいずれかのイベントが発生します。
   - FCC はフォームを表示し、ユーザ認証情報を収集します。
   - 認証情報が利用できない場合、SFCC はフォームを表示し、ユーザ認証情報を収集します。
   - SCC はユーザ認証情報を収集します。
   - NTC はユーザの NT 認証情報を収集します。
5. 認証情報コレクタは、ユーザをポリシー サーバに直接ログインさせます。ついで、ポリシー サーバは 1 つのセッションを作成します。
6. 認証情報コレクタはセッション cookie をユーザのブラウザに書き込み、そのユーザを、元の Web エージェントへリダイレクトして返します。
7. Web エージェントはセッションを検証し、ユーザがそのリソースにアクセスすることを許可します。

r5.x、r6.x、および r12 の認証情報コレクタの動作は、4.x の認証情報コレクタとは異なります。4.x のエージェントとそれ以降のエージェントで構成される「混在環境」では、4.x の Web エージェントと通信できるように、r5.x、r6.x、または r12 の認証情報コレクタを設定する方法を検討する必要があります。

注: SSL 認証方式の詳細については、ポリシー サーバのマニュアルを参照してください。

認証とシングル サインオンを目的とした認証情報コレクタの使用方法

SiteMinder の認証情報コレクタは、Web エージェント内のアプリケーションです。特定のユーザ認証情報を収集してユーザを認証します。認証情報コレクタが収集する認証情報は、保護されたリソースからなる特定のグループに関して設定された認証方式のタイプを基礎としています。 認証情報コレクタは、フォーム、SSL、WIN の各認証方式と、複数の cookie ドメインにおけるシングル サインオンで使用されます。

使用可能な認証情報コレクタのタイプは、以下のとおりです。

フォーム認証情報コレクタ（FCC）

認証を要求する際にユーザに対して表示される HTML フォームに基づいて、認証情報を収集します。FCC が表示するフォームは、ファイル拡張 .fcc で終わるテンプレートに基づいています。たとえば、Web エージェントと共に、login.fcc というフォームがインストールされます。このフォームをカスタマイズし、ログインの目的で使用することができます。このファイルは、標準の HTML タグ、および SiteMinder が必要とする固有の表記を使用して作成されます。

注: FCC ベースの認証を使用している場合、認証情報が空のままでフォームが提示されると、フレームワーク Web エージェントはリクエストを処理せず、最初に要求された URL にリダイレクトします。このため、フレームワーク Web エージェントは、どの通信もポリシー サーバに送信しません。トラディショナル Web エージェントの場合、要求は処理され、ポリシー サーバへ送信されて、OnAuthAttempt イベントが生成されます。

SSL **認証情報コレクタ**（SCC）

SSL ベースの認証情報（SSL ベースの認証方式に必要な認証情報）を収集します。SSL 経由のベーシック方式、または X509 証明書とベーシックの組み合わせによる方式がこれに該当します。

注: SCC では、X.509 証明書とフォームの組み合わせによる方式や、X.509 証明書またはフォームの組み合わせによる方式は処理されません。「X.509 証明書とフォームの組み合わせによる方式」を処理するのは FCC であり、「X.509 証明書またはフォームによる方式」を処理するのは SFCC です。

cookie **プロバイダ**（CCC）

シングル サインオンのために複数の cookie ドメインで SiteMinder セッションを追跡します。他のタイプの認証情報コレクタとは異なり、cookie プロバイダは認証情報を収集しませんし、ユーザに対して認証を要求することもありません。cookie プロバイダは、認証情報を処理しますが、この場合、セッションは認証情報に基づいています。

デフォルト: SmMakeCookie.ccc

NTLM 認証情報コレクタ(NTC)

IIS Web サーバ上に保存されているリソースに関連する NT 認証情報を収集します。この認証情報は、Internet Explorer ブラウザからアクセスされます。この認証方式は、認証情報を要求する代わりに、ユーザの Windows NT ログイン名とパスワードを使用します。

SSL フォーム認証情報コレクタ(SFCC)

HTML フォームに基づいて認証情報を収集します(FCC と同様です)。ただし、SFCC は、X509 証明書またはフォームの認証方式の場合にのみその情報を収集します。

SFCC が表示するフォームは、.sfcc 拡張子で終わるテンプレートに基づいています。たとえば、Web エージェントと共に、login.sfcc というフォームがインストールされます。このフォームをカスタマイズし、ログイン フォームとして使用することができます。

認証情報コレクタの MIME タイプ

各認証情報コレクタに対して、1 つの MIME タイプが関連付けられています。その MIME タイプは、ユーザがリソースへのリクエストを行ったときに、どの認証情報コレクタが認証を要求するかを表しています。次の表は、各タイプを示しています。

認証情報コレクタ	MIME タイプ
フォーム認証情報コレクタ	.fcc
SSL 認証情報コレクタ	.scc
cookie プロバイダ	.ccc
NTLM 認証情報コレクタ	.ntc
SSL フォーム認証情報コレクタ	.sfcc

認証情報コレクタを使用する認証方式を設定する場合、または複数の cookie ドメイン間でシングル サインオンをセットアップする場合は、その認証方式またはそのシングル サインオンの設定によって参照されるファイル拡張子として、該当の MIME タイプが使用されます。たとえば、次のようになります。

- 複数の cookie ドメイン間でシングル サインオンを設定する場合、次のような 1 つの URL を入力して、cookie プロバイダを識別します。

  http://myserver.company.com:80/siteminderagent/SmMakeCookie.ccc

  SmMakeCookie.ccc は、デフォルトの cookie プロバイダの名前です。この名前を使用すること、または独自の名前を作成することができます。ただし、シングル サインオンを確立するには、拡張子として .ccc を使用する必要があります。

- Windows 認証の場合、この方式を有効にするデフォルトのターゲット ファイルは、以下のとおりです。

  /siteminderagent/ntlm/creds.ntc

  ここでも、正しい MIME タイプを拡張子としているファイルを使用する必要があります。

エージェントのインストール先である Web サーバ上に実際のファイルが存在していることを必要とする認証情報コレクタは、FCC と SFCC のみです。これらのコレクタは、フォームベースの認証方式を前提としています。ユーザに対して提示される HTML フォームを定義する上で、.fcc と .sfcc の各テンプレートが必要です。

各認証情報コレクタ用の MIME タイプの設定

認証情報コレクタとして動作する Web エージェントをインストールするときは、いくつかの設定手順に従う必要があります。その結果、コレクタが動作するようになります。この手順は、オペレーティング システムではなく、Web サーバ プラットフォームによって異なります。

IIS と Domino の各 Web サーバでのクレデンシャル コレクタのセットアップ

IIS と Domino の Web サーバでは、MIME タイプのマッピング（ファイル拡張子パラメータとして）が Web エージェント設定に必要になります。IIS および Domino Web サーバのクレデンシャル コレクタを設定する方法

1. 各クレデンシャル コレクタで使用する特定の MIME タイプをマップします。次の表に示すデフォルト値を使用することをお勧めします。

エージェント設定パラメータ	クレデンシャル コレクタ	MIME タイプ
CCCExt	cookie プロバイダ	.ccc
FCCExt	フォーム クレデンシャル コレクタ	.fcc
SCCExt	SSL クレデンシャル コレクタ	.scc
SFCCExt	SSL フォーム クレデンシャル コレクタ	.sfcc
NTCExt	NTLM クレデンシャル コレクタ	.ntc

注: デフォルトの拡張子を使用しない場合、またはデフォルト値が他の目的ですでに使用されている場合は、独自の拡張子を入力してください。Web エージェントはその値を優先します。たとえば、FCC に対応する FCCExt を .myext に設定し、その拡張子を使用するように FCC テンプレートの名前を変更して、たとえば login.myext にした場合、Web エージェントは .myext で終わる URL を、フォーム認証要求として認識します。

クレデンシャル コレクタが設定されます。

Sun Java System Web サーバでの認証情報コレクタのセットアップ

Windows または UNIX プラットフォーム上で動作する Oracle iPlanet Web サーバーの場合、Web エージェントをインストールする際に、MIME タイプが自動的にセットアップされます。他に必要な設定はありません。

Apache Web サーバでの認証情報コレクタのセットアップ

Windows または UNIX 上の Apache Web サーバでは、Web エージェントをインストールした*後*で httpd.conf ファイルに変更を加える必要があります。

特に、Web エージェントのインストール ロケーションと、フォーム テンプレートの配置先となる Web エージェントのサンプル ディレクトリを Web サーバに対して指示するエントリを Alias セクションに追加する必要があります。各 MIME タイプに関するエントリを AddHandler セクションに追加する必要もあります。

注: 詳細については、「SiteMinder Web エージェント インストール ガイド」を参照してください。

混在環境での認証情報コレクタの設定

5.x QMR 2 およびそれ以降では、フォーム（FCC/SFCC）、SSL（SCC）、および NTLM（NTC）の各認証情報コレクタは、4.x の認証情報コレクタとは異なる動作をします。ユーザが認証情報を発行した場合、認証情報コレクタはユーザのブラウザ内で認証情報 cookie を作成する必要も、そのユーザを元の Web エージェントへ送り返す必要もありません。代わりに、認証情報コレクタは、要求されたリソースを保護している Web エージェントの代わりに、そのユーザをポリシー サーバに直接ログインさせることができます。

注: cookie を設定するのではなく、認証情報コレクタを使用して直接ユーザをログインさせることを推奨します。認証情報コレクタを使用したユーザのログインの方が、ユーザの認証情報のセキュリティ保護が強化されます。これは、これらの認証情報が cookie に格納された状態でネットワーク上を転送されないためです。これは、混在環境で認証情報コレクタを設定する際の重要な考慮事項です。

認証情報コレクタはユーザをログインさせるために、要求されたリソースを保護している Web エージェントの名前と、ユーザが入力する認証情報を必要とします。

エージェント名を把握するために、認証情報コレクタは以下のプロセスを使用します。

1. SMAGENTNAME クエリ パラメータを使用します。これを、元の Web エージェントは、ユーザを認証情報コレクタへリダイレクトする際に URL のクエリ文字列に追加します。
2. URL に対してエージェント名が追加されていない場合、認証情報コレクタのエージェント設定ファイル、またはエージェント設定オブジェクトの中にある、エージェント名からホスト名へのマッピングからエージェント名を取り出して使用します。

   各マッピングは、コレクタを使用して自らのリソースを保護しているホストの名前と、その IP アドレスを指定します。マッピングは、AgentName パラメータ内で定義されています。
3. エージェント名マッピングが設定されていない場合は、エージェント名として、ターゲット URL の完全修飾ホスト名を使用します。完全修飾ホスト名は、エージェントの設定内で AgentNamesAreFQHostNames パラメータを有効にすることにより決定されます。

   このパラメータは、デフォルトで無効になっています。その場合、認証情報コレクタはエージェント名として、DefaultAgentName パラメータの値を使用します。

アップグレードを行う場合、認証情報コレクタのアルゴリズムと、4.x の Web エージェントとの通信を可能にする目的でそのアルゴリズムが FCC、SCC、SFCC、または NTC の設定にどのような影響を及ぼしているかを考慮する必要があります。

混在環境での FCC と NTC の使用

リクエストを処理する上で、FCC と NTC はユーザ認証情報、およびリクエストされたリソースを保護している Web エージェントの名前を必要とします。しかし、4.x の Web エージェントは、自らが送信する URL の一部としてエージェント名を渡すことはありません。FCC および NTC へのポストを行うサード パーティのエージェントも、同様にエージェント名を渡さない可能性があります。

FCC と NTC を 4.x の Web エージェントと組み合わせて使用する場合は、以下の設定オプションが役立ちます。

- 互換モードを使用する - 4.x の Web エージェントまたはサードパーティのアプリケーションによって保護されているリソースに対してフォームを提供するように r5.x、r6.x、または r12.0 SP3 の FCC/NTC を有効にするには、FCCCompatMode パラメータを有効にします。トラディショナル Web エージェントでは、FCCCompatMode パラメータはデフォルトで有効です。フレームワーク エージェントでは、FCCCompatMode パラメータはデフォルトで無効です。

  このパラメータを有効にすると、r5.x、r6.x、または r12.0 SP3 のエージェントは、フォームと NTLM の認証情報コレクションを、4.x のエージェントと同様の方法で扱うようになります。つまり、フォームまたは NTLM の認証情報 cookie がユーザのブラウザに書き込まれ、そのユーザはリダイレクトされてエージェントへ戻ってから、ポリシー サーバにログインできるようになります。この結果、Web エージェントの相互運用が可能になります。

  FCCCompatMode パラメータの値が no に設定されている場合、4.x のエージェントとの互換性は無効になります。r5.x、r6.x、または r12.0 SP3 環境の場合のみ、このパラメータの値を no に設定してください。

  重要: このパラメータを no に設定すると、Netscape ブラウザのバージョン 4.x のサポートが削除されます。

- エージェント名のマッピングを指定する - FCC のみ。後方互換性を無効にした（FCCCompatMode パラメータの値を no に設定した）場合は、AgentName パラメータを、その FCC を使用してリソースを保護している各ホストの名前と IP アドレスにマップします。これらのマッピングは、FCC のエージェント設定ファイル、またはポリシー サーバ上にある、その FCC に対応するエージェント設定オブジェクトの中でセットアップします。

  マッピングの例 :

  myagent, 123.1.12.1

  myagent, www.sitea.com

- ホスト名をエージェント名として使用する - FCC のみ。アルゴリズム内の最初の 2 つのオプションが適していない場合は、AgentNamesAreFQHostNames パラメータの値を yes に設定することができます。これは、ターゲット URL の中にある完全修飾ホスト名をエージェント名として使用するよう FCC に指示します。たとえば、URL 文字列の中に、次の内容が含まれているとします。

  url?A=1&Target=http://www.nete.com/index.html

  Target 文字列のうち、www.nete.com の部分がエージェント名として提供されます。

  デフォルトでは、このパラメータは no に設定されます。その結果、DefaultAgentName パラメータの値がエージェント名として使用されます。

以下の表に、r5.x、r6.x、または r12.0 SP3 と 4.x の FCC および NTC を設定する場合のガイドラインを示し、混在環境でのそれぞれがどのように動作するかについて説明します。

注:

- NTLM クレデンシャル コレクタは、IIS 以外の Web サーバから IIS Web サーバへユーザをリダイレクトすることができます。
- フレームワーク Web エージェントに関しては、FCC 互換モードが無効になっている状態に関する説明のみを参照してください。

リソースを保護する Web エージェント	FCC 互換モードでの r5.x、r6.x、または r12.0 SP3 の FCC	r5.x、r6.x、または r12.0 SP3 の FCC - FCC 互換モード無効
r5.x、r6.x、または r12.0 SP3	■ FCC は認証情報 cookie を発行します。 ■ 「証明書**および**フォームの組み合わせによる認証」は機能しません。 ■ 「証明書**または**フォームによる認証」は機能しません。	■ FCC は、セッション cookie を発行します。 ■ 「証明書**および**フォームの組み合わせによる認証」は機能します。 ■ 「証明書**または**フォームによる認証」は機能します。

リソースを保護する Web エージェント	4.x QMR 2/3/4 の FCC
4.x QMR 5 または 4.x QMR 6	■ エージェントは認証情報 cookie を発行します。 ■ 「証明書およびフォームの組み合わせによる認証」は機能しません。 ■ 「証明書またはフォームによる認証」は機能します。
r5.x、r6.x、または r12.0 SP3	■ エージェントは認証情報 cookie を発行します。 ■ 「証明書およびフォームの組み合わせによる認証」は機能しません。 ■ 「証明書またはフォームによる認証」は機能します。

注: SSL 認証方式の詳細については、ポリシー サーバのマニュアルを参照してください。

リソースを保護する Web エージェント	FCC 互換モードでの r5.x、r6.x、または r12.0 SP3 の FCC	r5.x、r6.x、または r12.0 SP3 の FCC - FCC 互換モード無効
4.x QMR 5 または 4.x QMR 6	■ NTC は認証情報 cookie を発行します。	■ NTC は、セッション cookie を発行します。
r5.x、r6.x、または r12.0 SP3	■ NTC は認証情報 cookie を発行します。	■ NTC は、セッション cookie を発行します。

リソースを保護する Web エージェント	4.x QMR 2/3/4 の NTC
4.x QMR 5、または 4.x QMR 6	■ エージェントは認証情報 cookie を発行します。
r5.x、r6.x、または r12.0 SP3	■ エージェントは認証情報 cookie を発行します。

混在環境での SCC の使用

4.x の Web エージェントと r5.x、r6.x、または r12.0 SP3 の SCC の相互運用を可能にするには、以下の手順のいずれかに従います。

- エージェント名のマッピングを指定する: AgentName パラメータを、SCC を使用してリソースを保護している各ホストの名前とその IP アドレスにマップします。これらのマッピングは、SCC のエージェント設定ファイル、またはポリシー サーバ上にある、その SCC に対応するエージェント設定オブジェクトの中でセットアップします。
- ホスト名をエージェント名として使用する: エージェント名のマッピングを指定しない場合は、AgentNamesAreFQHostNames パラメータを Yes に設定することができます。これは、ターゲット URL の中にある完全修飾ホスト名をエージェント名として使用するよう SCC に指示します。

  たとえば、次のような URL 文字列が発生したとします。

  url?A=1&Target=http://www.nete.com/index.html

  Target 文字列のうち、www.nete.com の部分がエージェント名として提供されます。

  デフォルトでは、このパラメータは no に設定されます。その結果、DefaultAgentName パラメータの値がエージェント名として使用されます。

以下の表に、混在環境で SCC として機能する 4.x のエージェントと r5.x、r6.x、または r12.0 SP3 のエージェントがどのように動作するかを示します。

Web エージェントのバージョン	4.x QMR 2/3/4 の SCC	r5.x、r6.x、または r12.0 SP3 の SCC
4.x QMR 5 または 4.x QMR 6	■ エージェントは SSL 認証情報 cookie を発行します。 ■ ブラウザから Web サーバに対する元の接続が SSL 経由であっても、リクエストをリダイレクトしない限り、証明書を収集することはできません。	■ AgentName パラメータでマッピングを作成するか、AgentNamesAreFQHostNames を Yes に設定します。 ■ SCC は、セッション cookie を発行します。 ■ ブラウザから Web サーバに対する元の接続が SSL 経由であっても、リクエストをリダイレクトしない限り、証明書を収集することはできません。

Web エージェントのバージョン	4.x QMR 2/3/4 の SCC	r5.x、r6.x、または r12.0 SP3 の SCC
r5.x、r6.x、または r12.0 SP3	■ エージェントは SSL 認証情報 cookie を発行します。 ■ リクエストをリダイレクトすることなく、証明書を収集できます。	■ SCC は、セッション cookie を発行します。 ■ リクエストをリダイレクトすることなく、証明書を収集できます。

注: SSL 認証方式の詳細については、ポリシー サーバのマニュアルを参照してください。

シングル リソース ターゲットを使用するための FCC の設定

テンプレート ファイル login.fcc の中で、ターゲットをハード コード化することにより、ユーザをシングル リソースへ振り向けるよう FCC を設定することもできます。

シングル リソース ターゲットを使用するように FCC を設定する方法

1. *agent_home*/Samples にある login.fcc ファイルを開きます。
2. @target=*target_resource* を FCC に追加します。
3. 次のエントリを追加します。

   @smagentname=*agent_name_protecting_resource*

   例: @smagentname=mywebagent
4. EncryptAgentName パラメータを no に設定します。エージェント名をファイル内でハードコード化した後、そのエージェント名を暗号化する手法は存在しないので、この設定が必要です。
5. この FCC を使用する他のすべてのエージェントに対して、EncryptAgentName を no に設定します。

注: 詳細については、ポリシー サーバドキュメントを参照してください。

フォーム キャッシュの利用によるパフォーマンスの向上

フォーム キャッシュは、フォーム テンプレート データを保管するための単一のリポジトリです。テンプレート データをキャッシュに保管することにより、エージェントは .fcc ファイルから同じデータを繰り返し読み取ることを回避できます。したがって、フォーム認証要求を処理する場合、フォーム キャッシュにより、エージェントは FCC スレッドが使用するデータを短時間でフェッチできるようになります。

FCC 拡張子を持つリソースへのアクセスが発生すると、対応するテンプレート ファイルが FCC によって読み取られて処理されます。Web エージェントはこのような要求を 1 秒間に数百件受け取ります。

フォーム キャッシュは、簡単に読み取ることができるメモリにフォーム テンプレート ファイルを保管することで FCC の処理を支援します。仮想メモリ アクセスはディスク アクセスよりも高速なので、FCC コンポーネントはフォームを短時間で処理でき、ホスト サーバにかかる負荷が軽減されます。処理時間の短縮により、FCC が各 Web サーバの要求に応える能力が向上し、フォーム認証を効率化することができます。

フォーム キャッシュに保管されるデータ

フォーム キャッシュに保管されるデータはフォーム テンプレート テキストから成り立ちます。このフォーム テンプレート テキストはデータ構造に事前に解析済みで、このデータ構造によって FCC による処理は効率化されます。

データ構造に含まれるものは、以下のとおりです。

- 国際化用のフォーム ロケール データ
- UTF-8 フォーマットの未処理のテキスト、テンプレート ディレクティブ情報、および要求環境からの置換用の関数/変数情報を含むデータ オブジェクトの順序付きリスト

キャッシュに保管する前に FCC ファイルから未処理のテキストを事前に処理することで、FCC は冗長な操作を削減し、処理を効率化することができます。データ オブジェクトの順序は、既存の FCC の機能デザインに従って維持する必要があります。デザイン上、ディレクティブ、関数、および変数は、常に FCC ファイルの上から下に順に処理されます。FCC は、さまざまな順序付きリストをどう処理するかを認識して、所定の FCC テンプレートの目的とされる機能を適切にサポートします。

フォーム キャッシュの設定

フォームは、パフォーマンスを改善し、不要なネットワークトラフィックを削減するためにキャッシュできます。以下のパラメータを使用して、フォーム キャッシュの設定を制御できます。

EnableFormCache

フォーム テンプレート キャッシュを制御します。このパラメータを yes に設定すると、フォーム認証のパフォーマンスが向上します。キャッシュを無効にするには、このパラメータを no に設定します。

デフォルト: yes

FormCacheTimeout

オブジェクトをキャッシュに保管しておくことのできる期間を秒数で指定します。この期間を過ぎると、そのオブジェクトは無効と見なされます。タイムアウト間隔を過ぎると、フォーム テンプレート ファイルの日時と、キャッシュ オブジェクトが作成された時刻が比較されます。キャッシュに保管されたオブジェクトがシステムのディスク上のファイルよりも新しい場合、タイムアウトはリセットされて、以下のタイムアウトまでオブジェクトは保管されます。それ以外の場合、オブジェクトはキャッシュから削除されます。

デフォルト: 600

フォーム キャッシュを設定する方法

1. EnableFormCache パラメータの値を yes に設定します。
2. フォーム キャッシュのタイムアウト間隔を変更する場合は、FormCacheTimeout の値を希望の秒数に設定します。

フォーム キャッシュが設定されます。

FCC レルム コンテキスト確認の無効化によるパフォーマンスの向上

フォーム認証時に、Web エージェントはポリシー サーバに対し IsProtected 呼び出しを行って、要求されたリソースが保護されているかどうかを判断します。この最初の呼び出しの後、Web エージェントは通常、ポリシー サーバに対する追加の IsProtected 呼び出しを行います。この 2 番目の呼び出しによってレルム コンテキストが確立され、Web エージェントは FCC を使用したユーザのログインを許可し、保護されているリソースにアクセスできるようにします。Web エージェントがこの追加呼び出しを行うかどうかは、以下のパラメータを使用して制御することができます。

FCCForceIsProtected

Web エージェントはポリシー サーバに対しもう一度 IsProtected 呼び出しを行うかどうかを指定します。この 2 回目の呼び出しによってレルム コンテキストが確立され、Web エージェントはユーザのログインを許可し、保護されているリソースにアクセスできるようにします。

このパラメータを no に設定すると、Web エージェントはポリシー サーバへの IsProtected の最初の呼び出しから取得されたレルム情報を代わりに使用します。

デフォルト: yes

FCC レルム コンテキスト確認を無効にすることによってパフォーマンスを向上させるには、FCCForceIsProtected パラメータの値を no に設定します。

認証情報コレクタのリダイレクトでの相対ターゲットの使用

リクエストを認証情報コレクタとターゲットリソースへ振り向ける際に、完全修飾 URL の代わりに相対 URI を使用するよう Web エージェントに指示することができます。相対 URI を使用すると、Web エージェントと共に他のシステム上に存在している認証情報コレクタがリクエストを処理することを防止できます。

注: この設定項目は、cookie 認証情報コレクタ(CCC)を*除く*、他のすべての認証情報コレクタに適用されます。CCC は、このパラメータの完全修飾ドメイン名を使用する必要があります。相対 URI を使用した場合、OnAuthAccept レスポンスは CCC で適切に動作しません。

通常、完全修飾 URL が認証情報コレクタの URL に付加されます。以下に例を示します。

url?A=1&Target=http://www.nete.com/index.html

相対 URI のみを使用するには、TargetAsRelativeURI パラメータを yes に設定します。yes に設定した場合、認証情報コレクタの URL に付加されるターゲット パラメータは相対ターゲット(url?A=1&Target=/index.html など)になります。その場合、認証情報コレクタがリダイレクトを行って、ターゲットリソースを保護している Web エージェントへ戻すときは、相対リダイレクトになります。また、Web エージェントは、スラッシュ(/)以外の文字で始まるすべてのターゲットを拒否します。

このパラメータのデフォルト値は no なので、常に完全修飾 URL が使用されます。

有効なターゲットドメインの定義

Web エージェントでは、悪意のある Web サイトにユーザをリダイレクトするフィッシング攻撃から保護するため、以下のパラメータを使用できます。

ValidTargetDomain

クレデンシャル コレクタがユーザをリダイレクトすることを許可されるドメインを指定します。URL 内のドメインがこのパラメータ内で設定されたドメインに一致しない場合は、リダイレクトが拒否されます。

デフォルト: デフォルトなし

このパラメータは、フォーム クレデンシャル コレクタ(FCC)を含むすべての高度な認証方式によってサポートされています。

処理の際、ValidTargetDomain パラメータはターゲットの有効なドメインを識別します。ユーザをリダイレクトする前に、Web エージェントはリダイレクト URL の値と、このパラメータ内のドメインを比較します。このパラメータがない場合、Web エージェントはユーザを任意のドメインのターゲットにリダイレクトします。

ValidTargetDomain パラメータには、有効なドメインごとに 1 つの値を設定し、複数の値を含めることもできます。

ローカル Web エージェント設定では、以下の例のように、ドメインごとに 1 行ずつ 1 つのエントリを指定します。

validtargetdomain=".xyzcompany.com"

validtargetdomain=".abccompany.com"

有効なフェデレーション ターゲットドメインの定義

SiteMinder が Federation セキュリティ サービス SP として動作している場合、SAML 2.0 トランザクション用に Identity Provider Discovery（IPD）プロファイルを設定できます。IPD によって、認証リクエストに対してどの IdP がアサーションを生成するかをユーザが選択できるようになります。

検出プロセスで、悪意のある Web サイトにユーザがリダイレクトされるのを防ぐことができます。認証リクエストを満たす IdP のドメインが検証されるよう Web エージェントを設定します。

検証プロセスを有効にするには、以下のパラメータの値を設定します。

ValidFedTargetDomain

（Federation のみ-SAML 2.0）Identity Provider Discovery を実装した場合に、フェデレーション環境の有効なドメインをすべてリスト表示します。

SiteMinder Identity Provider Discovery（IPD）サービスでリクエストを受信すると、リクエストの IPDTarget クエリ パラメータを調べます。このクエリ パラメータは、Discovery サービスでリクエストを処理した後にリダイレクトする URL をリスト表示します。IdP の場合、IPDTarget は SAML 2.0 シングル サインオン サービスです。SP の場合、ターゲットは共通ドメイン cookie を使用するリクエスト アプリケーションです。

フェデレーション Web サービスでは、IPDTarget URL のドメインを、ValidFedTargetDomain パラメータに指定されたドメインのリストと比較します。URL ドメインが ValidFedTargetDomain に設定されたドメインの 1 つと一致する場合、IPD サービスは IPDTarget パラメータに示された URL にユーザをリダイレクトします。このリダイレクトは SP の URL に対して行われます。

ドメインが一致しない場合、IPD サービスはユーザ リクエストを拒否し、ブラウザに 403 Forbidden が返されます。また、FWS トレース ログおよび affwebservices ログにエラーが報告されます。 これらのメッセージは、IPDTarget のドメインが有効なフェデレーション ターゲットドメインとして定義されないことを示します。

ValidFedTargetDomain を設定しない場合、検証は行われず、ユーザはターゲット URL にリダイレクトされます。

制限: フェデレーション ネットワーク内の有効なドメイン

デフォルト: デフォルトなし

ValidFedTargetDomain パラメータに有効なドメインを指定します。この設定は複数パラメータなので、複数のドメインを入力できます。

ローカル設定ファイルを変更している場合は、たとえば以下のように、ドメインを別々にリスト表示します。

validfedtargetdomain=".examplesite.com"

validfedtargetdomain=".abccompany.com"

Identity Provider Discovery プロファイルの詳細については、「Federation Security Services Guide」を参照してください。

FCC/SCC での完全修飾ホスト名としてのエージェント名の使用

AgentNamesAreFQHostNames パラメータを設定することで、フォームおよび SSL の各認証情報コレクタでターゲット URL の完全修飾ホスト名を Web エージェント名として使用できるようになります。たとえば、URL 文字列に以下が含まれているとします。

url?A=1&Target=http://www.nete.com/index.html

Target 文字列のうち、www.nete.com の部分が Web エージェント名として提供されます。

次の場合、認証情報コレクタはこのパラメータを使用します。

- ターゲット Web エージェントが URL にエージェント名を追加しなかった場合（これは、サード パーティのエージェントで発生することがあります）。
- AgentName パラメータ内で、エージェントからホスト名へのマッピングを設定しなかった場合。

AgentNamesAreFQHostNames が no に設定されている場合、認証情報コレクタは DefaultAgentName パラメータの値を、ターゲット Web エージェントの名前として使用します。

FCC と SCC で使用するためのエージェント ID と Web サーバのマッピング

AgentName パラメータとそれに関連付けられた IP アドレスによって、Web サーバ インターフェースと、ポリシー ストア内で定義済みのエージェント名の間のマッピングが実現されます。Web エージェントは、適用対象となるルールとポリシーの正しいセットを取得するために、適切なエージェント名のコンテキスト内でエージェント API 呼び出しを実行する必要があります。

Web エージェントが、フォームまたは SSL 認証情報コレクタとして機能している場合は、要求されたリソースを保護している Web エージェントの名前と、要求を処理するためのユーザの認証情報を必要とします。デフォルトでは、SiteMinder は、ユーザを Web エージェントから認証情報コレクタにリダイレクトする URL の中に、エージェント名を含めます。しかし、他のアプリケーションを組み合わせて使用している場合、AgentName パラメータを、そのコレクタを使用している各ホストの名前と IP アドレスにマップすることができます。認証情報コレクタは、そのマッピングを使用して、エージェント名を解決します。

次に、AgentName パラメータの値のサンプルを示します。

- myagent1,123.1.1.12
- myagent, www.sitea.com

フォームにポストされたデータの維持

SiteMinder 5.0 以降では、ポスト中にタイムアウトやその他の中断が発生しても、ユーザがフォームに入力したデータは保存されます。このことを「POST 維持」と呼びます。ユーザが認証情報を入力してフォームを送信した後、維持されたデータはアプリケーションに対して再ポストされます。

フォームの POST 維持を目的とした .fcc ファイルの変更

v5.0 より前のリリースの SiteMinder から認証情報を取得するフォームを用意することもできます。POST 時にデータを維持するには、ユーザがリソースにアクセスしようとしときに表示する .fcc フォーム テンプレートに以下の行を追加します。

```
<input type=hidden name=postpreservationdata
value="$$postpreservationdata$$">
```

注: POST 維持は、CGI および Perl ベースのパスワード サービス、ACE 認証、および FCC へのポストを行うカスタム認証方式のいずれでも *サポートされていません*。

フレームワーク エージェントと従来のエージェントの間の POST 維持の有効化

フレームワーク エージェントは POST 維持データを従来のエージェントとは異なる方法で処理します。SiteMinder 環境でフレームワーク エージェントと従来のエージェントが組み合わせて使用され、一方のタイプのエージェントによってホストされているリソースがもう一方のタイプのエージェントでホストされているフォーム認証情報コレクタ(FCC)によって保護される場合、以下のパラメータを備えた適切なテンプレート ファイルを指定する必要があります。

PostPreservationFile

以下の POST 維持テンプレート ファイルの *いずれか*に対するパスを指定することで、トラディショナル エージェントとフレームワーク エージェントとの間の POST 維持データの転送を有効にします。

- tr2fw.pptemplate - トラディショナル エージェントが稼働しているサーバでホストされているリソースが、フレームワーク エージェント上で実行されている FCC によって保護されていることを示します。
- fw2tr.pptemplate - フレームワーク エージェントが稼働しているサーバでホストされているリソースが、トラディショナル エージェント上で実行されている FCC によって保護されていることを示します。

デフォルト: デフォルトなし

例: *web_agent_home*/samples/forms/fw2tr.pptemplate

フレームワーク エージェントと従来のエージェントの間の POST 維持を有効にする方法

1. 別のタイプのエージェント上で実行されている FCC によって保護されるリソースを決定します。
   a. フレームワーク エージェントで実行されている FCC によって保護されるリソースをホストしている従来のエージェントのリストを作成します。
   b. 従来のエージェントで実行されている FCC によって保護されるリソースをホストしているフレームワーク エージェントのリストを作成します。
2. リソースをホストしている従来のエージェント(事前に手順 1a で列挙したもの)で、PostPreservationFile パラメータの値を tr2fw.pptemplate ファイルのパスに設定します。
3. リソースをホストしているフレームワーク エージェント(事前に手順 1b で列挙したもの)で、PostPreservationFile パラメータの値を fw2tr.pptemplate ファイルのパスに設定します。

4. 従来のエージェントと通信するフレームワーク Web エージェントのすべてで、以下のパラメータの値を yes に設定します。

   LegacyPostPreservationEncoding

   Web エージェントが POST 維持データをエンコードするときに、以前のトラディショナル Web エージェントと互換性のある方法を使用するか、新規のフレームワーク Web エージェントと互換性のある方法を使用するかを指定します。このパラメータの値が yes の場合は、トラディショナル Web エージェントと互換性のあるエンコーディングが使用されます。このパラメータの値が no の場合は、フレームワーク Web エージェントのみと互換性のあるエンコーディングが使用されます。

   デフォルト: No

5. ユーザのリソースをホストしている Web サーバを再起動します。

   フレームワーク エージェントと従来のエージェントの間の POST 維持が有効になります。

POST 維持の無効化

POST 維持を使用する必要がない場合は、以下のパラメータを使用してそれを無効にすることができます。

PreservePostData

要求をリダイレクトする場合に Web エージェントが POST データを維持するかどうかを指定します。 ユーザが、フォーム認証や証明書認証など、高度な認証を受ける場合、POST データは認証フェーズの間、維持されます。

デフォルト: yes

POST 維持を無効にするには、PreservePostData パラメータの値を no に設定します。

SafeWord フォーム認証に対する safeword.fcc ファイルの使用

ポリシー サーバは、SafeWord 認証サーバに基づいてユーザを認証することができます。この中には、SafeWord ハードウェア トークンを使用してログインするユーザも含まれます。

SafeWord フォーム ベースの認証方式を使用するための前提条件の 1 つは、SiteMinder Web エージェントのインストール先の Web サーバに、カスタマイズされた safeword.fcc ファイルが存在していることです。この Web サーバは、管理者が HTML フォーム認証を実装したのと同じ cookie ドメインの中に存在している必要があります。

safeword.fcc ファイルは、SafeWord 認証を実行する際にユーザが目にするフォームを定義します。ポリシー サーバが認証コレクタへ送信する認証コードの値によって、ユーザに対して入力を要求するフォームが変化します。safeword.fcc ファイルの中で、smauthreason ディレクティブが示しているように、各認証コードに対応するさまざまなテキストを観察することができます。

組織に合わせて safeword.fcc ファイルをカスタマイズするために、フォームの HTML レイアウトを変更することができます。しかし、ユーザが特定のフォームに対して入力する必要のある認証情報のタイプを変更することはできません。また、フォームのロゴを変更することもできます。このファイルは、ISO-8859-1 エンコードを使用しています。

サンプルの safeword.fcc ファイルは、以下のディレクトリ内に配置されています。

web_agent_home/Samples/Forms

注: 詳細については、ポリシー サーバ ドキュメントを参照してください。

Passport 認証用の特別なフォーム テンプレートの使用

Web エージェント 5.x QMR1 以降では、Passport 認証方式用の loginusername.fcc という FCC ファイルが用意されるようになりました。このフォームを使用するように SiteMinder が設定されている場合に、保護されているリソースをユーザが要求すると、SiteMinder は以下の処理を行います。

1. サインインした Passport ユーザを、SiteMinder ユーザ ディレクトリからマップされたユーザとして認識します。
2. 次のような特徴を備えたフォームを表示します。
   - マップ済みユーザのユーザ名を自動的に表示します。
   - 対応するパスワードの入力を要求します。

loginusername.fcc ファイルを使用するには、以下の手順に従います。

1. エージェントが無視する必要のある拡張子のリストから .fcc エントリを削除することにより、Web エージェントの IgnoreExt パラメータの値を編集します。
2. Passport（カスタム）認証方式を使用して、loginusername.fcc を保護します。

   注: 詳細については、ポリシー サーバ ドキュメントを参照してください。
3. Passport 認証方式によって保護されたレルムごとに、ポリシー サーバ上で 1 つのレスポンスを作成します。各レスポンスに対して、Web エージェントのレスポンス属性を以下のように設定します。
   a. ［属性］ドロップダウン リストから［WebAgent-HTTP-Header-Variable］を選択します。
   b. ［属性の種類］グループ ボックスで、［ユーザ属性］ラジオ ボタンを選択します。
   c. ［属性名］フィールドに、ユーザ名またはユーザ ID に対応するユーザ ディレクトリ属性の名前を入力します。たとえば、LDAP ディレクトリに、Passport 所有者にマップされるユーザが含まれている場合は、uid と入力します。
   d. ［変数名］フィールドに、LDAPUID のようなレスポンス変数の名前を入力します。

      注: 詳細については、ポリシー サーバ ドキュメントを参照してください。
4. ［変数名］の値を反映するように loginusername.fcc フォームを編集します。この例の設定では、変数名は LDAPUID です。

これらの高度な機能をエージェント設定ファイルまたはエージェント設定オブジェクトに追加することができます。

ACE 認証でフォームを使用する方法

ACE 認証方式と共に SiteMinder フォームを使用する場合は、以下のいずれかを実行します。

- 新しい認証方式を作成する場合は、以下のフォームをテンプレートとして使用します。

  web_agent_home¥samples_default¥forms¥smaceauth.fcc

- ACE を使用するように既存の認証方式を変更する場合は、forms.fcc ファイル内の他のディレクティブの下に以下のディレクティブを追加します。

  @smacefcc=1

小文字の URL プロトコルの指定

RFC 2396 に準拠しないレガシー アプリケーションをフォーム ベース認証方式で保護する場合、URL のプロトコル部分を小文字にする必要があれば、以下のパラメータを設定します。

LowerCaseProtocolSpecifier

リダイレクト URL のスキーム（プロトコル）部分で小文字のみを使用するかどうかを指定します。

デフォルト: No

例: http、https

お使いの環境で URL に小文字のプロトコルを指定するには、LowerCaseProtocolSpecifier パラメータの値を yes に設定します。

第 21 章: パスワード サービスの管理

このセクションには、以下のトピックが含まれています。

Web エージェントでパスワード サービスを使用するためのサポートされている方法

パスワード サービスをサポートするには、以下のいずれかの方法を使用することができます。

- FCC ベースの方法(POST 維持に必須)
- CGI ベースの方法(非推奨)

注: POST 維持は、CGI および Perl ベースのパスワード サービス、ACE 認証、および FCC へのポストを行うカスタム認証方式のいずれでも *サポートされていません*。

ここでは、FCC ベースの方法をお勧めします。CGI ベースのパスワード サービスは廃止される予定で、今後のリリースではサポートされません。

FCC パスワード サービスと URL クエリ暗号化

FCC パスワード サービス アプリケーションでは、URL 上のクエリ データを暗号化することで、エージェントの対話を保護することができます。クエリ データは、FCC パスワード サービスでのみ暗号化することができます。FCC パスワード サービスのファイルには以下のものがあります。

- smpwservices.fcc

  この FCC ファイルは Web エージェントと一緒にインストールされます。このファイルの場所は以下のとおりです。

  web_agent_home/samples/forms

  パスワード サービスの呼び出し時に、パスワード ポリシーが設定されていなかった場合、ポリシー サーバの SiteMinder 管理者は、環境変数 NETE_PWSERVICES_REDIRECT を smpwservices.fcc の相対パスに設定する必要があります。

  パスは以下のとおりです。

  /siteminderagent/forms/smpwservices.fcc

  新しい FCC では、FCC ディレクティブ authreason および username に基づいてパスワード サービスのフォームが表示されます。

- smpwservices.unauth

  このファイルは、パスワード サービス フォームの GET/POST アクション時に発生したエラーを処理します。

  このファイルは、POST 時に要求処理で障害が発生した場合に呼び出されるその他の FCC 無許可ファイルと同じです。この FCC は、空の TARGET 変数などのエラー状況を処理します。このエラー報告は、CGI ベースのパスワード サービスと同期させて、FCC POST によって生じるその他の不明なエラーを処理することを目的としています。

- smpwservicesUS-EN.properties

  この properties ファイルは、ユーザにわかりやすいメッセージをパスワード サービス フォームに表示するために smpwservices.fcc で使用されます。

  この properties ファイルにはユーザにわかりやすいメッセージが収められています。管理者は、パスワード サービスのフォームにどのメッセージを表示するかに応じてこのファイルを変更することができます。メッセージのフォーマットは name=value です。

FCC パスワード サービスの設定

FCC パスワード サービスの設定は、CGI バージョンや JSP バージョンとほとんど同じです。SiteMinder 管理者は、ユーザ ディレクトリまたはネームスペースに関連付けるパスワード ポリシーを設定する必要があります。ただし、必須の[リダイレクト URL]フィールドには、smpwservices.fcc の相対パスを設定する必要があります。このパスは以下のとおりです。

/siteminderagent/forms/smpwservices.fcc

リダイレクト パスを設定することにより、FCC パスワード サービスは正しく機能するようになります。

注: 詳細については、ポリシー サーバドキュメントを参照してください。

SiteMinder X.509 証明書および基本認証方式を使用する場合にユーザによるパスワード変更を有効にする方法

ユーザが必要に応じていつでも自分のパスワードを変更できるように SiteMinder の FCC パスワード サービス機能を設定できます。SiteMinder X.509 証明書および基本認証方式を使用している場合、HTTPS プロトコルで始まるパスワード変更 URL を使用します。

SiteMinder X.509 証明書および基本的な認証方式を使用して、ユーザによるパスワード変更を有効にするには、以下の手順に従います。

1. ユーザ ディレクトリにパスワード ポリシーをサポートする属性が含まれていることを確認します。
2. 管理 UI を使用して以下のタスクを実行します。
   a. FCC ベースのパスワード ポリシーを作成し、対象のリソースを保護します。
   b. 権限のあるユーザがパスワードを変更できるようにパスワード ポリシーを設定します。
3. 以下が含まれるパスワード変更 URL を FCC 形式で作成します。
   - HTTPS スキーム(プロトコル)
   - ログオン サーバの FQDN(例: http:*logonserver.example.com*)
   - FCC ベースのパスワード サービスの URI(例: siteminderagent/forms/smpwservices.fcc?)
   - SiteMinder エージェントの名前(SMAGENTNAME)
   - SiteMinder によって保護されるターゲット リソース(TARGET)。
4. 以下の例のように、1 つまたは複数の保護されていない Web ページにリンクとして URL を埋め込みます。

   ```
   <a
   href="https:logonserver.example.com/siteminderagent/forms/smpwservices.fcc?SMAUTHREASON=
   34&SMAGENTNAME=$$smencode(smagentname)$$&TARGET=$$smencode(target)$$">Change Password</font></a>
   ```

5. 以下の手順でパスワード変更をテストします。
   a. 手順 3 で作成したパスワード変更リンクがある Web ページを表示します。
   b. パスワード変更リンクをクリックします。

パスワード変更フォームが表示されます。

c. パスワード変更フォームに入力し、それをサブミットします。

パスワードの変更に成功した場合、確認ページが表示され、保護されているターゲットリソースへのリンクが表示されます。

d. リンクをクリックし、リソースが表示されることを確認します。

e. ブラウザを閉じて再度開きます。新しいパスワードを使用して、保護されているリソースへのアクセスを試みます。

新しいパスワードでリソースをアクセスできれば、パスワード変更は成功です。

FCC でのユーザによるパスワード変更を有効にする方法

ユーザが必要に応じていつでも自分のパスワードを変更できるように SiteMinder の FCC パスワード サービス機能を設定できます。

注: SiteMinder Web エージェント設定で SecureURLs パラメータの値も no に設定されている場合のみ、以下のプロセスを実行します。

FCC でのユーザによるパスワード変更を有効にするには、以下の手順に従います。

1. ユーザ ディレクトリにパスワード ポリシーをサポートする属性が含まれていることを確認します。
2. 管理 UI を使用して以下のタスクを実行します。
   a. FCC ベースのパスワード ポリシーを作成し、対象のリソースを保護します。
   b. 権限のあるユーザがパスワードを変更できるようにパスワード ポリシーを設定します。
3. 以下が含まれるパスワード変更 URL を FCC 形式で作成します。
   - ログオン サーバの FQDN（例: http:*logonserver.example.com*）
   - FCC ベースのパスワード サービスの URI（例: siteminderagent/forms/smpwservices.fcc?）
   - SiteMinder エージェントの名前（SMAGENTNAME）
   - SiteMinder によって保護されるターゲット リソース（TARGET）。
4. 以下の例のように、1 つまたは複数の保護されていない Web ページにリンクとして URL を埋め込みます。

```
<a
href="http:logonserver.example.com/siteminderagent/forms/smpwservices.fcc?SMA
UTHREASON=
34&SMAGENTNAME=$$smencode(smagentname)$$&TARGET=$$smencode(target)$$">Change
Password</font></a>
```

5. 以下の手順でパスワード変更をテストします。
   a. 手順 3 で作成したパスワード変更リンクがある Web ページを表示します。
   b. パスワード変更リンクをクリックします。

      パスワード変更フォームが表示されます。
   c. パスワード変更フォームに入力し、それをサブミットします。

パスワードの変更に成功した場合、確認ページが表示され、保護されているターゲット リソースへのリンクが表示されます。

d. リンクをクリックし、リソースが表示されることを確認します。

e. ブラウザを閉じて再度開きます。新しいパスワードを使用して、保護されているリソースへのアクセスを試みます。

新しいパスワードでリソースをアクセスできれば、パスワード変更は成功です。

FCC でのユーザによるパスワード変更を有効にする方法（SecureURLs=Yes）

ユーザが必要に応じていつでも自分のパスワードを変更できるように SiteMinder の FCC パスワード サービス機能を設定できます。

注: SiteMinder Web エージェント設定で SecureURLs パラメータの値も yes に設定されている場合のみ、以下のプロセスを実行します。

FCC でのユーザによるパスワード変更を有効にするには、以下の手順に従います。

1. ユーザ ディレクトリにパスワード ポリシーをサポートする属性が含まれていることを確認します。
2. 管理 UI を使用して以下のタスクを実行します。
   a. FCC ベースのパスワード ポリシーを作成し、対象のリソースを保護します。
   b. 権限のあるユーザがパスワードを変更できるようにパスワード ポリシーを設定します。
   c. エージェント設定で ValidTargetDomain パラメータの値を、保護するターゲット リソースのドメインに設定します。
3. 以下が含まれるパスワード変更 URL を FCC 形式で作成します。
   - ログオン サーバの FQDN（例: http:*logonserver.example.com*）
   - FCC ベースのパスワード サービスの URI（例: siteminderagent/forms/smpwservices.fcc?）
   - SiteMinder エージェントの名前（SMAGENTNAME）
   - SiteMinder によって保護されるターゲット リソース（TARGET）。
4. Web サーバで以下のファイルを開きます。

   web_agent_home/samples/forms/smpwservices.fcc

   a. 次の行を検索します。

      ```
      @smpwselfchange=0
      ```

   b. 以下の例のように、この行の最後の値を 0 から 1 に変更します。

      ```
      @smpwselfchange=1
      ```

   c. smpwservices.fcc ファイルを保存して閉じます。
5. 以下の例のように、手順 3 で作成した URL を、保護されていない Web ページにリンクとして埋め込みます。

```
<a
href="http:logonserver.example.com/siteminderagent/forms/smpwservices.fcc?SMA
UTHREASON=
34&SMAGENTNAME=$$smencode(smagentname)$$&TARGET=$$smencode(target)$$">Change
Password</font></a>
```

6. 以下の手順でパスワード変更をテストします。
   a. 手順 3 で作成したパスワード変更リンクがある Web ページを表示します。
   b. パスワード変更リンクをクリックします。

      パスワード変更フォームが表示されます。
   c. パスワード変更フォームに入力し、それをサブミットします。

      パスワードの変更に成功した場合、確認ページが表示され、保護されているターゲット リソースへのリンクが表示されます。
   d. リンクをクリックし、リソースが表示されることを確認します。
   e. ブラウザを閉じて再度開きます。新しいパスワードを使用して、保護されているリソースへのアクセスを試みます。

      新しいパスワードでリソースをアクセスできれば、パスワード変更は成功です。

FCC パスワード サービスを伴う SecureID 認証の設定

認証方式として SecureID を使用し、以下の両方の条件が環境内に存在する場合は、管理 UI を使用して SecureID HTML フォーム テンプレートを変更する必要があります。

- FCC パスワード サービス機能が設定されます。
- Web エージェントの SecureUrls パラメータの値が yes に設定されます。

SecureID はパスワード サービスを使用して実装されます。認証方式のテンプレートを変更する必要があるのはこのためです。

FCC パスワード サービスを使用して SecureID 認証を設定するには、以下の例に示されるように、SecureID テンプレートの[ターゲット]フィールドに smpwservices.fcc ファイルのパスを追加します。

```
/siteminderagent/forms/smpwservices.fcc
```

FCC ベースのパスワード サービス変更フォームのローカライズ

FCC ベースのパスワード サービスのユーザ メッセージをローカライズすることができます。FCC ベースのパスワード サービスを変更するための以下の手順は、どのロケールにも使用できます。

1. 新規ロケール用の新しい FCC フォルダを Web サーバに作成します。または、すでにロケールに適したフォルダがある場合は、その既存のフォルダを使用します。フォルダの命名規約は forms*locale* です。
2. 新しいフォルダに、関連するパスワード サービス ファイルをコピーします。
3. ロケールに合わせてファイルを変更します。たとえば、英語のメッセージをロケールの言語に変更します。ロケール内で使用するファイルに対してこの手順を繰り返します。
4. 管理 UI で、[パスワード ポリシー]の[リダイレクト URL]フィールドの値を変更します。

   たとえば、日本語のユーザに FCC パスワード サービスを使用するには、*web_agent_home*/samples にある以下のファイルをフォルダ formsja にコピーします。

   - smpwservices.fcc（*web_agent_home*>/samples/forms 内）
   - smpwservices.unauth（*web_agent_home*>/samples/forms 内）
   - 新しい properties ファイル smpwservicesja.properties

CGI ベースのパスワード サービス変更フォームのローカライズ

パスワードサービス変更フォームに関連して、Web ブラウザから言語エンコードの読み込みが行われます。パスワードサービスの CGI は、Web ブラウザから ACCEPT_LANGUAGE 変数を読み込み、パスワード変更フォームをどの言語で表示するかを決定します。

言語ごとに、変更フォームは異なるプロパティファイルを使用します。プロパティファイルは、HTML ページ内で表示されるテキストの特定の要素を定義します。

次のプロパティファイルは、Web エージェントと共に自動的にインストールされます。

- 英語: PasswordServicesUS-EN.properties
- フランス語: PasswordServicesFR-FR.properties
- 日本語: PasswordServicesJP-JP.properties

ACCEPT_LANGUAGE 変数が、特定の言語に対応するプロパティファイルが存在しないことを示している場合、英語のプロパティファイルが使用されます。

他の言語のプロパティ ファイルを作成および使用する方法

1. SiteMinder Web エージェントに付属しているプロパティ ファイルのいずれかを、希望の言語に翻訳します。これらのファイルは、*web_agent_home*/pw 内にあります。

   たとえば、PasswordServicesUS-EN.properties ファイルをコピーし、等号 (=) より右にあるすべての文字を翻訳します。

2. パスワード サービスの規約に従ってファイルの名前を変更します。ファイルの命名規約は PasswordServices*language-country*.properties です。

   注: RFC 3066 で、命名規約の一部として使用できる言語タグが定義されています。「IETF」を参照してください。さらに、ブラウザによって返された ACCEPT_LANGUAGE 変数でその値として言語タグのみが使用される場合は、ファイル名を PasswordServices*language*.properties に変更します。

   ブラウザが正しい言語に設定されていることを確認してください。

   1 つのブラウザで複数の言語を設定した場合、パスワード サービスは、そのリストの中にある最初の言語だけを使用します。パスワードサービスの CGI が、その言語に対応するプロパティ ファイルを見つけることができない場合、英語のプロパティ ファイルが使用されます。ブラウザに対して指定された他の言語は無視されます。

パスワード サービス リダイレクトでの完全修飾 URL の使用

以下のパラメータを設定することによって、Web エージェントにパスワード サービス アプリケーションへのリダイレクト用の完全修飾 URL を使用させることができます。

ConstructFullPwsvcUrl

ユーザをパスワード サービス アプリケーションにリダイレクトするための完全修飾ドメイン名を備えた URL を生成するように Web エージェントに指示します。これにより、特定の Web サーバ上のパスワード サービス アプリケーションをホストすることができます。Web エージェントは、以下の例のような URL を生成します。

```
HTTP://my.server.com:80/path/to/passwordservices.cgi
```

完全修飾 URL が使用されない場合、Web エージェントは、パスワード サービス アプリケーションが同じ Web サーバ上でホストされ、リダイレクト用の相対 URL を使用すると想定します。

デフォルト: No

たとえば、特定の Web サーバにすべてのパスワード サービスがある場合は、このパラメータを yes に設定します。このパラメータを no に設定した場合は、パスワード サービスと共に使用されるあらゆるカスタマイズされたコンテンツがすべての Web サーバにコピーされる必要があります。

パスワード サービスのリダイレクト用の完全修飾 URL を使用するには、ConstructFullPwsvcUrl パラメータの値を yes に設定します。

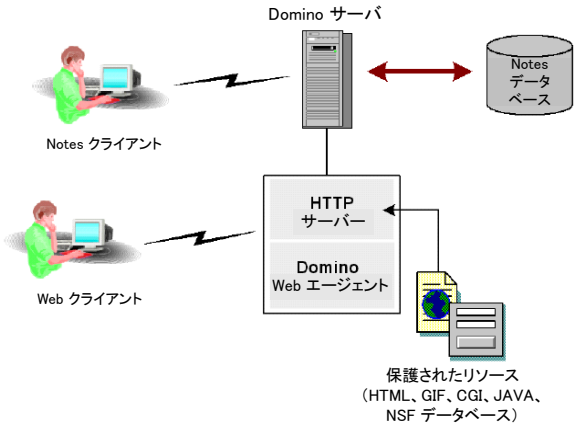
第 22 章: Domino Web エージェント

このセクションには、以下のトピックが含まれています。

Domino エージェントの概要

Domino アプリケーション サーバはメッセージング/Web アプリケーション プラットフォームであり、セキュリティ保護されたアクセスを LotusNotes クライアントに対して提供します。Domino Web エージェントは、HTTP インタフェースである Domino アプリケーション サーバのみを保護し、HTML、JAVA、CGI などの Web リソースへのアクセス制御を行います。Domino Web エージェントは Notes サーバを保護しません。

以下の図に、Domino Web エージェントが Domino サーバとどう統合されるかを示します。



Domino では、データは複数の Notes データベース内に保存されます。データベースに保存されるリソースとしては、ドキュメント、ビュー、フォーム、ナビゲータなど、さまざまな種類のオブジェクトが考えられます。 これらのオブジェクトには、テキスト、ビデオ、グラフィック、オーディオなどのコンテンツを含めることができます。

Notes オブジェクトを開くには URL を使用します。データベース内の Notes オブジェクトを Web 経由で利用できるように、Domino はオブジェクトから Web ページを動的に作成します。データベース ビューの場合、Domino はビュー内の各ドキュメントへの URL リンクも作成します。Notes データベースからページを動的に作成することにより、最新の情報をユーザに提供できます。

Domino URL コマンド

Domino サーバ上のリソースへのインターネット経由でのユーザ アクセスは URL に基づきます。このため、Domino URL の構文を理解することが重要になります。

ただし、Domino は以下のような標準 URL を解釈することができます。

http://www.mycompany.com/index.html

Domino の URL コマンドは、以下の構文にすることもできます。

http://host/database/Domino_object?Action_Argument

host

サーバの DNS エントリまたは IP アドレスを示します。

database

notes ¥data ディレクトリを基準とするパスまたはデータベース レプリカ ID で、データベース ファイル名を指定します。

Domino_object

ビュー、ドキュメント、フォーム、ナビゲータなど、データベース内のオブジェクトを指定します。

Action

Notes オブジェクトに対して実行する操作を示します。たとえば、?OpenDatabase、?OpenView、?OpenDocument、?OpenForm、?ReadForm、?EditDocument などがあります。URL にアクションが指定されていない場合は、デフォルトが使用されます。

デフォルト: ?Open

Argument

オブジェクトが Domino サーバによってどのように提供されるかの詳細な定義を提供します。たとえば、アクションと引数が ?OpenView&Expand=5 である場合、この引数は展開形式で表示する際のビュー内の行数を指定しています。

financials.nsf という名前の Notes データベース内の特定のビューにアクセスする URL の例を次に示します。

http://www.anysite.com/financials.nsf/reports?OpenView

Domino のエイリアス

Notes データベース規約の 1 つにオブジェクトのエイリアスの作成があります。たとえばエイリアスでは、オブジェクト名の代わりに Notes ID またはレプリカ ID を使ってリソースを識別できます。エイリアスを使用すると、開発者のプログラミング作業が簡単になります。というのも、Notes リソースの名前を変更しても、コードを変更する必要がなくなるからです。

次の Domino URL はそれぞれ異なるエイリアスによって識別されていますが、すべて同一のリソースにアクセスします。

- http://www.domino.com/85255e01001356a8852554c20756?OpenView
- http://www.domino.com/85267E00075A80C/people?OpenView
- http://www.domino.com/__852567E00075A80C.nsf/people?OpenView

Domino Web エージェントはデータベースリソースの識別方法に関係なく、Domino 命名規約に従ったすべての ID を、リソースの名前に基づく標準の URL に変換します。これにより、SiteMinder ポリシー ストアへのデータ入力が簡略化されます。

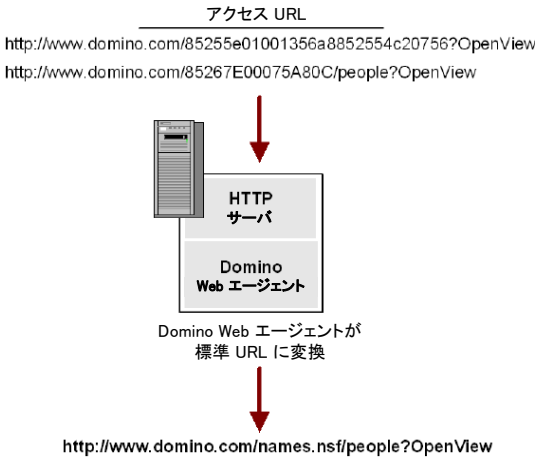
たとえば以下の Domino URL は、names.nsf データベース内の people ビューを指しています。データベースとビューはそれぞれ、レプリカ ID と Notes ID で参照されています。

- http://www.domino.com/85255e01001356a8852554c20756?OpenView
- http://www.domino.com/85267E00075A80C/people?OpenView

Domino Web エージェントはこれらの URL を次の標準 URL に変換します。

- http://www.domino.com/names.nsf/people?OpenView

以下の図に、エイリアスから名前付きオブジェクトへの変換を示します。



Notes ドキュメント名の変換

ビューやフォームと異なり、Notes ドキュメントは名前を持ちません。Notes ドキュメントは、ドキュメント作成時に使用したフォームのリファレンスと共にデータベースに保存されます。ユーザがドキュメントにアクセスしようとしたときに、Domino Web エージェントがそのドキュメントを読み取り可能な名前に変換できなかった場合、エージェントはそのドキュメントを生成したフォームの名前を使って URL を作成します。ただし、この規則はドキュメントにのみ適用されます。元のフォームが存在しない場合、エージェントは埋め込まれたフォームを使用します。どちらも存在しない場合、Domino の識別子 $defaultForm によってドキュメントが保護されます。

たとえば、次のような URL を受信したとします。

http://www.domino.com/names.nsf/8567489d60034we50938450098?OpenDocument

この場合、エージェントは次の URL を使用します。

http://www.domino.com/names.nsf/Person?ReadForm

この例で、Person はドキュメントの名前です。

Domino Web エージェントの設定

Domino Web エージェントは、すべての Web エージェント標準設定を使って次のことを実行できます。

- ポリシー サーバと通信する Web エージェントの設定
- 仮想サーバのエージェント ID の追加と削除
- Web エージェントの設定の変更
- シングル サインオンの設定
- エラー メッセージ ロギングの設定

これらの設定は、ポリシー サーバ上で集中的に、またはエージェント設定ファイル内でローカルに実行することができます。

標準的な機能に加えて、設定可能な Domino 特有のパラメータもあります。

Domino 固有のエージェント機能の設定

Web エージェント標準設定のほかに、Domino Web エージェントの場合に限って設定可能な Domino 固有の設定パラメータが存在します。これらの設定により、Domino が SiteMinder と連携してユーザを認証して許可する方法が決まります。この設定は、ポリシー サーバ上のエージェント設定オブジェクトに一元的に設定するか、または Web サーバ上のエージェント設定ファイルにローカルに設定できます。

注: Domino Web エージェントでは、ユーザ アクティビティの追跡に使用される監査機能はサポートされません。

Domino サーバによるユーザ認証

SiteMinder がすでにユーザの認証および許可を完了している場合でも、Domino サーバはユーザの認証および許可を行う必要があります。SiteMinder は Domino の認証プロセスと連携して Domino サーバにユーザ ID を提供します。この ID は、ユーザとその権限の一覧が保存された Domino ディレクトリ内にも設定されます。Domino サーバはこの ID を使って、ユーザの認証およびデータベースリソースへのアクセス許可を行います。

注: ユーザ名は明確に解決される必要があります。そうしないと、Domino エージェントで認証リクエストが拒否されます。その場合は、使用中のユーザ ディレクトリに調整を加える必要が生じることがあります。

Domino エージェントは、ユーザ ID を次のいずれかとして、Domino サーバに提供します。

- スーパーユーザ
- 実ユーザ
- デフォルト ユーザ

Domino サーバとの通信時に Domino Web エージェントが使用する ID を決定するには、次のパラメータを設定します。

SkipDominoAuth

サーバ認証のために Domino サーバに渡す名前を指定します。

DominoSuperUser

Domino サーバ上のすべてのリソースにアクセスできるユーザを指定します。

DominoDefaultUser

Notes データベースに対するデフォルトのアクセス権を持つユーザを指定します。これは、そのユーザが一般的なアクセス権限を持っていることを意味します。

注: DominoSuperUser と DominoDefaultUser は、ローカルのエージェント設定ファイルで設定することも、エージェント設定オブジェクトで一括設定することもできます。エージェント設定ファイル内では、これらの設定項目の値は暗号化されています。 エージェント設定オブジェクト内では、それらの値を暗号化するか、プレーンテキストのままにするかを選択できます。

詳細情報

SiteMinder によるユーザ認証 (P. 309)
Domino スーパー ユーザとしての認証 (P. 310)
実ユーザまたはデフォルト ユーザとしての認証 (P. 310)

SiteMinder によるユーザ認証

Domino ではなく SiteMinder でユーザの認証を行うには、SkipDominoAuth パラメータを yes に設定します。

SkipDominoAuth に yes が設定され、スーパーユーザーが定義されていると、最初に SiteMinder がユーザを識別して許可します。次に、Domino Web エージェントがそのユーザをスーパーユーザーとして Domino サーバに通知します。そのユーザはスーパーユーザーなので、適切な ACL が割り当てられていることを前提として、Domino サーバ上のすべてのリソースにアクセスできます。

ユーザが Domino ディレクトリに保存されていない場合にも、SkipDominoAuth パラメータに yes を設定する必要があります。これは、許可権限に使用すべき ID が Domino 内に存在しないためです。

SkipDominoAuth を no に設定した場合、Domino は実ユーザ名またはデフォルトユーザ名を使って独自にユーザを認証します。

以下の表は、SkipDominoAuth パラメータの設定がユーザの識別方法にどう影響するかを示しています。

SkipDominoAuth の値	Domino サーバでの識別の種類	Notes
はい	スーパーユーザ	スーパーユーザは Domino ディレクトリに定義されている必要があります。
いいえ	実ユーザ	ユーザは Domino ディレクトリに存在している必要があります。
いいえ	デフォルトユーザ	ユーザは Domino ディレクトリに存在している必要があります。
いいえ	スーパーユーザー	要求されたリソースは自動的に許可されます。つまり、このユーザには認証チャレンジは表示されません。

Domino スーパー ユーザとしての認証

Domino スーパー ユーザは、Domino サーバ上のすべてのリソースにアクセスできるユーザです。Web サイトやポータルの設計で SiteMinder の使用が考慮されている場合は、SiteMinder ポリシーを実装することによって、リソースおよびアプリケーションを保護します。その結果、Domino サーバが独自のセキュリティ機構を使ってユーザのアクセスを制限する必要性はなくなります。この場合、ユーザを Domino の認証目的ではスーパー ユーザとして識別することができます。

ユーザをスーパー ユーザとして識別するには、SkipDominoAuth パラメータを有効化し、DominoSuperUser パラメータに値を指定します。このアクションにより、Domino ではなく SiteMinder がユーザを認証するようになります。指定したユーザは、Domino ディレクトリ内にも存在している必要があります。

実ユーザまたはデフォルト ユーザとしての認証

対象ユーザが Domino ディレクトリに定義されている場合は、Domino はそのユーザ名を使ってユーザ認証を行います。ただし、そのユーザが Domino ディレクトリに存在せず、SiteMinder によって別のユーザ ディレクトリに対して認証済みである場合は、Domino Web エージェントは Domino サーバに対してそのユーザを DominoDefaultUser として識別します。

デフォルト ユーザは Notes データベースに対するデフォルトのアクセス権を持ちます。つまりこのユーザは、ACL で設定されている Domino のディポジッタ、リーダ、作成者レベルのアクセス権などの一般アクセス権限を持つことになります。

Domino エージェントがこの値を使用するには、SkipDominoAuth パラメータに no を設定する必要があります。

SiteMinder で保護する必要のない Notes データベースが存在することがあります。SiteMinder によって保護されないリソースは、デフォルトの Domino ユーザとして認証されません。代わりに、Domino サーバは（匿名アクセスが無効である場合）、ユーザに認証情報を求めるプロンプトを表示します。

Domino デフォルト ユーザおよび Domino スーパー ユーザの変更

DominoDefaultUser および DominoSuperUser の各パラメータを変更するには、以下の作業のいずれかを実行します。

- ローカルで設定する場合は、エージェント設定オブジェクト内でこれらのパラメータを変更します。

  DominoDefaultUser および DominoSuperUser の各設定は、エージェント設定オブジェクト内で変更できます。値を暗号化するか、プレーンテキストのままにするかを選択できます。

  注: 詳細については、ポリシー サーバドキュメントを参照してください。

- encryptkey ツールを使用して、エージェント設定ファイル内のパラメータを変更します。

  エージェント設定ファイル内では、DominoDefaultUser および DominoSuperUser の各値を暗号化する必要があります。したがって、encryptkey ツールを使用して、これらの値を変更する必要があります。

  重要: エージェント設定ファイルの中で、これらの設定項目を直接編集することは避けてください。

Encryptkey の使用による Domino デフォルト ユーザまたは Domino スーパー ユーザの設定

エージェント設定ファイル内で DominoSuperUser または DominoDefaultUser の値を設定または変更するには、以下の手順に従います。

1. 以下のいずれかの操作を行います。
   - UNIX: Domino エージェントの bin ディレクトリに移動します。以下に例を示します。

     /$HOME/ca/SiteMinder/Web Agent/bin
   - Windows: コマンド プロンプト ウィンドウを開き、Domino エージェントの Bin ディレクトリに移動します。以下に例を示します。

     C:¥Program Files¥ca¥SiteMinder Web Agent¥Bin
2. 以下の引数を指定して、encryptkey ツールを実行します。
   - DominoSuperUser の場合:

     encryptkey -path *path_to_Agent_config_file*

     -dominoSuperUser *new_value*
   - DominoDefaultUser の場合:

     encryptkey -path *path_to_Agent_config_file*

     -dominoDefaultUser *new_value*

   以下に例を示します。

   encryptkey -path "c:¥program files¥ca¥SiteMinder Web Agent¥Bin¥Lotus Domino5¥webagent.conf"

   -dominoSuperUser admin

   注: エージェント設定ファイルのパスには、webagent.conf などのファイル名を含める必要があります。また、パス内の任意の値にスペースが含まれている場合、パス全体を引用符で囲む必要があります。

   注: encryptkey ツールは、SiteMinder Web エージェント キットには含まれていませんが、Domino ユーザに役立つツールです。Domino ユーザはこのツールを扱って、ローカル設定用の暗号化された DominoSuperUser 値を生成することができます。このツールのダウンロードについては、サポートにお問い合わせください。

FCC リダイレクト用 URL のマップ

URL 変換を受け入れるには、DominoMapUrlForRedirect パラメータを設定します。このパラメータはデフォルトで有効になっています。

この設定パラメータが YES に設定されている場合、または存在しない場合、Web エージェントは URL を Domino サーバの表現からフォーム認証情報コレクタ（FCC）のリダイレクト用の URL 対応の名前にマップ（正規化）します。

このパラメータが NO に設定されている場合、Web エージェントは URL をマップせず、Domino サーバの表現を使って FCC リダイレクトを実行します。

SiteMinder と Domino 認証の整合

次に、SiteMinder と Domino 認証の整合について説明します。

SiteMinder ヘッダを使用した認証

DominoUseHeaderForLogin および DominoLookUpHeaderForLogin の各パラメータを使用して、Domino ユーザを認証することもできます。

DominoUseHeaderForLogin

SiteMinder のヘッダの値を Domino Web サーバに渡すように Domino Web エージェントに指示します。Domino サーバはそのヘッダのデータを使用して、自らのユーザ ディレクトリの中に存在しているユーザを識別します。

このパラメータは、ヘッダ名と同じ値に設定します。たとえば、DominoUseHeaderForLogin="HTTP_SM_USER" と指定した場合、Web エージェントはユーザのログイン名を Domino サーバに渡します。

DominoLookUpHeaderForLogin

ユーザがリソースへのアクセスを要求したときに、そのユーザが Domino ユーザ ディレクトリ内で一意か、またはあいまいかを、Domino Web サーバに問い合わせることを Domino Web エージェントに指示します。Jones という 1 人のユーザがリソースへのアクセスを試み、ユーザ ディレクトリ内に Jones というユーザが複数存在しているときに、このチェックは役立ちます。このパラメータが no に設定されている場合、Domino Web エージェントは Domino Web サーバに確認しません。

デフォルト: yes

Domino セッション認証の無効化

SiteMinder には、認証機能と許可機能が用意されているので、Domino のセッション認証機能は必要ありません。Web エージェントがインストールされている場合は、このセッション機能を無効にする必要があります。

場合によって、Domino のセッション認証を有効にしておくと、ユーザ セッションが不正な動作を引き起こすことがあります。この動作の変化は、SiteMinder 対応サイトのセキュリティには影響しません。これは、SiteMinder と Domino のセッション管理ルールの AND 演算を反映しています。

Domino Web エージェントによる FCC リダイレクト用 URL のマップ

フォーム認証方式を使用して Domino ビュー（.nsf）のリソースを保護している場合は、フォーム認証情報コレクタにリダイレクトする前に、URL をマップする必要があります。

リダイレクトの前に URL をマップするには、以下の手順に従います。

1. DominoNormalizeUrls パラメータに yes を設定します。
2. DominoMapUrlForRedirect パラメータに yes を設定します。

   Domino の URL は、フォーム認証情報コレクタにリダイレクトされる前にマップされます。

URL 正規化の無効化

URL 正規化の目的は、URL を Domino の表現から一般的なブラウザで使用される URL 形式に変更することです。Domino Web エージェントは Domino Web サーバの API を利用して Domino の URL を正規化します。

正規化プロセスの際、Domino サーバの API は、正規化された URL に復帰文字（16 進数の 0x0D）または改行文字（16 進数の 0x0A）あるいはその両方を追加して URL を定期的に返します。これらの文字の追加は、特定の Notes データベース（.nsf）ファイルおよびこれらのファイル内のアクセス パターンに関連していると考えられます。

以下の例に、復帰文字が追加された正規化後の URL を示します。

- URL：
  http://server.ca.com:80/agentrunner.nsf/be68f4545348400461332?OpenView
- URL のマップ先：
  http://server.ca.com:80/agentrunner.nsf/AgentContext?OpenView
- URL の正規化：
  http://xxxxx.ca.com:port/agentrunner.nsf/0x0d/AgentContext?OpenView

必要に応じて、以下のパラメータを使用して、Domino リソース ID を含む URL が正規化されないようにすることができます。

DominoNormalizeUrls

SiteMinder Web エージェントが、フォーム認証情報コレクタにリダイレクトする前に、Domino の URL を URL フレンドリ名に変換するかどうかを指定します。

Domino の URL を変換するためには、MapUrlsForRedirect パラメータも yes に設定する必要があります。

DominoNormalizeUrls パラメータが no の場合は、MapUrlsForRedirect パラメータが yes に設定されていても、URL は正規化されません。

重要： DominoNormalizeUrls パラメータを no に設定した場合、Notes データベース内の個々のドキュメントを保護することはできません。Domino Web サーバのデータベース全体またはサブディレクトリのみを保護することができます。

デフォルト： yes

正規化をオフにして URL が変更されないようにするには、DominoNormalizeUrls パラメータを no に設定します。

Lotus Notes ドキュメントへのアクセスの制御

Web エージェントでは、Domino 上の Lotus Notes ドキュメントの保護を詳細に制御することができます。この保護は、以下のパラメータで制御します。

DominoLegacyDocumentSupport

Web エージェントが Domino 環境内の保護されている Lotus Notes ドキュメントに対するユーザ要求を処理する方法を指定します。このパラメータを yes に設定すると、要求されたドキュメントに対してのみ、ユーザに ReadForm 許可が与えられます。

デフォルト: No

Notes ドキュメントにアクセスしているときにユーザが要求したアクションを処理するように Web エージェントを設定するには、DominoLegacyDocumentSupport パラメータを使用します。この方法で、Domino の保護をより詳細に制御できます。

Notes ドキュメントには、個別の名前がありません。それらのドキュメントは、作成の際に使用されたフォームへの参照と共に、データベース内に保存されています。ユーザが何らかの Notes ドキュメントをリクエストした場合、Domino Web エージェントはそのリクエストを URL へ変換することにより、該当するフォームを見つけます。この URL の中に、Domino に対する元のアクションが含まれています。フォームが見つからない場合は、何も使用されません。

以下に例を示します。

"http://server.domain.com/db.nsf?OpenDocument"

?OpenDocument や ?EditDocument など、ユーザがこの URL で指定されているドキュメントに対して要求した Domino アクションを Web エージェントが実行するようにするには、DominoLegacyDocumentSupport パラメータを no に設定します。

たとえば、次のような URL へのリクエストが発生したとします。

http://www.dominoserver.com/names.nsf/9348730948938987785784395880920398579834 9?EditDocument

Domino エージェントは、上記の URL を、次のように変換します。

http://www.dominoserver.com/names.nsf/Person?EditDocument

Person は、どのドキュメントを作成する際に使用されたフォームの名前であり、元の URL の中では、NotesID によって指定されています。

Notes ドキュメントにアクセスする際に、4.6 より前の動作を実行するよう Domino Web エージェントに指示するには、このパラメータを yes に設定します。これは、?ReadForm アクションのみを許可することを意味します。レガシードキュメントサポートが有効になっている場合、Domino エージェントは上記の例の URL を、次のように変換します。

http://www.dominoserver.com/names.nsf/Person?ReadForm

認証を目的とした Domino エージェントによる認証情報の収集

認証情報コレクタは、Web エージェント内にあるアプリケーションの 1 つで、フォーム、SSL、Windows の各認証方式、および複数の cookie ドメインへのシングル サインオンに関して、ユーザ認証情報を収集します。認証情報コレクタが収集する認証情報は、保護されたリソースからなる特定のグループに関して設定された認証方式のタイプを基礎としています。

Domino Web エージェントを認証情報コレクタとして機能させるには、エージェント設定ファイルの中でファイル拡張子として表現されているさまざまな MIME タイプを設定する必要があります。

認証情報コレクタは、一般的に自動認証されます。つまり、1 つのファイル拡張子をこれらのパラメータに追加した時点で、その拡張子はデフォルトで IgnoreExt パラメータの中に含まれます。Domino サーバは、これらの拡張子がついたファイルを含む URL を正しく処理することができません。そのため、Domino エージェントはそのようなファイルを無視する必要があります。

注: 詳細については、ポリシー サーバドキュメントを参照してください。

omino に関するユーザ ディレクトリの指定

Domino ディレクトリはすべての Domino サーバと統合化されます。Domino サーバの LDAP サービスを有効化すると、ポリシー サーバによるユーザ認証/許可時に Domino ディレクトリを使用できます。Domino の LDAP サービスを有効化した場合、認証用に別のユーザ ディレクトリを設定する必要はありません。

LDAP サービスを有効化する方法については、Domino サーバのマニュアルを参照してください。

詳細情報:

CAへの連絡先 (P. iii)

Domino のポリシーの設定

Domino サーバでは、同一の Notes オブジェクトをさまざまな方法を使って表現できます。オブジェクトの識別には、名前、レプリカ ID、ユニバーサル ID、およびエイリアスが使用できます。

Domino Web エージェントは、Domino サーバとの通信を効率的に行うために、Notes リソースへのアクセス リクエストを処理する際にオブジェクト名のみを使用します。これにより、SiteMinder ポリシー ストアはエントリを認識できるようになります。

任意のリソースへのアクセス方法を URL で表現すると、次のようになります。

```
http://host/database.nsf/resource_name?Open
```

Domino サーバ リソースのルールの作成

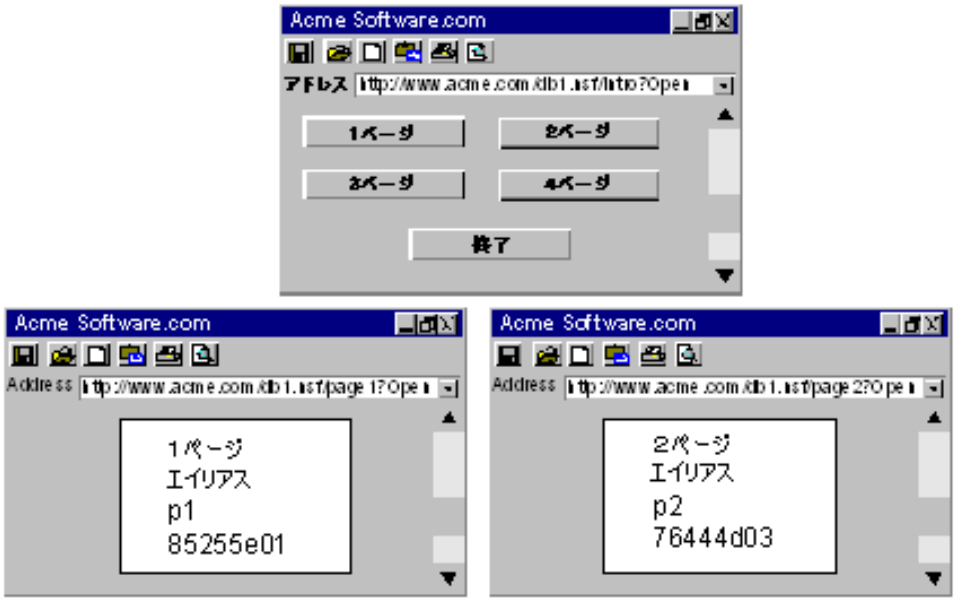
ルールを作成するには、Notes データベースリソースのアクションについて考慮する必要があります。アクションが指定されていないリソースのデフォルトのアクションは ?Open になります。SiteMinder ポリシーに含まれるルールは、デフォルトのアクション ?Open、および ?OpenDatabase、?OpenView、?OpenDocument、?OpenFrameset など、?Open の同等のアクションを考慮する必要があります。

Domino Web エージェントを使用すると、ポリシー管理者は、同一リソースを参照する多くのエイリアスに対して 1 つのルールを作成することができます。ルールを 1 つしか作成する必要がないのは、Domino エージェントは複数の Domino リソース表現を 1 つの URL に変換するためです。SiteMinder ポリシーのルールを作成する際は、Domino エージェントのこの機能を検討することが重要です。

レルムとルールの作成には、管理 UI を使用します。

注: 詳細については、ポリシー サーバドキュメントを参照してください。

以下の図の URL は、db1.nsf という Notes データベースが存在する Acme の Domino サーバへのリンクです。このデータベースには、2 つのファイル (ページ 1 とページ 2) が含まれています。



例 1: 1 つのドキュメントとそのすべてのエイリアスの保護

ページ 1 およびそのすべてのエイリアスへのアクセスに対して、レルム db1.nsf にルールを 1 つのみ作成します。Domino エージェントは、異なるすべての命名規則を解釈して 1 つの標準 URL 形式に変換することができます。

レルムおよびルールについて、次の手順に従います。

- レルムの作成時に、ページ 1 が含まれているデータベースに以下のようにリソース フィルタを指定します。たとえば、データベース内のすべてのファイルを保護するには、以下のように設定します。

  Resource filter: /db1.nsf/

  ページ 1 に加えて、そのすべてのエイリアスを保護するには、以下のように設定します。

  Resource filter: /db1.nsf/page1

- ページ 1 の任意のアクションを保護するルールを作成するには、アスタリスク(*)を[ルールのプロパティ]ダイアログ ボックスの[リソース]フィールドに入力します。以下に例を示します。

  Resource: *

  このワイルド カード * は、ポリシーに結び付けられているユーザが、ページ 1 に対して ?Open、?EditDocument など任意のアクションを実行できることを表します。

例 2: 同一データベース内の異なるドキュメントの保護

ページ 1 のほかに db1.nsf データベース内の ページ 2 を保護するには、以下のような 2 番目のルールを作成する必要があります。

Resource Filter: /db1.nsf/page2

Resource: *

例 3: 同一リソースに対する異なるアクションの保護

あるリソースのアクションを別々に保護するには、たとえば、特定のユーザだけにアクション ?EditDocument を実行させ、その他のユーザにアクション ?ReadForm を実行させる場合は、各リソースのアクションごとに異なるルールを次のように定義する必要があります。

- ルール 1

  Resource Filter: /db1.nsf/page1

  Resource: ?OpenView

- ルール 2

  Resource Filter: /db1.nsf/page1

  Resource: ?EditDocument

また、次のように 1 つのルールを使用することもできます。

Resource Filter: /db1.nsf/page

Resource: ?Open*

注: [リソース]フィールドでは、?Open の前にスラッシュ(/)を付けません。

このリソースのエイリアスが存在する場合でも、このルール 1 つでオリジナルのページとそのすべてのエイリアスが保護されます。

アクションごとにルールを作成する代わりに、次のように 1 つのルールを指定して、すべてのアクションをカバーするワイルドカードを使用することもできます。

Resource filter: /db1.nsf/page

Resource: ?Open*

このルールを使用すると、次のようなリソースを保護することになります。

http://www.acme.com/db1.nsf/page*?Open*

注: ルールをリテラルにするには、正規表現を記述してください。

Domino エージェントの完全ログオフ サポートの設定

完全ログオフでは、以下のパラメータで作成するカスタム ログオフ ページが使用されます。

LogOffUri

完全なログ オフを有効にし、正常にログ オフした後にユーザに表示される Web サーバ上のカスタム Web ページの場所を指定します。ブラウザ キャッシュ内に格納できないようにこのページを設定する必要があります。設定しなかった場合、ブラウザは、ユーザをログ オフせずに、そのキャッシュからログオフ ページを表示する場合があります。これは、不正なユーザにセッションの支配権を握る機会を与える場合があります。

注: CookiePath パラメータが設定されているときは、LogOffUri パラメータの値が同じ cookie パスを指している必要があります。たとえば、CookiePath パラメータの値が example.com に設定されている場合、LogOffUri は example.com/logoff.html を指している必要があります。

デフォルト: デフォルトなし

制限: 完全修飾 URL は使用しないでください。相対 URI を使用する必要があります。

例: /Web pages/logoff.html

完全ログオフの設定方法

1. ユーザのログオフ用のカスタム HTTP アプリケーションを作成します。たとえば、ユーザを指定した URL にリダイレクトするための終了ボタンまたはサインオフ ボタンを追加します。
2. HTML ログオフ ページが、ブラウザのキャッシュからではなく、確実に Web サーバからロードされるようにするには、ログオフ ページをブラウザ内にキャッシュできないように設定します。たとえば、HTML ページの場合は、ページに次のようなメタタグを追加します。

```
<META HTTP-EQUIV="Pragma" CONTENT="no-cache">
<META HTTP-EQUIV="Expires" CONTENT="-1">
```

重要: メタ タグは一部の Web ブラウザで機能しない場合があります。その場合、Cache-Control HTTP ヘッダを使用します。

3. 以下の手順で LogOffURI パラメータを設定します。
   a. 必要に応じて、ポンド記号(#)を削除します。

b. ユーザをログオフするカスタム HTTP ファイルの URI を入力します。URL の絶対パスは入力しないでください。

完全ログオフが設定されます。

Domino サーバに関するポリシーを作成する場合の考慮事項

Domino サーバ用の SiteMinder ポリシーを作成する場合は、以下の点を考慮してください。

- ユーザは、親ドキュメントを持つフォームを開いてそのフォームのデフォルト値を参照することができます。親ドキュメントとは、そのドキュメントの作成時に使用した元のフォームのことです。アクセスが禁止されているフォームのデフォルト値をユーザが参照するのを防ぐには、SkipDominoAuth パラメータに no を設定します。
- データベースを同一マシン上で複製した場合、各データベースを別々に保護するには、対応するルールも複製する必要があります。
- Domino エージェントが Notes ドキュメントのエイリアスをフォームに関連付けることができない場合、ドキュメントごとに異なるルールを指定してドキュメントを保護する必要があります。
- 特定の種類のデータベースドキュメントの場合、Domino サーバは URL コマンド内で、$DefaultView、$DefaultForm、$DefaultNav、$SearchForm などの特殊な識別子を使用します。Domino Web エージェントはこれらの識別子を標準の URL に変換してからドキュメントにアクセスします。$defaultNav の場合、Domino エージェントは ?OpenDatabase というアクションを実行します。これらの種類の識別子用に、追加のルールを作成する必要はありません。
- Notes データベース内のリソースはリソースのエイリアスによって保護されます。エイリアスが存在しない場合は、リソース名またはコメントによって保護されます。
- Notes では、種類の異なる複数のオブジェクトが、同一の名前またはエイリアスを持つことができます。?Open* などのように、?Open アクションでワイルドカードを使用するルールを作成した場合、このルールによって、同一のエイリアスまたは名前を共有するさまざまな種類のリソースがすべて保護されることに注意してください。
- ドキュメントは、そのドキュメントの作成元フォームによって保護されます。そのフォームで使用されるアクションは ?ReadForm です。
- Domino エージェントは拡張子 .nsf のファイルを無視しません。この拡張子を IgnoreExt パラメータに追加しないでください。

Domino Web エージェントと WebSphere Application Server の連動

Domino Web サーバは、リクエストが WebSphere サーバに転送される前に、それらのリクエストをインターセプトするフィルタ プラグインを提供することにより、WebSphere Application Server のフロント エンドとして動作します。

第 23 章: 高度な認証方式の設定

このセクションには、以下のトピックが含まれています。

Passport 認証による IIS 6.0 の Web サーバ リソースの保護

IIS 6.0 の Web サーバ リソースを Passport 認証で保護するためには、Passport 認証のための SiteMinder の設定手順に従うほかに、IIS 6.0 の Web サーバ自体で Passport アプリケーションを有効にする必要があります。

Passport 認証を有効にする方法

1. IIS マネージャを開きます。
2. 適切な Web サイトで、Passport アプリケーションを有効にする必要があるディレクトリまたはファイルを右クリックし、[プロパティ]を選択します。

   [プロパティ]ダイアログ ボックスが開きます。
3. 以下のタスクのいずれかを実行します。
   - 手順 2 でディレクトリを右クリックした場合は、[ディレクトリ セキュリティ]タブをクリックします。
   - 手順 2 でファイルを右クリックした場合は、[ファイル セキュリティ]タブをクリックします。

   該当するタブが開きます。
4. [編集]をクリックします。

   [認証方法]ダイアログ ボックスが表示されます。
5. [.NET パスポート認証]チェック ボックスをオンにします。
6. [OK]をクリックします。

   [認証方法]ダイアログが閉じます。
7. [OK]をクリックします。

   ダイアログ ボックスが閉じ、選択したアイテムで Passport 認証が有効になります。

Stronghold からの証明書の削除（Apache エージェントのみ）

Stronghold Web サーバはクライアント証明書をローカルの一時ファイル内に書き込みます。Web エージェントはこのファイルを使って証明書に基づく認証を行います。Stronghold サーバはこのファイルを使って、クライアント証明書の情報を認証時に利用できるようにします。ユーザが Web サイトにアクセスするたびにこれらの証明書ファイルは大きくなって、サーバのディスク領域を消費します。Web エージェントが証明書ファイルを使用し終わったら削除するように、エージェントを設定することができます。

証明書ファイルを削除するには、DeleteCerts パラメータを yes に設定します。

レガシー URL エンコードの受け入れ

CA によって使用されるレガシー URL エンコーディングでは、ドル記号（$）文字を使用します。ドル記号が問題を引き起こす場合、以下のパラメータを使用して、Web エージェントにドル記号の代わりにハイフン（-）文字を使用させることができます。

LegacyEncoding

Web エージェントで、レガシー URL 内のすべてのドル記号（$）文字を強制的にハイフン（-）に置換します。これにより、MSR、パスワード サービス、および DMS に対する下位互換性も保証されます。このパラメータを no に設定すると、Web エージェントは文字列の SM を -SM- に変換します。このパラメータを yes に設定すると、Web エージェントはドル記号（$）文字を変換しません。

デフォルト:（フレームワーク エージェント）No

デフォルト:（従来のエージェント）Yes

ドル記号の代わりにハイフンを使用してレガシー URL をエンコードするには、LegacyEncoding パラメータの値を no に設定します。

第 24 章: シングル サインオンのセキュリティ ゾーン

このセクションには、以下のトピックが含まれています。

セキュリティ ゾーンの定義

以下の用語は、シングル サインオン(SSO)セキュリティ ゾーンに関するものです。

CAC(Centralized Agent Configuration、中央エージェント設定)

Web エージェントが、ポリシー ストアに定義された Web エージェント設定オブジェクトから自身の設定プロパティを取得するためのメカニズムを示します。

cookie プロバイダ

複数のドメインにまたがって Web エージェントにシングル サインオンを実装するためのメカニズムを示します。いずれかのドメインがマスタドメインとして指定されます。その他のドメインの Web エージェントは、マスタドメイン内の Web エージェントにリダイレクトされ、そのドメインの cookie が提供されます。

SSO(Single Sign-On、シングル サインオン)

一度認証されたユーザが、認証情報を再要求されないようにするためのメカニズムを示します。

SSO ゾーン

任意の識別子(ゾーン名)によって定義される SSO のサブセットを示します。単一の cookie ドメイン内でアプリケーション SSO をセグメント化するために使用されます。同一 SSO ゾーン内のすべてのアプリケーションは、相互間で SSO を許可します。SSO ゾーン間でのやり取りは、ゾーン信頼関係の定義に従って許可するかどうかが決まります。

トラステッド SSO ゾーン

SSO に関してローカル ゾーンから信頼されている外部ゾーンを示します。

セキュリティ ゾーンの概要

ユーザは、同じ cookie ドメイン内に（単一のゾーンを表す）、または複数の cookie ドメインにまたがって（別々のゾーンを表す）、シングル サインオン セキュリティ ゾーンを定義できます。その結果、ユーザには同じゾーンの中ではシングル サインオンが適用されますが、別のゾーンに入るときは、ゾーン間に定義された信頼関係に応じて認証情報を再要求されることもあります。トラステッド関係に含まれているゾーンでは、グループ内のいずれかのゾーンで有効なセッションを持つユーザには認証情報が再要求されません。

シングル サインオン セキュリティ ゾーンは、SiteMinder Web エージェントによって完全に実装されます。各ゾーンは、別々の Web エージェント インスタンス上に存在する必要があります。同じエージェント インスタンス上に複数のゾーンを作成することはできません。

セキュリティ ゾーンは、Web エージェントによって生成された cookie によって識別されます。デフォルトで、Web エージェントは、SMSESSION という名前のセッション cookie と SMIDENTITY という名前の ID cookie を生成します。セキュリティ ゾーンの設定時に、Web エージェントは固有の名前を持つセッション cookie と ID cookie を生成して、ゾーンのアフィリエイト関係を cookie 名に反映させることができます。

セキュリティゾーンの利点

SSO セキュリティゾーン機能は、SiteMinder 管理者が同じ cookie ドメイン内にあるシングル サインオン環境をセグメント化する必要がある場合に使用することを目的としています。たとえば、CA.COM というドメインがあるとします。標準の SiteMinder SSO 機能では、CA.COM 内の SiteMinder 保護下にあるすべてのアプリケーションは、SMSESSION cookie を使用してシングル サインオンを管理します。以下のような、SSO セキュリティゾーンが存在しないシナリオを考えてみます。

1. ユーザがアプリケーション（APP1）にアクセスします。ユーザは SiteMinder から認証情報を要求されます。SiteMinder にログインすると、SMSESSION cookie が作成されます。
2. ユーザは 2 つ目のアプリケーション（APP2）にアクセスし、SiteMinder から認証情報を再要求されます（ユーザは APP1 のユーザ認証情報を使用して APP2 にアクセスすることができないため、ルールに従って、SSO は適用されません）。ユーザがログインすると、新しい SMSESSION cookie が作成され、前の cookie は APP2 への新たなログイン セッションによって上書きされます。
3. ここでユーザが APP1 に戻ると、さらにもう一度認証情報を要求されます。これは、ユーザは元の APP1 セッションを失っており、かつ、APP1 は APP2 セッションを受け入れないためです。したがって、APP1 と APP2 の間で SSO は適用されず、非常に面倒な状況になります。

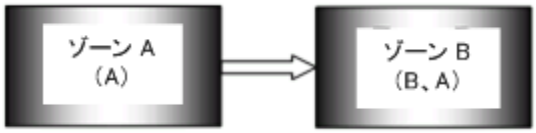
SSO セキュリティゾーンを使用すると、APP1 をゾーン Z1 に、APP2 をゾーン Z2 に置くことができます。ここで、APP1 にログインすると Z1SESSION cookie が作成され、APP2 にアクセスすると Z2SESSION cookie が得られます。名前が異なるので、cookie は互いを上書きすることがなくなります。したがって、上の例のようにユーザはアプリケーション間を移動するたびにログインする必要がなくなり、アプリケーションごとに 1 回のログインで済むようになります。

SSO セキュリティゾーン機能が提供されるまでは、アプリケーションについて SSO を同様にグループ化する唯一の方法は、異なるネットワークドメインの作成を経て、異なる cookie ドメイン（CA1.COM、CA2.COM、その他）を作成し、各種のマルチ cookie ドメイン設定を cookie プロバイダと併用することでした。これは、多くの企業にとって望ましくない方法です。複数のネットワーク ドメインを使用すると、相応の IT 保守およびサポートが必要になるからです。

セキュリティ ゾーンの基本ユースケース

シングル サインオンは、制御された基準に応じて、設定可能な信頼関係を持ついくつかのセキュリティ ゾーンに分けることができます。たとえば、以下のゾーン A とゾーン B について考えてみます。

- ゾーン A の持つトラステッド ゾーンは、ゾーン A 自身のみです。
- ゾーン B は、ゾーン B 自身とゾーン A の 2 つのトラステッド ゾーンを持ちます。



上の図の信頼関係は、矢印で示されています。つまり、ゾーン A で確立されるユーザ セッションは、ゾーン B のシングル サインオンに使用できます。

この例では、ゾーン A は管理者専用ゾーンであるのに対して、ゾーン B は共通のアクセス ゾーンである可能性があります。ゾーン A で認証された管理者は、再認証を要求されることなくゾーン B へのアクセス権を取得できます。ただし、ゾーン B で認証されたユーザは、ゾーン A にアクセスしようとすると、再認証を要求されます。

別のゾーン内のユーザ セッションは、相互に独立しています。ユーザがゾーン B で最初に認証された場合は、ゾーン B で再度認証されます。2 つの異なるセッションが作成されます。実際は、ユーザは両方のセッションで異なる ID を持ちます。ユーザがゾーン A に戻ると、このゾーンで確立されたセッションが使用されます。

ユーザが、まだセッションを確立していないゾーンでシングル サインオンを使用して検証されると、どうなるかについて考えてみます。ユーザがゾーン A で認証されてから、初めてゾーン B にアクセスすると、ゾーン A のセッション情報に基づいてユーザ セッションがゾーン B に作成され、場合によってはポリシー サーバによって更新されます。ゾーン A のユーザ セッションは、ユーザがゾーン A に戻るまで更新されないことに注意してください。

セキュリティ ゾーン間のユーザ セッション

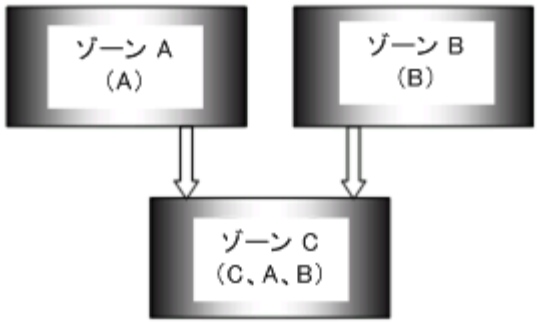
シングル サインオン セキュリティ ゾーンは、すべて同一ドメインに属する必要はありません。 実際は、ゾーンは複数のドメインにまたがることがあります。 ただし、Web エージェントが検索できるのは、ローカル cookie ドメイン内のトラステッド ゾーンの cookie のみです。適切な cookie が見つからない場合は、Web エージェントは自身のゾーンのみで cookie プロバイダへのリダイレクトを続行します。

トラステッド ゾーンの順序

シングル サインオン セキュリティ ゾーンは、以下の 1 組のパラメータによって定義されます。

- セキュリティ ゾーン名
- トラステッド ゾーンの番号付きリスト

トラステッド ゾーンがリストされている順序は重要です。以下の例について考えてみます。



この図では、ゾーン C はゾーン A とゾーン B の両方を信頼しています。ゾーン A とゾーン B はどちらも他のゾーンを信頼していませんが、すべてのゾーンが自身を信頼しています。

ユーザがゾーン C で要求を行うと、Web エージェントは、ゾーンがリストされている順に、トラステッド ゾーンでセッションまたは ID cookie を検索します。この例では、ゾーン C には、C、A、および B が含まれたトラステッド ゾーン リストがあります。

発生する可能性があるイベントの順序を以下に示します。

1. Web エージェントは、ユーザがゾーン C でセッションを確立しているかどうかを最初に確認します。
2. セッションが見つからない場合は、Web エージェントは、ユーザがゾーン A でセッションを確立しているかどうかを確認します。
3. セッションが見つからない場合は、Web エージェントは、ユーザがゾーン B でセッションを確立しているかどうかを確認します。
4. 見つかった各 cookie からのセッションの指定は、正常にログインできるまで認証要求を処理するために使用されます。
5. 認証が正常に行われると、Web エージェントは許可プロセスに進みます。
6. cookie が見つからないか、cookie が認証を渡していない場合は、エージェントは通常どおりユーザに認証情報を要求します。

ゾーンにアクセスする順序によって、ユーザの操作が異なる場合があることに注意してください。この例では、ユーザがゾーン B に最初にアクセスして、次にゾーン C にアクセスする場合は、ゾーン B でのユーザの ID がゾーン C でも使用されます。ユーザがゾーン A に最初にアクセスして、次にゾーン B とゾーン C にアクセスする場合は、ユーザはゾーン C に移動する前にゾーン B で再認証を要求されるにもかかわらず、ゾーン A でのユーザの ID が使用されます。

これは、異なる最大セッション タイムアウトとアイドル セッション タイムアウトが指定されているセッションの有効期限が切れ始めたときにも当てはまります。現在の例では、ゾーン A とゾーン C で有効な cookie を持つユーザは、最初にゾーン C の cookie へのアクセス権を取得します。ゾーン C の cookie の有効期限が切れると、ゾーン A の cookie が使用されます（有効期限が切れていない場合）。そのため、ユーザの ID は、認証情報の要求が行われることなく、ゾーン C の ID からゾーン A の ID に変更されることがあります。

複数の Web エージェントが異なるトラステッド ゾーン リストを持つ場合でも、共通のトラステッド ゾーン名を使用します。この場合は、エージェントは相互を暗黙的に信頼しますが、同一の外部ゾーンは信頼しません。この機能を使用すると、シングル サインオンでアプリケーションをセグメント化できます。Web エージェントは、シングル サインオン ゾーン名のみをサポートします。このエージェントによって生成されるすべてのセッション、ID、および状態 cookie が、同一のシングル サインオン ゾーン名を使用します。そのため、2 つのアプリケーションが同一のシングル サインオン信頼要件を共有していない場合、それらのアプリケーションは、それぞれ独自の Web エージェントとトラステッド ゾーン リストを持つ別々の Web サーバ上でホストする必要があります。

注: 外部ゾーンとは、特定の Web エージェントによってサポートされるゾーン以外のゾーンです。たとえば、エージェントが SSOZoneName="Z1" として設定されている場合は、その他すべてのゾーンはこのエージェントにとって外部ゾーンになります。これには、デフォルト ゾーンである「SM」も含まれます。

デフォルトのシングル サインオン ゾーンおよびトラステッド ゾーン リスト

セキュリティ ゾーン名を指定しない Web エージェント（SiteMinder 6.x QMR 5 より前のすべての Web エージェントなど）は、デフォルト ゾーンに属していると見なされます。後方互換性を確保するために、デフォルト ゾーンは、ゾーン名 SM が付いていると暗黙的に想定されます。これによって、SiteMinder r12.0 SP3 Web エージェントは、設定変更せずにデフォルトで SMSESSION および SMIDENTITY をサポートすることができます。

トラステッド ゾーン リストを指定しない Web エージェントは、自身のシングル サインオン ゾーン（指定されたゾーン名、またはゾーン名が指定されていない場合はデフォルト ゾーン）のみを信頼します。

Web エージェントは、デフォルト ゾーンに加えて他のゾーンを信頼するように設定できます。また、指定されたゾーン名を使用できますが、他のトラステッド ゾーンはリストできません。追加のトラステッド ゾーンが指定されているかどうかに関係なく、エージェントは常に自身のゾーンを最初に信頼します。デフォルト以外のゾーンを使用する Web エージェントがデフォルト ゾーンも信頼するようにするには、トラステッド ゾーン リストに「SM」を追加する必要があります。

複数のユーザ セッションによる要求処理

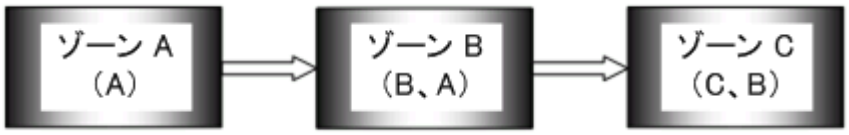
Web エージェントは、トラステッド ゾーンの順序でユーザ セッションを検索します。有効なユーザ セッションが見つかった場合は、Web エージェントはセッション情報を使用して、ユーザ要求を処理します。セッションが見つからない場合は、Web エージェントは、信頼の順序で次に有効なユーザ セッションにフェールオーバします。

検査対象の別のセッションがある場合は、失敗した検証からのレスポンスは無視されます。別のものがない場合は、通常どおり処理されます。これは、Web エージェントが処理対象のトラステッド セッションを 3 つ見つけた場合に、最初の 2 つの検証に失敗すると、3 番目（最後）のセッションの検証からのレスポンスのみが処理されることを意味します。

検証が正常に行われた場合は、Web エージェントはセッションの処理を停止して、正常に行われた検証からのレスポンスの処理を即時に開始します。エージェントが、検証対象のセッションを 3 つ持っている場合に、最初のセッションが正常に検証されると、残りの 2 つは無視され、エージェントは正常に行われた最初の検証に関するレスポンスの処理に移行します。

ゾーン間の推移的な関係

シングル サインオン ゾーン間の信頼関係は、完全に推移的ではありません。ゾーン A がゾーン B によって信頼され、ゾーン B がゾーン C によって信頼される場合は、以下の図に示すように、ゾーン A は必ずしもゾーン C によって信頼されるとは限りません。



シングル サインオン ゾーンの影響を受けるその他の Cookie

SiteMinder は、状態 cookie を使用して、認証と許可に関係するさまざまなイベントを管理します。これらの cookie はすべて、デフォルトで先頭にシングル サインオン セキュリティ ゾーンのプレフィックス SM が付いています。新しいシングル サインオン ゾーン名を指定すると、これらの cookie にも、指定されたデフォルト以外のゾーン名を反映するように名前が付けられます。新しいシングル サインオン ゾーンを定義することで影響を受ける cookie のリストを以下に示します。

- SMCHALLENGE
- SMDATA
- SMIDENTITY
- SMONDENIEDREDIR
- SMSESSION
- SMTRYNO

たとえば、ゾーン名 Z1 を指定すると、Web エージェントは、基本認証を行うために Z1CHALLENGE=YES cookie の作成を開始します。これによって、管理者は、エージェントが相互に干渉しない単一の cookie ドメイン（たとえば、ca.com）で SiteMinder アプリケーションのシングル サインオンの島を作成できるようになります。このシングル サインオンのトラステッド ゾーンのリストでは、予測可能な方法で、これらの孤立したシングル サインオン ゾーン間でシングル サインオンを行うことができます。

シングル サインオン ゾーンと許可

シングル サインオン ゾーンを使用すると、変更を行わずに認証を正常に行った後で、通常は次に許可が行われます。検証プロセスで有効なセッションが識別されると、残りの要求処理ではこのセッションが使用され、要求で識別されたその他のセッションはすべて無視されます。許可が失敗すると、正常に許可される可能性があるセッションがほかにあるかどうかに関係なく、ユーザは再認証を要求されます。

検証に合格した最初のトラステッド セッションは、許可に渡されるセッションです。このセッションの許可が失敗すると、ユーザは認証情報を要求されます。

セキュリティ ゾーンの設定

2 つのシングル サインオン パラメータが、ポリシー ストア内の Web エージェント設定オブジェクトに追加されました。これらの設定は、ローカル設定ファイルでも使用される可能性があり、インストール時に作成されるサンプルのローカル設定ファイルに追加されます。

注: 同一のエージェント設定オブジェクトを使用して設定される Web エージェントはすべて、同一のシングル サインオン ゾーンに属します。

SSOZoneName

Web エージェントがサポートするシングル サインオン セキュリティー ゾーンの(大文字と小文字を区別する)名前を指定します。このパラメータの値は Web エージェントが作成する cookie の名前に先頭に付けられます。これは、それぞれの cookie ドメインと cookie を関連付けるのに役立ちます。このパラメータが空でない場合、SiteMinder は以下の規則を使用して、cookie を生成します。

ZonenameCookiename

デフォルト: 空(ゾーン名として SM を使用します。これは cookie に以下のデフォルトの名前を与えます)

- SMSESSION
- SMIDENTITY
- SMDATA
- SMTRYNO
- SMCHALLENGE
- SMONDENIEDREDIR

制限: 単一値

例: Z1 に値を設定すると以下の cookie が作成されます。

- Z1SESSION
- Z1IDENTITY
- Z1DATA
- Z1TRYNO
- Z1CHALLENGE
- Z1ONDENIEDREDIR

SSOTrustedZone

シングル サインオン セキュリティ ゾーンの信頼の信頼された SSOZoneNames の順序づけられた（大文字と小文字を区別する）リストを定義します。必要に応じて、SM を使用してデフォルト ゾーンを追加します。エージェントは常に、その他すべてのトラステッド シングル サインオン ゾーンより、自身の SSOZoneName を信頼します。

デフォルト: 空（提供されている場合は SM または SSOZoneName）

制限: 複数値

エージェントのシングル サインオン ゾーンの指定

SSOZoneName

Web エージェントがサポートするシングル サインオン セキュリティー ゾーンの（大文字と小文字を区別する）名前を指定します。このパラメータの値は Web エージェントが作成する cookie の名前に先頭に付けられます。これは、それぞれの cookie ドメインと cookie を関連付けるのに役立ちます。このパラメータが空でない場合、SiteMinder は以下の規則を使用して、cookie を生成します。

ZonenameCookiename

デフォルト: 空（ゾーン名として SM を使用します。これは cookie に以下のデフォルトの名前を与えます）

- SMSESSION
- SMIDENTITY
- SMDATA
- SMTRYNO
- SMCHALLENGE
- SMONDENIEDREDIR

制限: 単一値

例: Z1 に値を設定すると以下の cookie が作成されます。

- Z1SESSION
- Z1IDENTITY
- Z1DATA
- Z1TRYNO
- Z1CHALLENGE
- Z1ONDENIEDREDIR

Web エージェントがサポートするシングル サインオン ゾーンの名前を入力するには、SSOZoneName パラメータを使用します。このパラメータでは、大文字と小文字が区別されます。指定されていない場合は、デフォルトで SM に設定されます。SSOZoneName パラメータの値が空ではない場合は、Web エージェントは、以下の命名規則を使用して cookie を生成します。

zone_namecookie_name

ここで、*zone_name* はパラメータ値で、*cookie_name* は作成される cookie の一般名です。

この規則の影響を受ける Cookie には、以下のものがあります。

- SESSION
- IDENTITY
- DATA
- TRYNO
- CHALLENGE
- ONDENIEDREDIR

ユーザが、まだセッションを確立していないシングル サインオン ゾーンで検証される場合は、ポリシー サーバによって返されるセッションの指定が、そのゾーンの新規セッション cookie の作成に使用されます。

新しい cookie が作成されると、ユーザが、単に名前変更することによって別のゾーンからの cookie を交換できないようにするために、そのゾーン パラメータはゾーン名に設定されます。cookie の検証エンジンは、ゾーン名が cookie の名前で使用されているプレフィックスと一致するかどうかを検証します。これが適用されるのは、SESSION cookie と IDENTITY cookie のみです。

Web エージェントのサポート対象のシングル サインオン ゾーンの名前を指定するには、SSOZoneName パラメータにゾーンの名前を追加します。

信頼とフェールオーバの順序

シングル サインオン ゾーンの信頼順序を指定するには、SSOTrustedZone パラメータを使用します。要求の処理時に、Web エージェントは、リストに表示される順に、各ゾーンの SESSION cookie または IDENTITY cookie を検索します。

見つかった cookie はすべて通常どおり検証されます（暗号化解除され、有効なホスト名、シングル サインオン ゾーン名、およびタイムアウトが指定されているかどうかがテストされます）。次に、有効な場合は、トラステッド セッションの番号付きリストに格納されます。認証される前は、ユーザのアクティブ セッション（およびユーザ ID）は、有効なセッションの番号付きリストにおける最初のセッションであると見なされます。

認証中に、Web エージェントはリスト内の最初のセッションを使用して、検証を呼び出します。検証が成功した場合、エージェントは先へ進んでユーザ ID を確立し、アクティブな ID として確認します。検証が失敗した場合、新たな検証呼び出しで以下のセッションが使用されます。検証が成功するか、エージェントが指定のセッション数を使い果たすまで、同じことが繰り返されます。どのセッションでも検証されていない場合、エージェントは通常どおりユーザに認証情報を要求します。

いったん検証されると、エージェントは他のすべてのセッションを無視して、検証済みのセッションのみに対する要求処理の残り部分について処理を進めます。つまり、認証が失敗した場合、ユーザは直ちに認証情報を要求されることになります。要求内の他のすべての既存セッションは使用されません。

付録 A: トラブルシューティング

このセクションには、以下のトピックが含まれています。

LLAWP がすでに実行しているかまたはログ メッセージが適切なログ ファイルに書き込まれていないためにエージェントが起動しない

UNIX で有効

症状:

以下のような問題が 1 つ以上あります。

- LLAWP プロセスがすでに実行されているため、Web エージェントが開始しません。
- Web エージェントは開始しますが、ログ メッセージが 2 番目のエージェント インスタンスのログ ファイルに書き込まれます。

解決方法:

この問題は、同じコンピュータ上の複数のディスクが同じマウント ポイントを使用する場合に発生する場合があります。Web エージェントは、ディレクトリの inode を使用してシステム リソースを割り当てます。また、inode が同じである場合は、リソースの衝突が発生してエラーが表示される結果になります。この問題を修正するには、以下の手順に従います。

1. Web サーバに新しいサブディレクトリを作成します(これにより、一意の inode が作成されます)。
2. Web エージェントの ServerPath パラメータに表示されているパスを変更してそれが新しいサブディレクトリをポイントするようにします。

Web サーバがユーザ名またはパスワードを要求しない

症状:

Web サーバがユーザ名とパスワードを要求しません。

解決方法:

以下の手順を実行します。

- ポリシー サーバに、リソースが正しく定義されていることを確認します。
- Web エージェントが有効であることを確認します。
- レルムとルールが適切に定義されていることを確認します。

Web サーバが認証に失敗する

症状:

Web サーバによりユーザ名とパスワードを要求されるが、認証に失敗します。

解決方法:

以下の手順を実行します。

- ポリシー サーバに、リソースが正しく定義されていることを確認します。
- ユーザがポリシーに結び付けられていることを確認します。
- ポリシーにルールが適切に定義されていることを確認します。

カスタム エラー ページではなくサーバ エラー 500 が表示される

Windows で有効

症状:

サーバ エラーが発生したときに、カスタム エラー メッセージではなく、汎用的な 500 エラーがブラウザに表示されます。

解決方法:

Microsoft Internet Explorer 内で[HTTP エラー メッセージを簡易表示する]オプションを無効にします。

注: 詳細については、Microsoft のドキュメントを参照するか、または http://support.microsoft.com/ に移動します。

設定した属性が Web アプリケーションに反映されない

症状:

設定した属性が Web アプリケーションに反映されません。

解決方法:

次のように、ポリシー サーバに、リソースが正しく定義されていることを確認します。

- CGI スクリプトの場合、アプリケーションから属性名のクエリを実行するときは、属性名の前に **HTTP_** というプレフィックスを付加する必要があります。たとえば、属性 CAN_READ はポリシー サーバで HTTP_CAN_READ として定義する必要があります。
- Java サーブレットの場合、属性名の前に HTTP_ が*付いていない*ことを確認します。
- ASP ページの場合、ALL_HTTP 変数を解析します。

SiteMinder テスト ツールを使用して、[Server Response]グループ ボックスの[Attributes]フィールドに属性が表示されているかどうかを確認します。

エージェントがポリシー サーバを無視するように設定された認証要求を送信する

症状:

エージェントが、無視するように設定された拡張のポリシー サーバに認証要求を送信します。

解決方法:

IgnoreExt の値を確認します。文字列は、スペースは挿入せずにカンマで区切ります。URL にピリオドを含むセクションが複数ある場合、Web エージェントは、必ず、認証リクエストを送信します。

ブラウザが Cookie を送信しない

症状:

最初のサーバでの認証後、Web ブラウザから他の Web サーバに cookie が送信されません。

解決方法:

以下の例のように、シングル サインオンの CookieDomain フィールドに少なくとも 2 つのピリオドが含まれていること、それがサーバ名のサブセットになっていることを確認します。

mycompany.com cookie ドメインは、名前の末尾が .mycompany.com であるすべてのマシンで使用されます。完全なドメイン名を使用しなくてもローカル Web サーバにアクセスできますが、この場合、Web ブラウザは cookie を送信できません。この結果、URL http://www/index.html は Web ページを表示しますが、cookie ドメインが一致しません。そのため、cookie は送信されません（再認証を求められる場合もあります）。

たとえば、www.company.abc.uk のように国名を表すドメインを使用している場合、URL にピリオドが 3 つあることを確認します。また、以下の点にも注意してください。

- ブラウザに、完全修飾ドメイン名が指定されていることを確認します。
- CookieDomain を none に設定した場合、Web エージェントは cookie を作成しますが、それらの cookie は、作成元のサーバ上でのみ有効です。
- CookieDomain を空白のままにした場合、Web エージェントは CookieDomainScope パラメータの値に基づいて、HTTP_HOST ヘッダから cookie ドメインを取得します。
- CookieDomainScope パラメータが 0 に設定されている場合、エージェントは特定のサーバ専用の cookie を作成することなく、そのホストにとって最も具体的な cookie ドメインを追跡します。これは、ホスト myserver.netegrity.com に対応する cookie ドメインが netegrity.com であり、ホスト myserver.metals.ge.com に対応する cookie ドメインが .metals.ge.com であることを意味します。CookieDomainScope が 2 に設定されている場合、cookie ドメインはそれぞれ、.netegrity.com および .ge.com になります。

Solaris/Sun Java System Web エージェントがロードされないか、Web サーバが起動しない

症状:

Solaris/Oracle iPlanet Web エージェントがロードされないか、Web サーバが起動しません。obj.conf ファイルには該当する行が正しく入力されています。

解決方法:

以下のファイル内で、サーバに関するエラー ログを確認します。

/https-*server_name*/logs/errors

WriteLine Failed エラーが表示される

症状:

Windows/Oracle iPlanet Web エージェントを再インストールしたが、"Could not write hostname into the Webagent.conf file, error -1" という WriteLine Failed エラーが表示される。

解決方法:

Web エージェントをアンインストールした後に、Web サーバ サブディレクトリから WebAgent.conf ファイルが削除されたことを確認します。

Solaris/Sun Java System Web エージェントがポリシー サーバと通信しない

症状:

Solaris/Oracle iPlanet Web エージェントが、ロードはされているがポリシーサーバと通信していません。

解決方法:

以下の手順を実行します。

- エージェント設定ファイルに、EnableWebAgent パラメータが記述され、それが yes に設定されていることを確認します。デフォルトの設定値は no です。
- エージェントがポリシー サーバに対して TCP 接続されていることを確認します。Web エージェントとポリシー サーバ間にファイアウォールが構築されている場合、TCP ポート 44441、44442、44443 がファイアウォールによって双方向ブロックされていないことを確認します。

Web エージェントを有効にすると、Apache Web サーバが起動/再起動しない

症状:

Web エージェントを有効にすると、Apache Web サーバが起動または再起動しません。

解決方法:

Web サーバが起動しない場合は、以下を実行します。

- *apache_server_home*/logs/error_log に格納されている Apache エラー ログを確認し、SiteMinder エラーが発生しているかどうかを調べます。Apache Web エージェントは、カーネルが許容するサイズを超える共有メモリまたはセマフォを割り当てようとしていることが考えられます。
- Apache Web エージェントに合わせて Solaris オペレーティング システムを調整します。

  注: 詳細については、「SiteMinder Web エージェント インストール ガイド」を参照してください。
- マルチ ユーザ アカウントを使って Apache Web サーバを起動すると、システム上にセマフォが残ってしまうことがあります。残ってしまったセマフォを削除するには、システムを再起動するか、ipcrm -s コマンドを使用します。Web サーバの起動には、常に同じユーザ アカウントを使用することをお勧めします。

Web サーバが再起動しない場合、restart コマンドは使用しないようにしてください。サーバの再起動には、stop および start コマンドを使用します。

URL に % 文字が含まれている場合に Apache リバース プロキシ サーバが 500 エラーを示す

症状:

私は、リバース プロキシ サーバとして Apache Web サーバを使用しています (ProxyPass ディレクティブは httpd.conf ファイル内で有効です) が、パーセント (%) 文字が含まれている URL を要求すると HTTP 500 エラーが返されます。

解決方法:

エージェント設定オブジェクトまたはローカル設定ファイル内で BadURLChars パラメータの値として使用される文字のリストから %25 を削除します。

Solaris 上の Sun Java System Web エージェントがロードされない

症状:

Solaris 上の Oracle iPlanet Web エージェントがロードされないか、またはポリシー サーバと通信していません。

解決方法:

以下の手順を実行します。

- WebAgent.so.1 の Init ステートメントが、bj.conf ファイル内の、他の Init ステートメントと同じセクションに記述されていることを確認します。以下の例のように、この行では、WebAgent.so.1 ファイルに有効なパスが含まれていること、2 つの関数(SiteMinder Agent と SmRequireAuth)が引用符で囲まれ、(スペースなしの)カンマで区切られていることが必要です。

  ```
  Init fn="load-modules"
  shlib="/usr/iPlanet/servers/https-myserver/config/WebAgent.so.1"
  funcs="SiteMinderAgent,SmRequireAuth"
  ```

- Web エージェントの名前が、ポリシー サーバ上にあるエージェント設定オブジェクトの中で指定されたものと一致していることを確認してください。
- EnableWebAgent パラメータが yes に設定されていることを確認します。
- AuthTrans ステートメントがデフォルトのオブジェクトに記述されており、SiteMinderAgent 関数を参照していることを確認します。 以下の例のように、obj.conf ファイルに別の AuthTrans 行が記述されていて、Web エージェントが動作していない場合は、それらの行をコメント アウトし、WebAgent AuthTrans 関数のみを使用します。

  ```
  AuthTrans fn="SiteMinderAgent"
  ```

- PathCheck ステートメントがデフォルトのオブジェクトに記述されていること、その記述は、オブジェクトにおける他の PathCheck 行の後ろに挿入されていることを確認します。また、以下の例のように、SmRequireAuth 関数を参照していることを確認します。

  ```
  PathCheck fn="SmRequireAuth"
  ```

iPlanet Web サーバで、SSL を使用した基本認証の使用時に空のページが表示される

症状:

iPlanet Web サーバで、SSL を使用した基本認証の使用時に空のページが表示されます。(Microsoft Internet Explorer(MSIE)が SSL バージョン 3(SSLv3)および Transport を処理する方法)

Transport Layer Security(TLS)キープアライブ接続を処理する方法によって、iPlanet Web などの Microsoft 以外の Web サーバとの相互運用性の問題が発生します。SSL(https://)接続を介して Web サーバにアクセスする場合、Internet Explorer に、エラー メッセージまたは空のページが不適切に表示されることがあります。

解決方法:

iPlanet Web サーバ 6.0 SP2 は、この問題を回避するための新機能を導入しました。以下の 2 つの対応策があります。

- サーバの obj.conf ファイルで <Object name="default"> 行のすぐ下に以下の行を追加します。

  ```
  AuthTrans fn="match-browser" browser="*MSIE*" ssl-unclean-shutdown="true"
  ```

  この行は、サーバに対して、MSIE ブラウザから SSLv3 接続を閉じるときに close_notify アラートを送信しないように指示します。close_notify パケットは、SSLv3 および TLS 仕様の必須コンポーネントですが、MSIE では誤って解釈されます。

  注: close_notify パケットは、トランザクションの相手側に接続を閉じることを通知するために、SSLv3 および TLS 接続で使用されます。iPlanet Web サーバに対して close_notify パケットを送信しないように指示すると、MSIE が切断攻撃に対して脆弱になることがあります。

- サーバの obj.conf ファイルで <Object name="default"> 行のすぐ下に以下の行を追加します。

  ```
  AuthTrans fn="match-browser" browser="*MSIE*" keep-alive="disabled"
  ```

  この行は、サーバに対して、Internet Explorer ブラウザのキープアライブ接続を無効にするように指示します。キープアライブ接続を無効にすると、サーバのパフォーマンスが低下することがあります。

フォームベースの認証要求後に Sun Java System Web サーバが再起動する

Solaris で有効

症状:

SiteMinder がフォームを使用して認証要求を行うと、Web サーバはそのコアをダンプし、再起動します。

解決方法:

magnus.conf ファイル内の StackSize パラメータの値を大きくします。

例: StackSize が 131072 に設定されている場合は、その設定を 262144 まで大きくします。

認証要求が失敗する

症状:

Web エージェントによって行われた認証要求が失敗します。

解決方法:

キー ストアが利用可能であることを確認します。

注: 詳細については、ポリシー サーバドキュメントを参照してください。

Web エージェントがクライアント要求を 2 度処理する

SunOne Web Server 6.1 上の SiteMinder Web エージェント r6.x QMR5 以上で有効

症状:

SunOne 6.1 Web サーバ上の Web エージェントを SiteMinder r6.x QMR5（またはそれ以上）にアップグレードした後、Web エージェントが以下の動作の 1 つ以上を示します。

- クライアント要求が Web エージェントによって 2 度処理されます。最初の要求では HTTP 1.0 または 1.1 が使用され、2 度目の要求では HTTP 0.9 が使用されます。
- Web エージェントが、予想外に、要求内の有効な要求変数（rq->vars）を HTTP ヘッダとして設定します。
- カスタム ヘッダで、以前使用されていた HTTP_NAME=*Value* 形式ではなく、NAME=*Value* 形式が使用されるようになり、後者の形式のヘッダが使用されるアプリケーションが動作しなくなりました。

解決方法:

DisableDirectoryList パラメータの値を yes に設定します。

OnAuthAccept ルールおよび OnAuthAcceptText レスポンスを FCC で使用したときにレスポンス テキストが表示されない

症状:

OnAuthAccept ルールが OnAcceptText レスポンスと組み合わされ、ユーザが SiteMinder FCC を使用して認証された場合、レスポンスのテキストが表示されません。

解決方法:

FCC 互換モード(FCCCompatMode)パラメータを no の値(デフォルト)に設定すると、この動作が発生します。値が no である場合、ユーザ認証は、レスポンスがトリガされるフォーム認証情報コレクタ(FCC)で行われますが、FCC が要求されたリソースにユーザをリダイレクトするとテキストが失われます。Web エージェントはユーザのセッションを検証しますが、これは認証イベントでは*ない*ので、OnAcceptText レスポンスは*トリガされません*。

この動作を変更するには、はいに FCCCompatMode パラメータの値を yes に設定します。

カスタム エラー ページが表示されない

Oracle Directory Enterprise Edition（以前の Oracle iPlanet Directory Server Enterprise Edition）で有効

症状:

以下のいずれかの設定パラメータを設定しましたが、代わりにサーバから汎用的なエラー メッセージを受信しました。

CSSErrorFile

ユーザが可能なクロスサイト スクリプティング文字が含まれている URL を開こうとした場合に、ユーザに表示するカスタム エラー メッセージ ファイルまたは URL の場所を指定します。

デフォルト: デフォルトなし

ServerErrorFile

サーバ エラーに遭遇したユーザに対してカスタム エラー ページを表示するように Web エージェントに指示します。このパラメータのファイル パスまたは URL を指定します。

デフォルト: デフォルトなし

解決方法:

以下の手順を実行します。

1. Web サーバ上の instance_name-obj.conf ファイルを開きます。
2. 次の行を検索します。

   ```
   AuthTrans fn="SiteMinderAgent"
   ```

3. 以下の例のように、この行の末尾に UseOutputStreamSize="0" を追加します。

   ```
   AuthTrans fn="SiteMinderAgent" UseOutputStreamSize="0"
   ```

4. ファイルを保存し、Web サーバを再起動します。

付録 B: Web エージェント設定パラメータ

このセクションには、以下のトピックが含まれています。

エージェント設定パラメータ

以下に、Web エージェントの設定パラメータをアルファベット順に示します。

AcceptTPCookie

Web エージェントがサードパーティ(SiteMinder 以外)の Web エージェントによって作成されたセッション(SMSESSION) cookie を受け取ることを可能にします。サードパーティ エージェントは、SiteMinderSDK を使用して、SMSESSION cookie を生成したり読み取ったりします。

デフォルト: デフォルトなし

AgentConfigObject

ローカル エージェント設定ファイル内にエージェント設定オブジェクト(ポリシー サーバに格納された)の名前を定義します。このパラメータはエージェント設定オブジェクトでは*使用されません*。

デフォルト: デフォルトなし

注: このパラメータの値を変更した場合は、変更を適用するために Web サーバを再起動する必要があります。

AgentName

Web エージェントの ID を定義します。エージェント名は、その名前と、エージェントをホストしている各 Web サーバ インスタンスの IP アドレスの間で、マッピングを確立します。

このパラメータに対して値が設定されていない場合、または列挙された値の中の一致を Web エージェントが検索しない場合、Web エージェントは、DefaultAgentName パラメータ内で設定されている値を代わりに使用します。

注: このパラメータは複数の値を持つことができます。エージェント設定オブジェクト内でこのパラメータを設定する場合は、複数値オプションを使用します。ローカル設定ファイルでは、パラメータ名に続いて各値をファイルの個別の行に追加します。

デフォルト: デフォルトなし

制限: 32-127 の範囲内に 7 ビット ASCII 文字が含まれている必要があり、1 つ以上の印刷可能文字が含まれている必要があります。アンパサンド(&)およびアスタリスク(*)文字は含めることができません。大文字と小文字は区別されません。たとえば、MyAgent と myagent という名前は、同じように処理されます。

例: myagent1,192.168.0.0

例: myagent, www.sitea.com

AgentNamesAreFQHostNames

FCC と SCC で Web エージェント名としてターゲット URL の完全修飾ホスト名の使用を有効にします。

デフォルト: No

AgentWaitTime

Low Level Agent Worker Process（LLAWP）が使用可能になるまで、Web エージェントが何秒待つかを指定します。指定した時間を過ぎると、Web エージェントはポリシー サーバに接続しようとします。

このパラメータの設定は、LLAWP 接続に関連するエージェント スタートアップ エラーを解決するのに役立つ場合があります。デフォルト値で設定を開始し、エージェントが正常に開始されるまで、5 秒ずつ間隔を増やすことをお勧めします。
エージェント設定オブジェクトまたは LocalConfig.conf ファイルにこのパラメータを設定することが望ましくない場合は、代わりに WebAgent.conf ファイルに設定することもできます。

デフォルト: 5

例: プライマリおよびセカンダリのポリシー サーバを使用している場合は、30 から 40 の値で開始してみてください。

制限: なし

注: ネットワーク遅延の問題が発生している場合、このパラメータをフレームワーク エージェントで使用できます。

AllowCacheHeaders

Web エージェントで、保護されたリソースに対するリクエストを Web サーバに渡す前に、リクエストから以下のキャッシュ関連 HTTP ヘッダを削除するか動かを指定します。

- if-modified-since
- if-none-match

この設定は、ブラウザでキャッシュされたページをするかどうかに影響しますが、自動許可されたリソース(IgnoreExt パラメータの値によって一致したリソースを含む)には影響しません。自動許可されるリソースのキャッシュは、Web サーバおよびブラウザの設定によって決定されます。

このパラメータには、以下の値を設定できます。

- Yes -- エージェントはキャッシュ関連の HTTP ヘッダを削除しません。セッションを検証するためには SMSESSION cookie が引き続き追跡されます。セッションが期限切れになると、Web エージェントは、キャッシュに格納されているリソースで、if-modified-since HTTP ヘッダに示される時間以降 Web サーバによって変更されていないリソースについて、更新された SMSESSION cookie を 304「変更なし」のレスポンスで送信します。

重要: このパラメータが yes に設定された場合、Web サーバ上のアプリケーションによってパーソナライズされたページで、適切なキャッシュ制御ヘッダを設定されていないページは、ブラウザまたは HTTP に一時的にキャッシュされる可能性があります。これは、予期しない動作を引き起こす可能性があり、ブラウザが機密データをディスクに保存することを可能にします。

- No -- エージェントは、保護されているリソースのリクエストからのみ、キャッシュ関連の HTTP ヘッダを削除します。Web サーバでは、リクエストを無条件として取り扱い、キャッシュ検証操作は実行されません。
- None -- Web エージェントは、保護されているリソースおよび保護されていないリソースについて、キャッシュ関連のヘッダをすべて削除します。

終了したセッションの場合、AllowCacheHeaders パラメータの値にかかわらず、ブラウザはキャッシュされたコンテンツを使用しません。

このパラメータの設定は以下のパラメータに影響します。

- LogOffURI -- LogOffURI パラメータも使用している場合、AllowCacheHeaders パラメータの値を no に設定することをお勧めします。そうしないと、キャッシュされたログオフ ページがユーザに提示され、そのセッションが適切に終了されません。

デフォルト: No

制限: Yes、No、None

AllowLocalConfig

ローカル設定ファイルを読み取って Web エージェントの設定パラメータを取得するように、ポリシー サーバ上のエージェント設定オブジェクトに指示します。このパラメータはエージェント設定オブジェクトでのみ使用されます。

このパラメータの複数の値をエージェント設定オブジェクトに追加して、ローカル設定ファイルで変更可能なパラメータを制御することもできます。複数の値がこのパラメータに設定される場合、それらは以下の順序で処理されます。

- 同じエージェント設定オブジェクト内の他の設定パラメータに yes の値が存在する場合は、yes が優先されます。他の設定パラメータはローカル設定ファイル内で変更できる唯一のパラメータです。
- 同じエージェント設定オブジェクト内の他の設定パラメータに no の値が存在する場合は、no が優先されます。これにより、エージェント設定オブジェクトの他の設定パラメータのいずれも削除せずに、迅速かつ完全にローカル設定を無効にすることができます。
- yes と no の複数の値が同じエージェント設定オブジェクトに存在する場合は、no が優先されます。これにより、エージェント設定オブジェクトの他の設定パラメータのいずれも削除せずに、迅速かつ完全にローカル設定を無効にすることができます。

デフォルト: No

例: yes、EnableAuditing、EnableMonitoring（前の 2 つのパラメータについてのみ、ローカル制御が可能です）

AppendIISServerLog

認証されたユーザ名と IIS サーバ ログに対する SiteMinder トランザクション ID を個別の行に追加するように Web エージェントに指示します。

デフォルト: No

注: このパラメータは IIS 6.0 Web エージェントにのみ適用されます。

詳細情報:

BadCSSChars

Web エージェントがクロスサイト スクリプティング攻撃の可能性として解釈する URL 文字を指定します。

デフォルト: <、'、>

制限:

- 実際に文字を指定するか、その文字の URL エンコード形式を入力することができます。たとえば、文字 a を入力するか、または、そのエンコード値である %61 を入力できます。
- 最大 4096 文字まで指定できます（区切り文字として使用するカンマを含みます）。
- また、文字の範囲をハイフンで区切って指定することもできます。構文は *starting_character-ending_character* です。たとえば、一連の文字として「a-z」と入力できます。
- 引用符（"）を URL エンコード値 %22 でを指定します。ASCII は使用できません。

BadFormChars

フォーム上で出力として使用する前に、Web エージェントがリテラル HTML 文字としてエンコードする文字を指定します。ディレクティブの置換文字列のみが未処理の HTML としてエンコードされます。フォーム テンプレート（login.fcc テンプレートなど）のソース行は変更されません。ソース行を変更せずに維持することで、スクリプティング コードを含む動的なデータがフォーム内のデータとしてブラウザに戻されることを防ぎます。

デフォルト: 無効（リテラル エンコーディングなし）

例: <、>、&、%22

制限:

- 次の文字のみが許可されます: <、>、&、%22
- 実際に文字を指定するか、その文字の URL エンコード形式を入力することができます。たとえば、文字 a を入力するか、または、そのエンコード値である %61 を入力できます。
- 最大 4096 文字まで指定できます（区切り文字として使用するカンマを含みます）。
- また、文字の範囲をハイフンで区切って指定することもできます。構文は *starting_character-ending_character* です。たとえば、一連の文字として「a-z」と入力できます。
- 引用符（"）を URL エンコード値 %22 でを指定します。ASCII は使用できません。

BadQueryChars

Web エージェントによって、URL のクエリ文字列部分（'?' の後）で禁止される文字を指定します。

デフォルト: 空（クエリ文字列に禁止される文字はありません）

制限:

- 実際に文字を指定するか、その文字の URL エンコード形式を入力することができます。たとえば、文字 a を入力するか、または、そのエンコード値である %61 を入力できます。
- 最大 4096 文字まで指定できます（区切り文字として使用するカンマを含みます）。
- また、文字の範囲をハイフンで区切って指定することもできます。構文は *starting_character-ending_character* です。たとえば、一連の文字として「a-z」と入力できます。
- 引用符（"）を URL エンコード値 %22 でを指定します。ASCII は使用できません。

例: %25 は、クエリ内の URL エンコード文字をブロックします。

BadUrlChars

URL リクエストに使用できない文字シーケンスを指定します。Web エージェントは、このパラメータよって指定された文字シーケンスに対して、「?」文字の前にある URL 内の文字を確認します。指定された文字のいずれかが見つかった場合、Web エージェントは要求を拒否します。

以下の文字を指定できます。

- 円記号(¥)
- ダブルスラッシュ(//)
- ピリオドとスラッシュ(./)
- スラッシュとピリオド(/.)
- スラッシュとアスタリスク(/*)
- アスタリスクとピリオド(*.)
- ティルダ(~)
- %2d
- %20
- %00-%1f
- %7f-%ff
- %25

複数の値はカンマで区切ります。スペースは*使用しないでください*。

無効な URL 文字は、その前に疑問符(?)が付いている場合にのみ、CGI パラメータの中で使用できます。

デフォルト: <、>、&、;

制限:

- 実際に文字を指定するか、その文字の URL エンコード形式を入力することができます。たとえば、文字 a を入力するか、または、そのエンコード値である %61 を入力できます。
- 最大 4096 文字まで指定できます(区切り文字として使用するカンマを含みます)。
- また、文字の範囲をハイフンで区切って指定することもできます。構文は *starting_character-ending_character* です。たとえば、一連の文字として「a-z」と入力できます。
- 引用符(")を URL エンコード値 %22 でを指定します。ASCII は使用できません。

CacheAnonymous

Web エージェントが匿名のユーザ情報をキャッシュするかどうかを指定します。このパラメータは、たとえば以下の状況に対して設定できます。

- Web サイトのユーザのほとんどが匿名ユーザで、それらのユーザのセッション情報を格納したい場合。
- 登録ユーザと匿名ユーザの両方が Web サイトにアクセスする場合。

  匿名ユーザの情報のみでキャッシュが満杯になり、登録ユーザ用の領域がなくなってしまう可能性がある場合は、このパラメータを無効にすることをお勧めします。

デフォルト: No

注: このパラメータの値を変更した場合は、変更を適用するために Web サーバを再起動する必要があります。

CCCExt

cookie プロバイダ認証情報コレクタの MIME タイプを指定します。

このパラメータは以下のパラメータに影響します。

- CookieProvider

デフォルト: .ccc

ConformToRFC2047

Web エージェントが RFC 2047 に準拠しているかどうかを示します。このパラメータが見当たらない場合、Web エージェントはデフォルトの動作に従います。

デフォルト: yes

ConstructFullPwsvcUrl

ユーザをパスワード サービス アプリケーションにリダイレクトするための完全修飾ドメイン名を備えた URL を生成するように Web エージェントに指示します。これにより、特定の Web サーバ上のパスワード サービス アプリケーションをホストすることができます。Web エージェントは、以下の例のような URL を生成します。

```
HTTP://my.server.com:80/path/to/passwordservices.cgi
```

完全修飾 URL が使用されない場合、Web エージェントは、パスワード サービス アプリケーションが同じ Web サーバ上でホストされ、リダイレクト用の相対 URL を使用すると想定します。

デフォルト: No

CookieDomain

Web エージェントのインストール時に指定した Web エージェントの cookie ドメインを定義します。この場合、完全修飾ドメイン名を指定する必要があります。つまり、ドメインには、少なくとも 2 つのピリオドが含まれている必要があります。たとえば、.myorg.com という cookie ドメインは以下のサーバと一致します。

- w1.myorg.com
- w2.myorg.com
- w3.sales.myorg.com

このドメイン内のすべての Web サーバは、ユーザのブラウザとの間で cookie を送受信できます。同一 cookie ドメイン内のサーバは、cookie を使用してユーザの認証情報を確認します。

デフォルト: 空白

例: .mycompany.com

注: この値では、大文字と小文字が区別されます。

CookieDomainScope

ドメイン名のセクション（ピリオドで区切られた領域）の数を指定します。

デフォルト: 0

例: division.myorg.com という cookie ドメインの、server.division.myorg.com という名前のドメインの場合は、CookieDomainScope を 3 に設定します。

CookiePath

以下のセカンダリ エージェント ブラウザ cookie の cookie パスを指定します。

- xxSESSION
- xxIDENTITY
- xxDOMINODATA
- xxCHALLENGE（SSL_CHALLENGE_DONE を含む）
- xxDATA
- xxSAVEDSESSION

たとえば、このパラメータを /BasicAuth に設定すると、前述のリスト内のセカンダリ エージェントはすべて、パスとして /BasicAuth を使用して作成されます。指定しない場合は、デフォルト値が使用されます。

4.x のエージェントとの後方互換性を維持するため、CookiePath は認証情報 cookie（xxxxCRED など）には追加*されません*。

以下の cookie は*常に*ルート パス（/）を使用します。

- ONDENIEDREDIR
- TRYNO

CookiePathScope パラメータが 0 より大きい場合は、CookiePath パラメータの設定が上書きされます。

デフォルト: /（ルート）

CookiePathScope

以下のセカンダリ エージェント cookie の cookie パスの範囲を指定します。

- xxSESSION
- xxIDENTITY
- xxDOMINODATA
- xxCHALLENGE（SSL_CHALLENGE_DONE を含む）
- xxDATA
- xxSAVEDSESSION

CookiePathScope が 0 より大きい場合は、CookiePath パラメータの設定が上書きされます。

デフォルト: 0

CookieProvider

cookie プロバイダとして動作している Web エージェントがある Web サーバの URL を(完全修飾ドメイン名を使用して)を指定します。cookie プロバイダ名には、拡張子 .ccc が必要です。

- IIS、Sun Java System、Domino Web サーバでは、以下の URL 構文を使用します。

  http://*server.domain:port*/siteminderagent/SmMakeCookie.ccc

- Apache および Apache ベースの Web サーバでは、以下の URL 構文を使用します。

  http://*server.domain:port*/SmMakeCookie.ccc

このパラメータは以下のパラメータに影響します。

- CCCExt
- SessionUpdatePeriod

デフォルト: デフォルトなし

例: (IIS、Sun Java System、および Domino Web サーバ)
http://server1.myorg.com:80/siteminderagent/SmMakeCookie.ccc

例: (Apache および Apache ベースの Web サーバ)
http://server1.myorg.com:80/SmMakeCookie.ccc

CookieValidationPeriod

エージェントがセッション cookie 受け取る期間を(秒単位で)指定します。この期間を過ぎると、セッション cookie は受け取られません。このフィールドが使用されていないか、ゼロに設定されている場合、アイドル タイムアウトおよび最大セッション タイムアウト値に達すると、セッション cookie は期限切れになります。

デフォルト: 空白

CSSChecking

Web エージェントが、実行可能スクリプトの一部である場合がある、(BadCSSChars パラメータのリストによって定義された)エスケープ文字とエスケープされていない文字の URL (クエリ文字列を含む)を確認するかどうかを指定します。

デフォルト: yes

CSSErrorFile

ユーザが可能なクロスサイト スクリプティング文字が含まれている URL を開こうとした場合に、ユーザに表示するカスタム エラー メッセージ ファイルまたは URL の場所を指定します。

デフォルト: デフォルトなし

注: Oracle Directory Enterprise Edition（以前の Sun Java System Directory Server Enterprise Edition）のバージョン 7.x 以降でこのパラメータを設定する場合、Web サーバ上の instance_name-obj.conf ファイルを変更します。

詳細情報:

カスタム エラー処理の指定 (P. 158)
エラー処理をセットアップする方法 (P. 160)
カスタム エラー ページが表示されない (P. 356)

Custom401ErrorFile

401（権限不足）のブラウザ エラーが発生した場合に表示する、カスタマイズされた HTML ページを指定します。

デフォルト: デフォルトなし

CustomIpHeader

Web エージェントがリクエスタの IP アドレスを検索する HTTP ヘッダを指定します。このパラメータに値が設定されていない場合は、デフォルトは空の文字列になります。最大長はないため、値は、有効な HTTP ヘッダ値を含む任意の文字列（たとえば、HTTP_ORIGINAL_IP）にできます。

デフォルト: デフォルトなし

DecodeQueryData

ポリシー サーバをコールする前に、Web エージェントが URL 内のクエリデータをデコードするかどうかを指定します。環境内で以下のタスクのいずれかを実行する必要がある場合は、このパラメータを yes に設定します。

- 正しい文字列に対してルール ファイラが機能していることを保証する必要がある場合
- クエリ文字列内のデータに対して書き込みルールが機能していることを保証する必要がある場合

デフォルト: No

DefaultAgentName

AgentName パラメータに指定されたエージェント名がない IP アドレスまたはインターフェースのリクエストを受け取った場合に Web エージェントが使用する名前を定義します。

仮想サーバを使用している場合、各仮想サーバに別々の Web エージェントを定義する代わりに、DefaultAgentName を使用することによって、SiteMinder 環境を迅速にセットアップできます。

重要: DefaultAgentName パラメータに値を指定しない場合は、AgentName パラメータにすべてのエージェント ID をリストする必要があります。そうしないと、ポリシー サーバは Web エージェントにポリシーを結び付けることができません。

デフォルト: デフォルトなし

制限: 32-127 の範囲内に 7 ビット ASCII 文字が含まれている必要があり、1 つ以上の印刷可能文字が含まれている必要があります。アンパサンド(&)およびアスタリスク(*)文字は含めることができません。大文字と小文字は区別されません。たとえば、MyAgent と myagent という名前は、同じように処理されます。

DefaultHostName

HOST ヘッダに対する値を定義します。HTTP バージョン 0.9 または 1.0 リクエスト(HOST ヘッダなし)を送信するテスト/パフォーマンス ツールを使用するには、エージェント設定オブジェクトまたは LocalConfig.conf ファイルにこのパラメータを追加します。このパラメータが設定されていない場合、Web エージェントでは HTTP 1.1 リクエストのみを受け入れます。

デフォルト: なし(空白)

例: webserver.example.com

詳細情報:

HOST ヘッダを送信しないテスティング ツールへの対応 (P. 211)

DefaultPassword

プロキシ ユーザとして IIS リソースにアクセスするために使用される関連する Windows ユーザのためのデフォルトのパスワードを指定します。

重要: このパラメータを暗号化する場合は、エージェント設定オブジェクトの中心に設定します。このパラメータがローカル設定ファイルに設定されると、それは暗号化されず、安全性が低下します。

デフォルト: デフォルトなし

注: このパラメータは IIS Web エージェントにのみ適用されます。

DefaultUserName

プロキシ ユーザとして IIS リソースにアクセスするために使用される Windows ユーザの名前を指定します。SiteMinder で保護されている IIS Web サーバのリソースにアクセスするユーザが、必要なサーバ アクセス権限を持つユーザではないことがあります。たとえば、UNIX システム上の LDAP ユーザ ディレクトリに格納されているユーザは、IIS Web サーバの Windows システムへのアクセス権限がない場合があります。

Web エージェントが SiteMinder によってアクセスを許可されたユーザのプロキシユーザ アカウントとして機能するためには、NT 管理者によって割り当てられたこの NT ユーザ アカウントを使用する必要があります。

デフォルト: デフォルトなし

注: このパラメータは IIS Web エージェントにのみ適用されます。

DeleteCerts

Web エージェントで使用されなくなったときに Stronghold サーバに格納されている証明書を削除するかどうかを指定します。

デフォルト: No

注: このパラメータは Apache Web エージェントにのみ適用されます。

DisableAuthSrcVars

Web エージェントが以下のデフォルト SiteMinder 認証ソースの HTTP ヘッダ変数を無効にするかどうかを指定します。

- SM_AUTHDIRNAME
- SM_AUTHDIRSERVER
- SM_AUTHDIRNAMESPACE
- SM_AUTHDIROID

注: 個々の変数を無効にすることはできません。いくつかの変数のカテゴリのみ無効にできます。

デフォルト: No

DisableDirectoryList

最初に認証情報を要求せずに、ユーザがディレクトリの内容を表示または参照することを Web エージェントが認めるかどうかを指定します。これは、以下の条件が すべて当てはまる場合に発生します。

- レルムがルートリソース(/)を保護するように設定されている。
- ディレクトリのデフォルト Web ページ(index.html など)が名前変更または削除されている。

デフォルト: No

注: このパラメータは Sun Java System エージェントにのみ適用されます。

DisableDNSLookups

Web エージェントが DNS の検索を実行することを防ぎます。

重要: このパラメータの値を yes に設定したときは、cookie ベースの機能が適切に動作するように、完全修飾ドメイン名を使用する必要があります。

DisableDotDotRule

Web エージェントがスラッシュ(/)で区切られた 2 つの期間が含まれる URL へのアクセスをブロックするかどうかを指定します。

このパラメータの設定は以下のパラメータに影響します。

- IgnoreExt

デフォルト: なし(ルールが適用されます)

DisableSessionVars

Web エージェントが以下のデフォルト SiteMinder ユーザ セッションの HTTP ヘッダ変数を無効にするかどうかを指定します。

- SM_SERVERSESSIONID
- SM_SERVERSESSIONSPEC
- SM_SERVERIDENTITYSPEC
- SM_SESSIONDRIFT
- SM_TIMETOEXPIRE

注: 個々の変数を無効にすることはできません。いくつかの変数のカテゴリのみ無効にできます。

デフォルト: No

DisableUserNameVars

Web エージェントが以下のデフォルトの SiteMinder ユーザ名の HTTP ヘッダ変数を無効にするかどうかを指定します。

- SM_USER
- SM_USERDN
- SM_DOMINOCN

注: 個々の変数を無効にすることはできません。いくつかの変数のカテゴリのみ無効にできます。

デフォルト: No

詳細情報:

デフォルトの HTTP ヘッダ変数の無効化 (P. 139)

DominoDefaultUser

Domino Web エージェントが SiteMinder が事前に Domino サーバへの別のディレクトリに対して認証したユーザを識別する名前を指定します。

重要: このパラメータがローカル設定ファイルに格納される場合は、暗号化する必要があります。このパラメータを暗号化するには、encryptkey ツールを使用します。パラメータを変更する場合は、ローカル設定ファイルを直接編集すしないでください。

デフォルト: デフォルトなし

注: このパラメータは Domino Web エージェントのみ適用されます。

DominoLegacyDocumentSupport

Web エージェントが Domino 環境内の保護されている Lotus Notes ドキュメントに対するユーザ要求を処理する方法を指定します。このパラメータを yes に設定すると、要求されたドキュメントに対してのみ、ユーザに ReadForm 許可が与えられます。

デフォルト: No

注: このパラメータは Domino Web エージェントのみ適用されます。

DominoLookUpHeaderForLogin

ユーザがリソースへのアクセスを要求した場合に、Domino Web エージェントはそのユーザが Domino ユーザ ディレクトリ内で一意であるかあいまいであるかを、Domino Web サーバに問い合わせます。これは、リソースへのアクセスを要求するユーザの名前がユーザ ディレクトリ内の他のユーザと同じである場合に役立ちます。

デフォルト: No

注: このパラメータは Domino Web エージェントのみ適用されます。

DominoMapUrlForRedirect

フォーム認証情報コレクタ(FCC)にリダイレクトするために、URL を Domino サーバの表現から URL フレンドリ名にマップ(正規化)するように Web エージェントに指示します。FCC は、要求された Domino リソースに対する要求を処理することができます。このパラメータがない場合は、デフォルトの動作が発生します。このパラメータが no の場合、Web エージェントは URL をマップせず、元の Domino サーバの表現を使用して FCC リダイレクトを実行します。

また、DominoNormalizeUrls パラメータも yes に設定する必要があります。そうしないと、URL は正規化されません。

デフォルト: yes

注: このパラメータは Domino Web エージェントのみ適用されます。

DominoNormalizeUrls

SiteMinder Web エージェントが、フォーム認証情報コレクタにリダイレクトする前に、Domino の URL を URL フレンドリ名に変換するかどうかを指定します。

Domino の URL を変換するためには、MapUrlsForRedirect パラメータも yes に設定する必要があります。

DominoNormalizeUrls パラメータが no の場合は、MapUrlsForRedirect パラメータが yes に設定されていても、URL は正規化されません。

重要: DominoNormalizeUrls パラメータを no に設定した場合、Notes データベース内の個々のドキュメントを保護することはできません。Domino Web サーバのデータベース全体またはサブディレクトリのみを保護することができます。

デフォルト: yes

注: このパラメータは Domino Web エージェントのみ適用されます。

DominoSuperUser

Domino サーバ上のすべてのリソースへのアクセス権があるユーザを識別し、SiteMinder に正常にログインしたすべてのユーザが Domino SuperUser として Domino にログインすることを確認します。

この値は暗号化できます。

このパラメータは以下のパラメータに影響します。

- SkipDominoAuth

デフォルト: デフォルトなし

注: このパラメータは Domino Web エージェントのみ適用されます。

DominoUseHeaderForLogin

SiteMinder のヘッダの値を Domino Web サーバに渡すよう Domino Web エージェントに指示します。Domino サーバはそのヘッダのデータを使用して、自らのユーザ ディレクトリの中に存在しているユーザを識別します。

デフォルト: デフォルトなし

注: このパラメータは Domino Web エージェントのみ適用されます。

EnableAuditing

ユーザ セッション キャッシュに格納される正常に行われたすべてのユーザ許可に関する情報を Web エージェントが記録するかどうかを指定します。ユーザ許可を有効にすると、Web エージェントがポリシー サーバへ問い合わせを行わずにキャッシュの情報を使用しても、ユーザ許可情報は記録されます。ユーザがリソースにアクセスすると、Web エージェントはユーザ名とアクセス情報を固有の Web サーバ ログファイルに記録します。

デフォルト: No

EnableFormCache

フォーム テンプレート キャッシュを制御します。このパラメータを yes に設定すると、フォーム認証のパフォーマンスが向上します。キャッシュを無効にするには、このパラメータを no に設定します。

デフォルト: yes

EnableIntroscopeApiSupport

SiteMinder Web エージェントに関する情報を収集し、プラグインを使用して CA Wily Introscope に送ります。このパラメータは以下の設定を使用します。

- yes に設定されたとき、Wily プラグインは、データを収集するために API をコールします。
- no に設定されたとき、Wily プラグインはデータを備えた HTTP ヘッダを作成します。
- both に設定されたとき、Wily プラグインは API をコールし、かつデータを備えた HTTP ヘッダを作成します。
- none に設定されたとき、データは収集されません。

デフォルト: no

制限: yes、both、no、none

例:（HTTP ヘッダ）sm-wa-perf-counters = *server_name*.example.com:6180,86117203,86118343,1,0,0,1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,1125,0,15,1,1,750,750,

EnableMonitoring

SiteMinder Web エージェントが監視情報をポリシー サーバに送信するかどうかを指定します。

デフォルト: No

EnableOtherAuthTrans

SiteMinder で他の AuthTrans 機能の使用を許可します。

デフォルト: No

注: このパラメータは Sun Java System エージェントにのみ適用されます。

EnableWebAgent

Web エージェントをアクティブにし、それがポリシー サーバと通信することを可能にします。すべての設定パラメータの変更を完了してから、このパラメータを yes に設定します。

デフォルト: No

注: このパラメータはローカル設定ファイルでのみ使用されます。

EncryptAgentName

URL により、ユーザがフォーム、SSL、または NTLM 認証情報コレクタへリダイレクトされる場合に Web エージェントがその URL に自らの名前を追加するときに、エージェント名を暗号化するかどうかを制御します。このパラメータは、認証情報コレクタが URL を受け取った時点で、その名前を復号化するかどうかを制御する役割も果たします。

デフォルト: yes

EnforceRealmTimeouts

Web エージェントが、ユーザがアクセスする最初のレルムのセッション タイムアウト値をユーザがシングル サインオンを使用してアクセスする後続のレルムのセッション タイムアウト値で上書きするかどうか決定します。このパラメータを yes に設定した場合、Web エージェントは SMSESSION cookie を検証する際に返されたセッション タイムアウトを参照し、ユーザの検証を行う後続のレルムに関しては、そのタイムアウト値を優先します。このパラメータを設定した場合、エージェントは元のログイン セッションのタイムアウトを優先します。ユーザが新しいレルムへ移動した場合、Web エージェントは、後続のレルムではなく、最初のレルムから取得したアイドル タイムアウトまたはセッション タイムアウトを適用します。

デフォルト: No

ExpiredCookieURL

(省略可) セッション cookie の期限切れ後にエージェントがユーザをリダイレクトする先の URL を指定します。セッションの作成日も CookieValidationPeriod も設定されていない場合、エージェントはこの設定を無視し、cookie を通常どおり処理します (後方互換性)。

ExpireForProxy

クライアントがコンテンツ（ページおよび潜在的なヘッダまたは cookie）をキャッシュしないようにします。このパラメータの値が Yes に設定された場合、Web エージェントは以下の HTTP ヘッダのいずれかを HTTP レスポンスに挿入します。

- Expires
- Cache-control

コンテンツをキャッシュしない場合、後続の要求は引き続き転送されます。

ExpireForProxy パラメータが yes に設定された場合、Web エージェントでは、適切な ProxyHeaders<*suffix_name*> パラメータに指定された文字列を、エージェントが実行したリクエストの種類に基づいて、HTTP レスポンスに挿入します。

HTTP/1.1 リクエストの場合、エージェントは、以下のパラメータの値をヘッダとしてレスポンスに挿入します。

- ProxyHeadersAutoAuth
- ProxyHeadersProtected
- ProxyHeadersUnprotected

HTTP/1.0 リクエストの場合、エージェントは、以下のパラメータの値をヘッダとしてレスポンスに挿入します。

- ProxyHeadersAutoAuth10
- ProxyHeadersProtected10
- ProxyHeadersUnprotected10

デフォルト: No

注: このパラメータ名には 'proxy' という単語が含まれていますが、このパラメータの設定は、Web ブラウザ、またはこのパラメータ設定を使用する SiteMinder エージェントが動作する Web サーバに接続するすべてのクライアントの動作にも影響があります。

詳細情報:

プロキシ サーバの背後にあるエージェントの設定 (P. 213)
Cache-Control ヘッダ設定と ExpireForProxy ヘッダ設定のカスタマイズ (P. 216)
プロキシ ヘッダの使用に関する注意事項 (P. 219)

FCCCompatMode

4.x の Web エージェントまたはサード パーティのアプリケーションによって保護されているリソースに対してフォームを提供するよう FCC/NTC を有効にします。

注: SMUSRMSG は、FccCompatMode が yes に設定されている場合のみ、カスタム認証方式でサポートされます。
デフォルト: (従来のエージェント) Yes

デフォルト: (フレームワーク エージェント) No

重要: このパラメータを no に設定すると、Netscape ブラウザのバージョン 4.x のサポートが削除されます。

詳細情報:

フォームの認証要求に関する SM_AGENT_ATTR_USRMSG レスポンスの使用 (P. 152)
リダイレクト URL のクエリ文字列暗号化 (P. 236)
混在環境での FCC と NTC の使用 (P. 269)

FCCExt

IIS または Domino Web サーバ上の認証情報コレクタに対する MIME 値を指定します。

デフォルト: .fcc

詳細情報:

IIS と Domino の各 Web サーバでのクレデンシャル コレクタのセットアップ (P. 266)

FCCForceIsProtected

Web エージェントはポリシー サーバに対しもう一度 IsProtected 呼び出しを行うかどうかを指定します。この 2 回目の呼び出しによってレルム コンテキストが確立され、Web エージェントはユーザのログインを許可し、保護されているリソースにアクセスできるようにします。

このパラメータを no に設定すると、Web エージェントはポリシー サーバへの IsProtected の最初の呼び出しから取得されたレルム情報を代わりに使用します。

デフォルト: yes

詳細情報:

FCC レルム コンテキスト確認の無効化によるパフォーマンスの向上 (P. 276)

ForceCookieDomain

ドメインが指定されていない URL リクエスト内や IP アドレスのみで構成されている URL リクエスト内のホスト名に cookie ドメインを追加するように Web エージェントに強制します。このパラメータを ForceFQHost パラメータと組み合わせると、機能を追加できます。

デフォルト: No

詳細情報:

cookie ドメインの強制 (P. 113)
完全修飾ドメイン名の強制 (P. 116)

ForceFQHost

完全修飾ドメイン名を使用するように Web エージェントに強制します。このパラメータは設定されたドメイン ネーム システム(DNS)サービスを使用して、URL リクエストの中に存在しているホスト名に対して、cookie ドメインを強制的に追加します。エージェントではなく DNS サービスを使用します。Web エージェントは、URL の一部が含まれたリクエストを受信すると、元の URI に指定されている同じ転送先リソースにそのリクエストをリダイレクトします。リダイレクト要求では、完全修飾ホスト名が使用されます。完全修飾ホスト名は、設定されている DNS サービスを使用して Web エージェントが決定するものです。このパラメータを ForceCookieDomain パラメータと組み合わせて使用すると、機能を追加できます。

デフォルト: No

例: Web エージェントが http://host1/page.html から要求を受け取ると、それは http://host1.myorg.com/page.html で応答します。Web エージェントが http://123.113.12.1/page.html などの要求を受け取ると、それは http://host1.myorg.com/page.html で応答します。

注: これらの例のように変換されるのは、適切な DNS 検索テーブルが設定されている場合のみです。部分的なドメインが入力された場合、DNS 検索がそのドメインを解決できるかどうかによって、結果は異なります。解決の結果が無効なホストである場合、エラーになります。 多くの場合、そのような要求は Web サーバに届くことすらできません。

詳細情報:

cookie ドメインの強制 (P. 113)
完全修飾ドメイン名の強制 (P. 116)

ForceIISProxyUser

Web エージェントが IIS プロキシ アカウントを使用して、IIS Web サーバ上の要求されたリソースへのアクセスを、IIS Web サーバにアクセスするのに十分な権限が通常は不足しているユーザに許可するかどうかを指定します。

このパラメータは以下のパラメータに影響します。

- DefaultUserName
- DefaultPassword

デフォルト: No

注: このパラメータは IIS Web エージェントにのみ適用されます。

詳細情報:

IIS Web エージェントで REMOTE_USER 変数を取り込む方法 (P. 133)
IIS プロキシ ユーザ アカウントの使用 (IIS のみ) (P. 163)

FormCacheTimeout

オブジェクトをキャッシュに保管しておくことのできる期間を秒数で指定します。この期間を過ぎると、そのオブジェクトは無効と見なされます。タイムアウト間隔を過ぎると、フォーム テンプレート ファイルの日時と、キャッシュ オブジェクトが作成された時刻が比較されます。キャッシュに保管されたオブジェクトがシステムのディスク上のファイルよりも新しい場合、タイムアウトはリセットされて、以下のタイムアウトまでオブジェクトは保管されます。それ以外の場合、オブジェクトはキャッシュから削除されます。

デフォルト: 600

詳細情報:

フォーム キャッシュの設定 (P. 275)

GetPortFromHeaders

Web サーバ サービス構造からポート番号を取得する代わりに、HTTP HOST リクエスト ヘッダからポート番号を取得するように Web エージェントに指示します。

デフォルト: No

注: このパラメータは、Apache Web エージェントにとって必須です。

詳細情報:

ポート番号に関する HTTP HOST 要求の使用 (P. 69)

HostConfigFile

トラステッド ホスト コンピュータがポリシー サーバに正常に登録された後に作成される SMHost.conf ファイル(IIS 6.0 または Apache のエージェント内)のパスを指定します。コンピュータ上のすべての Web エージェントが SMHost.conf ファイルを共有します。

デフォルト: デフォルトなし

注: このパラメータを変更した場合は、変更を適用するために Web サーバを再起動する必要があります。

詳細情報:

フレームワーク エージェントの WebAgent.conf ファイル (P. 33)
変更時にサーバの再起動を必要とするパラメータ (P. 25)
ローカル設定ファイルのみにあるパラメータ (P. 35)

HTTPHeaderEncodingSpec

HTTP ヘッダ値およびすべてのカスタム HTTP-COOKIE レスポンスのエンコードのために、Web エージェントが使用する仕様を選択します。このパラメータの値では以下の構文が使用されます。

encoding_spec, wrapping_spec

ラッピング仕様(RFC-2047)を含めることはオプションですが、使用することをお勧めします。

デフォルト: デフォルトなし(ブランクにした場合、Web エージェントではラッピングのない UTF-8 エンコーディングが使用されます)

例: Shift-JIS、RFC-2047

詳細情報:

HTTP ヘッダのエンコード仕様の設定 (P. 150)

HttpsPorts

ユーザが Web サーバへの SSL 接続を使用しているかどうかを Web エージェントがリスンする安全なポートを指定します。このパラメータの値を指定する場合、安全な要求を提供するすべての Web サーバの対象となるすべてのポートを含める必要があります。値を指定しなかった場合、Web エージェントは HTTP スキームをサーバのコンテキストから読み取ります。

サーバが、(HTTPS を HTTP へ変換する) HTTPS アクセラレータの背後にある場合、ブラウザはその要求を SSL 接続として扱います。

デフォルト: 空白

例: 80

例: (複数のポート) 80,8080,8083

詳細情報:

HTTPS ポートの定義 (P. 155)
SiteMinder リバース プロキシ展開の考慮事項 (P. 224)

IdleTimeoutURL

セッションのアイドル タイムアウトが発生したときに、Web エージェントがユーザをリダイレクトする先の URL を指定します。

例: http://example.mycompany.com/sessionidletimeoutpage.html

注: IdleTimeoutURL は、非永続セッションにのみ使用してください。永続セッションに対して設定した場合は無効になります。

詳細情報:

セッション タイムアウト後のユーザのリダイレクト (P. 109)

IgnoreCPForNotprotected

保護されていないリソースの要求に関して、cookie プロバイダがクエリを行わないようにします。このパラメータを no に設定すると、Web エージェントによってすべての要求が cookie プロバイダに送られます。従来の(非フレームワーク)エージェントでは、このパラメータの値が Web エージェントのログ ファイルに表示されるように cookie プロバイダを設定する必要があります。

デフォルト: No

詳細情報:

保護されていないリソースにおける cookie プロバイダの無視 (P. 112)

IgnoreExt

Web エージェントが SiteMinder ポリシーを確認せずに Web サーバに要求を渡すリソースのタイプを指定します。SiteMinder ポリシーによって保護されるレルムにアイテムが存在する場合でも、Web エージェントはこのパラメータによって指定されたそのアイテムへのアクセスを許可します。

以下の条件のどちらかを満たすリソースに対する要求を無視することができます。

- リソースが、Web エージェントに対して無視するよう指定した拡張子で終わっている場合。
- 保護されているリソースを表す URI にピリオド(.)が 1 つだけ含まれている場合。

  たとえば、要求されたリソースの URI が /my.dir/ である場合、Web エージェントは要求を直接 Web サーバへ渡します。

デフォルト: .class、.gif、.jpg、.jpeg、.png、.fcc、.scc、.sfcc、.ccc、.ntc

重要: IgnoreExt パラメータを設定する場合は注意してください。セキュリティの問題には、検討が必要なものがいくつかあります。

詳細情報:

Web エージェントで無視する仮想サーバの指定 (P. 79)
HTTP ヘッダ リソースのキャッシュ方法の制御 (P. 148)
保護されていないリソースのファイル拡張子を無視することによるオーバーヘッドの削減 (P. 244)
複雑な URI の処理 (P. 248)

IgnoreHost

Web エージェントで無視するあらゆる仮想サーバの完全修飾ドメイン名を指定します。 そのような仮想サーバ上にあるリソースは自動許可され、どのクライアントが要求を行ったかにかかわらず、Web エージェントは常にそれらのリソースへのアクセスを許可します。許可は、ポリシーではなく Web エージェントの設定に基づいて決定されます。

IgnoreExt や IgnoreURL の各設定など、自動許可に関する他の項目より先に、無視されるホストに関する上記のリストが最初にチェックされます。したがって、無視されるホスト上にあるリソースに関しては、ダブルドットルールがポリシー サーバに対する許可の呼び出しをトリガすることはありません。しかし、拡張子に関するルールがそのようなリソースを無視することはありません。

IgnoreHost パラメータに対する URL エントリのホスト部分は、Web エージェントが読み取る、要求されたリソースのホスト ヘッダと完全に一致する必要があります。

注: この値では、大文字と小文字が区別されます。

URL で特定のポートを使用する場合、ポートを指定する必要があります。

一元管理されたエージェントでは、いくつかのサーバを表わすためにエージェント設定オブジェクトで複数値パラメータを使用します。ローカル設定ファイルで設定されたエージェントでは、ファイル内の個別の行に各ホストを列挙します。

例:（指定したポートと共に表示される URL）

IgnoreHost="myserver.example.org:8080"

例:（ローカル設定ファイル）

IgnoreHost="my.host.com"

IgnoreHost="your.host.com"

デフォルト: デフォルトなし

詳細情報:

Web エージェントで無視する仮想サーバの指定 (P. 79)

IgnoreQueryData

Web エージェントが URL 全体(クエリ文字列を含む)をキャッシュに保管し、ルール処理のために URI 全体をポリシー サーバに送信するかどうかを指定します。 完全な URL 文字列には、以下の例に示すように、URI、フック(?)、およびクエリ データが含まれます。

URI?*query_data*

デフォルトでは、リクエストの対象となった URL がキャッシュに保管されます。後続のリクエストでは、一致する URL がキャッシュで検索されます。リクエスト内で URI が同一でもクエリ データが異なると、一致は失敗します。クエリ データを無視すると、パフォーマンスが向上します。

IgnoreQueryData パラメータが yes の場合は、以下の処理が発生します。

- URL はフックの箇所で切り捨てられます。 URI だけがキャッシュされ、ポリシー サーバへ送信されます。クエリ データは、リダイレクトの適正な状態を維持するために、他の場所で維持されます。
- フック(?)の前にある部分のみがポリシー サーバに送信されて、ルールの処理が行われます。
- 以下の例に示す 2 つの URI は、同一のリソースとして処理されます。

  /myapp?data=1

  /myapp?data=2

IgnoreQueryData パラメータが no の場合は、以下の処理が発生します。

- その場合、URL 全体がキャッシュされます。
- URI 全体がポリシー サーバに送信されてルールの処理が行われます。
- 以下の例に示す URI は、異なるリソースとして処理されます。

  /myapp?data=1

  /myapp?data=2

デフォルト: No

重要: URL クエリ データに依存するポリシーがある場合は、この設定を有効にしないでください。

詳細情報:

URL 内のクエリ データの無視 (P. 234)

IgnoreUrl

保護されていない URL 内の URI を指定します。URI と関連付けられたリソースにアクセスしようとしたユーザは、認証を要求されません。Web エージェントは、スラッシュが 3 つ登場した後で、それ以降の URI 部分を無視します。たとえば、このパラメータを以下の値に設定したとします。

```
http://www.example.com/directory
```

Web エージェントは以下の URI を無視します。

```
directory
```

指定された URI が別のドメインにあっても、出現場所にかかわらず Web エージェントはそれを無視します。たとえば、Web エージェントは、以下の URL のすべてに事前に表示された URI を無視します。

```
http://www.example.com/directory
http://www.example.net/directory
http://www.example.org/directory
```

注: この値では、大文字と小文字が区別されます。

デフォルト: デフォルトなし

例: (ローカル設定ファイル内の複数の URI)

IgnoreUrl="http://www.example.com/directory"

IgnoreUrl="http://www.example.com/directory2"

例: (ドメインを指定せずに URI のみを使用)

IgnoreUrl="/*resource*/"

詳細情報:

URI への無制限のアクセスの許可 (P. 241)
Web エージェントで無視する仮想サーバの指定 (P. 79)

LegacyCookieProvider

フレームワーク エージェントが cookie プロバイダに POST 要求を送るかどうかを制御します。フレームワーク エージェントが cookie プロバイダとして動作する従来のエージェントに POST 要求を送ると、リダイレクトされた要求は代わりに GET になって失敗します。no に設定すると、フレームワーク エージェントは cookie プロバイダに POST 要求を送ります。yes に設定すると、フレームワーク エージェントは cookie プロバイダに POST 要求を送りません。

集中的なエージェント設定を使用している場合は、このパラメータをエージェント設定オブジェクトに追加する必要があります。このパラメータは、すでにローカル設定ファイル内に存在します。

デフォルト: なし(送信された POST 要求)

注: このパラメータはフレームワーク エージェントにのみ適用されます。

詳細情報:

POST 要求における cookie プロバイダの無視(フレームワーク エージェントのみ) (P. 113)

LegacyEncoding

Web エージェントで、レガシー URL 内のすべてのドル記号($)文字を強制的にハイフン(-)に置換します。これにより、MSR、パスワード サービス、および DMS に対する下位互換性も保証されます。このパラメータを no に設定すると、Web エージェントは文字列の SM を -SM- に変換します。このパラメータを yes に設定すると、Web エージェントはドル記号($)文字を変換しません。

デフォルト: (フレームワーク エージェント) No

デフォルト: (従来のエージェント) Yes

詳細情報:

レガシー URL エンコードの受け入れ (P. 326)
リダイレクト URL のクエリ文字列暗号化 (P. 236)

詳細情報:

フレームワーク エージェントと従来のエージェントの間の POST 維持の有効化 (P. 283)

LegacyTransferEncodingBehavior

Web エージェントが使用するメッセージ エンコーディングのタイプを指定します。このパラメータの値が no の場合、転送エンコーディング（transfer-encoding）がサポートされます。

このパラメータの値が yes の場合、コンテンツ エンコーディングがサポートされます。transfer-encoding ヘッダは無視され、content-length ヘッダのみがサポートされます。

デフォルト: No

- **重要**: このパラメータの値を yes に設定すると、Federation や、4 KB より長い POST データの維持といった機能が動作せず、大きな証明書が認識されない場合があります。

注: このパラメータは Apache Web エージェントにのみ適用されます。

詳細情報:

Apache Web エージェントでのレガシー アプリケーションの使用 (P. 69)

LegacyVariables

Web エージェントが HTTP ヘッダ名でアンダースコアを使用するかどうかを指定します。一部の Web サーバ（Sun Java System など）では、HTTP ヘッダでアンダースコア文字を使用すると、一部のアプリケーションで問題が発生します。

このパラメータを no に設定すると、以下の例に示されるように、HTTP ヘッダでアンダースコアは使用されません。

SM*HeaderName*

このパラメータを yes に設定すると、以下の例に示されるように、HTTP ヘッダでアンダースコアが使用されます。

SM_*HeaderName*

デフォルト:（従来のエージェント）Yes

デフォルト:（フレームワーク エージェント）No

詳細情報:

HTTP ヘッダのレガシー変数の有効化 (P. 154)

LoadPlugin

IIS 6.0 および Apache 2.0 Web エージェントのためにロードされるプラグインを指定します。プラグインはさまざまな種類のエージェント機能をサポートします。

デフォルト: デフォルトなし

重要: WebAgent.conf ファイルに他のパラメータを追加しません。

以下のプラグインを使用できます。

HttpPlugin

Web エージェントが HTTP エージェントとして動作するかどうかを指定します。

デフォルト: Enabled

SAMLAffiliatePlugin

Web エージェントと SAML アフィリエイト エージェントの間の通信を許可します(Federation セキュリティ サービスを購入している場合)。

デフォルト: Disabled

Affiliate10Plugin

Web エージェントと 4.x アフィリエイト エージェントの間の通信を許可します。これは SAML アフィリエイト エージェントでは使用されません。

デフォルト: Disabled

localconfigfile

LocalConfig.conf ファイルの場所を指定します。このファイルにはエージェント構成設定の大部分が含まれます。

デフォルト: デフォルトなし

LogAppend

既存のログ ファイルの最後に新しいログ情報を追加します。このパラメータを no に設定した場合、ロギングが有効になるたびに、ログ ファイル全体が上書きされます。

デフォルト: No

LogFile

Web エージェントがログを記録するかどうかを指定します。ローカル設定ファイル内でこのパラメータを yes に設定すると、エージェント設定オブジェクトの AllowLocalConfig パラメータを no に設定しても、ロギングは有効になります。

デフォルト: No

LogFileName

ログ ファイルの完全パス(ファイル名を含む)を指定します。

デフォルト: No

例: /export/iPlanet/servers/https-jsmith/logs/WebAgent.log

LogFileSize

ログ ファイルのサイズ制限(メガバイト単位)を指定します。現在のログ ファイルがこの制限に到達すると、新しいログ ファイルが作成されます。新しいログ ファイルは、以下の命名規則のうちの 1 つを使用します。

- フレームワーク エージェントでは、新しいログ ファイルの名前は、元の名前にシーケンス番号が追加されたものになります。たとえば、サイズ制限に到達すると、myfile.log という名前のログ ファイルは myfile.log.1 に名前が変更されます。
- 従来のエージェントでは、新しいログファイルの名前は、元の名前に日付とタイムスタンプが追加されたものになります。たとえば、サイズ制限に到達すると、myfile.log という名前のログ ファイルは myfile.log.09-18-2003-16-07-07 に名前が変更されます。

古いファイルは手動でアーカイブするか削除する必要があります。

デフォルト: 0(ロールオーバーなし)

例: 80

LogFilesToKeep

保持する Web エージェント ログ ファイルの数を指定します。以下の場合に新しいログ ファイルが作成されます。

- Web エージェントが起動したとき。
- ログ ファイルのサイズ制限（LogFileSize パラメータの値で指定）に達したとき。

このパラメータの値を変更しても、保持数を超える既存のログ ファイルは自動的に削除*されません*。たとえば、システムに 500 個のログ ファイルが格納されているときに、それらのファイルのうち 50 個のみを保持することを指定しても、Web エージェントは、残りの 450 個のファイルを削除*しません*。

このパラメータの値を 0 に設定すると、すべてのログ ファイルが保持されます。

デフォルト: 0

LogLocalTime

ログがグリニッジ標準時（GMT）とローカル時間のどちらを使用するかを指定します。GMT を使用するには、この設定を no に変更します。このパラメータが存在しない場合は、デフォルト設定が使用されます。

デフォルト: yes

LogOffUri

完全なログ オフを有効にし、正常にログ オフした後にユーザに表示される Web サーバ上のカスタム Web ページの場所を指定します。ブラウザ キャッシュ内に格納できないようにこのページを設定する必要があります。設定しなかった場合、ブラウザは、ユーザをログ オフせずに、そのキャッシュからログオフ ページを表示する場合があります。これは、不正なユーザにセッションの支配権を握る機会を与える場合があります。

注: CookiePath パラメータが設定されているときは、LogOffUri パラメータの値が同じ cookie パスを指している必要があります。たとえば、CookiePath パラメータの値が example.com に設定されている場合、LogOffUri は example.com/logoff.html を指している必要があります。

デフォルト: デフォルトなし

制限: 完全修飾 URL は*使用しないでください*。相対 URI を使用する必要があります。

例: /Web pages/logoff.html

LowerCaseHTTP

Web エージェントが大文字と小文字のどちらの HTTP ヘッダを使用するかを指定します。一部の Web サーバでは大文字と小文字が区別される場合があります。大文字ヘッダを指定するには、このパラメータを no に設定します。

デフォルト: yes

注: このパラメータは、IIS エージェントには*適用されません*。

LowerCaseProtocolSpecifier

リダイレクト URL のスキーム(プロトコル)部分で小文字のみを使用するかどうかを指定します。

デフォルト: No

例: http、https

注: このパラメータはフレームワーク エージェントにのみ適用されます。

詳細情報:

小文字の URL プロトコルの指定 (P. 287)

MasterCookiePath

cookie プロバイダによって作成されたプライマリドメイン セッション cookie のパスを指定します。たとえば、このパラメータが /siteminderagent に設定されている場合、cookie プロバイダが作成するすべてのセッション cookie のパスに /siteminderagent が含まれます。cookie プロバイダ エージェントにこのパラメータが設定されていない場合は、デフォルト値が使用されます。

デフォルト: /(ルート)

MaxResourceCacheSize

Web エージェントがそのリソース キャッシュ内で保持するエントリの最大数を指定します。エントリには以下の情報が含まれます。

- リソースが保護されるかどうかに関するポリシー サーバのレスポンス
- レスポンスで返される追加属性

最大値に達すると、新しいリソース レコードが最も古いリソース レコードと置き換わります。

これらをより大きな数値に設定する場合は、十分なシステム メモリがあることを確認してください。

OneView モニタを使用して Web エージェント統計を表示している場合は、ResourceCacheCount に表示される値が MaxResourceCacheSize パラメータで指定された値より大きいことがあります。これはエラーではありません。Web エージェントは、MaxResourceCacheSize パラメータを 1 つのガイドラインとして使用します。また、値は状況により異なります。これは、MaxResourceCacheSize パラメータはリソース キャッシュ内の平均サイズのエントリの最大数を示すためです。実際のキャッシュ エントリは、あらかじめ識別された平均サイズより大きかったり小さかったりする可能性があります。したがって、実際の最大エントリ数は指定された値より多い場合や少ない場合があります。

注: フレームワーク エージェントなど、共有メモリを使用する Web エージェントの場合、キャッシュは MaxResourceCacheSize の値に基づいて一定サイズが事前に割り当てられ、それより増えることはありません。

デフォルト:（Domino Web サーバ）1000

デフォルト:（IIS および Sun Java System Web サーバ）700

デフォルト:（Apache Web サーバ）750

注: このパラメータの値を変更した場合は、変更を適用するために Web サーバを再起動する必要があります。

MaxSessionCacheSize

エージェントがそのセッション キャッシュ内で保持するユーザの最大数を指定します。セッション キャッシュには、認証するユーザのセッション ID が正常に格納されます。それらのユーザが同じセッション中に同じレルム内の別のリソースにアクセスした場合、エージェントはポリシー サーバをコールする代わりにセッション キャッシュの情報を使用します。この最大数に達すると、エージェントは最も古いユーザ レコードを新しいユーザ レコードと置き換えます。

このパラメータの値は、持続期間にリソースにアクセスしてそれを使用する予定のユーザの数に基づいて設定します。これらをより大きな数値に設定する場合は、十分なシステム メモリがあることを確認してください。

デフォルト:（Domino Web サーバ）1000

デフォルト:（IIS および Oracle iPlanet Web サーバ）700

デフォルト:（Apache Web サーバ）750

注: このパラメータの値を変更した場合は、変更を適用するために Web サーバを再起動する必要があります。

MaxTimeoutURL

セッションの最大タイムアウトが発生したときに、Web エージェントがユーザをリダイレクトする先の URL を指定します。

例: http://example.mycompany.com/maxtimeoutpage.html

デフォルト: デフォルトなし

MaxUrlSize

Web エージェントが処理できる URL の最大サイズ（バイト単位）を指定します。Web サーバによって、URL の長さ制限は異なるため、このパラメータを設定する前に Web サーバ ベンダーのマニュアルを確認してください。

デフォルト: 4096 B

NTCExt

NTLM 認証情報コレクタと関連付けられた MIME タイプを指定します。NTC は、Windows 認証方式によって保護されているリソースに関連する NT 認証情報を収集します。この方式は、Internet Explorer ブラウザからアクセスされる IIS Web サーバ上のリソースに適用できます。

このパラメータに複数の拡張子を持たせることもできます。エージェント設定オブジェクトを使用している場合は、複数値オプションを選択します。ローカル設定ファイルを使用している場合は、各拡張子をカンマで区切ります。

デフォルト: .ntc

OverlookSessionForMethods

Web エージェントがこのパラメータ内に列挙されたメソッドに対してすべての HTTP リクエストのリクエスト メソッドを比較するかどうかを指定します。一致した場合、Web エージェントは SMSESSION cookie の作成も更新も行いません。さらに、cookie プロバイダ（設定されている場合）はそのリクエストに対して更新されません。

デフォルト: デフォルトなし

OverlookSessionForMethodUri

Web エージェントが、すべての HTTP リクエストの URI を、このパラメータに示されている URI と比較するかどうかを指定します。一致した場合、Web エージェントは SMSESSION cookie の作成も更新も行いません。さらに、cookie プロバイダ（設定されている場合）はそのリクエストに対して更新されません。

デフォルト: デフォルトなし

OverlookSessionForUrls

Web エージェントが、すべての HTTP リクエストの URLs を、このパラメータに示されている URLs と比較するかどうかを指定します。一致した場合、Web エージェントは SMSESSION cookie の作成も更新も行いません。さらに、cookie プロバイダ（設定されている場合）はそのリクエストに対して更新されません。

デフォルト: デフォルトなし

例: /MyDocuments/index.html のような相対 URL を使用します。絶対 URL（http://fqdn.host/MyDocuments/index.html）は使用しません。

OverrideIgnoreExtFilter

Web エージェントがすべての URI と比較する目的で使用するために、複数の文字列から成る 1 つのリストを指定します。これは、通常は拡張子が Web エージェントによって無視されるリソース、または拡張子のないファイルやアプリケーションの保護に役立ちます。URI がリスト内の文字列の 1 つと一致する場合、Web エージェントはポリシー サーバへの問い合わせを行い、そのリソースが保護されているかどうかを決定します。

パスを厳密に指定するのではなく一般的な文字列を指定することをお勧めします。一群のリソースを保護するために部分的な文字列を含めることもできます。たとえば、指定された文字列が /servlet/ の場合、以下のリソースが保護されます。

- /dira/app1/servlet/app
- /dirb/servlet/app1
- /dirc/mydir/servlet/app2

デフォルト: デフォルトなし

P3PCompactPolicy

カスタム レスポンスがプライバシ優先プロジェクト用のプラットフォーム（P3P）レスポンス ヘッダに準拠するかどうかを決定します。P3P コンパクトポリシーは、P3P の用語に基づく特定の要素を表すトークンを使用します。P3PCompactPolicy パラメータを適切なポリシー構文に設定した場合、Web エージェントに関して P3P レスポンス ヘッダが指定されている状況で、正しい P3P レスポンス ヘッダを使用して、カスタム レスポンスが設定されることを保証できます。

デフォルト: デフォルトなし

例: NON DSP COR CURa TAI（これらはそれぞれ、none、disputes、correct、current/always、および tailoring を表します）

注: このパラメータは Apache 1.3 または Domino の Web エージェントでは *サポートされていません*。

PersistentCookies

エージェントが複数のブラウザ セッションのシングル サインオンを許可するかどうかを指定します。これが有効な場合、1 つのブラウザ セッション中に認証するユーザは後続のブラウザ セッションに対してシングル サインオン機能を保持します。

autoauthorizeoptions パラメータの値を yes に設定した場合、PersistentCookies パラメータの値は no に設定します。

さらに、永続的な cookie を有効にするためには、TransientIDCookies パラメータを no に設定する必要があります。

このパラメータは以下のパラメータに影響します。

- TransientIDCookies
- PersistentIPCheck

デフォルト: No

詳細情報:

シングル サインオンの設定方法 (P. 93)
永続的 cookie の設定 (P. 95)
セキュリティ侵害を防止するための IP アドレスの比較 (P. 258)

PersistentIPCheck

最後の要求から受信した IP アドレス(永続的な cookie に保存されている)と現在の要求の IP アドレスとを比較して、その 2 つが一致するかどうかを確認するように Web エージェントに指示します。IP アドレスが一致しない場合、Web エージェントは要求を拒否します。

注: SiteMinder ID cookie は IP チェックの影響を受けません。

このパラメータは以下のパラメータに影響します。

- PersistentCookies

デフォルト: yes

PostPreservationFile

以下の POST 維持テンプレート ファイルの*いずれか*に対するパスを指定することで、トラディショナル エージェントとフレームワーク エージェントとの間の POST 維持データの転送を有効にします。

- tr2fw.pptemplate - トラディショナル エージェントが稼働しているサーバでホストされているリソースが、フレームワーク エージェント上で実行されている FCC によって保護されていることを示します。
- fw2tr.pptemplate - フレームワーク エージェントが稼働しているサーバでホストされているリソースが、トラディショナル エージェント上で実行されている FCC によって保護されていることを示します。

デフォルト: デフォルトなし

例: *web_agent_home*/samples/forms/fw2tr.pptemplate

PreserveHeaders

新しいヘッダが生成されたときに Web エージェントが既存の HTTP ヘッダを置き換えずに保存するかどうかを指定します。Sun Java System、Domino、および Apache Web エージェントではこのパラメータを yes を設定します。

デフォルト: No

PreservePostData

要求をリダイレクトする場合に Web エージェントが POST データを維持するかどうかを指定します。 ユーザが、フォーム認証や証明書認証など、高度な認証を受ける場合、POST データは認証フェーズの間、維持されます。

デフォルト: yes

ProxyAgent

Web エージェントがリバース プロキシ エージェントとして動作するかどうかを指定します。

このパラメータの値が yes である場合、フロントエンド サーバ上の SiteMinder Web エージェントは SM_PROXYREQUEST HTTP ヘッダ内のユーザによって要求された元の URL を維持します。保護されているリソースと保護されていないリソースが要求された場合は常に、このヘッダが作成されます。バックエンド サーバは、元の URL に関する情報を取得するためにこのヘッダを読み取ることができます。

デフォルト: No

注: このパラメータは Apache Web エージェントにのみ適用されます。

ProxyDefinition

リクエスタ IP アドレスを解決するためにカスタム HTTP ヘッダを使用する必要があるプロキシ（キャッシュ デバイスなど）の IP アドレスを指定します。

デフォルト: デフォルトなし

制限: 文字列には IP アドレスが含まれている必要があります。サーバ名または完全修飾 DNS ホスト名は*使用しないでください*。

ProxyHeadersAutoAuth

Web エージェント設定の ExpireForProxy パラメータが yes に設定されている場合、Web エージェントがクライアントへの HTTP レスポンスに挿入する HTTP 1.1 ヘッダの値を指定します。このヘッダの値によって、自動許可されたリソースがキャッシュされるかどうか、またはキャッシュされる期間が決定します。

デフォルト: Expires: Thu, 01 Dec 1994 16:00:00 GMT

例（推奨される設定）: "Cache-control: max-age=60"

注: このパラメータ名には 'proxy' という単語が含まれていますが、このパラメータの設定は、Web ブラウザ、またはこのパラメータ設定を使用する SiteMinder エージェントが動作する Web サーバに接続するすべてのクライアントの動作にも影響があります。

詳細情報:

Cache-Control ヘッダ設定と ExpireForProxy ヘッダ設定のカスタマイズ (P. 216)
プロキシ サーバの背後にあるエージェントの設定 (P. 213)

ProxyHeadersAutoAuth10

Web エージェント設定の ExpireForProxy パラメータが yes に設定されている場合、Web エージェントがクライアントへの HTTP レスポンスに挿入する HTTP 1.0 ヘッダの値を指定します。このヘッダの値によって、自動許可されたリソースがキャッシュされるかどうか、またはキャッシュされる期間が決定します。

デフォルト: Expires: Thu, 01 Dec 1994 16:00:00 GMT

例(推奨される設定): "Expires: Thu, 01 Dec 1994 16:00:00 GMT"

注: このパラメータ名には 'proxy' という単語が含まれていますが、このパラメータの設定は、Web ブラウザ、またはこのパラメータ設定を使用する SiteMinder エージェントが動作する Web サーバに接続するすべてのクライアントの動作にも影響があります。

詳細情報:

Cache-Control ヘッダ設定と ExpireForProxy ヘッダ設定のカスタマイズ (P. 216)
プロキシ サーバの背後にあるエージェントの設定 (P. 213)

ProxyHeadersProtected

Web エージェント設定の ExpireForProxy パラメータが yes に設定されている場合、Web エージェントがクライアントへの HTTP レスポンスに挿入する HTTP 1.1 ヘッダの値を指定します。このヘッダの値によって、保護されているリソースがキャッシュされるかどうか、またはキャッシュされる期間が決定します。

デフォルト: Expires: Thu, 01 Dec 1994 16:00:00 GMT

Cache-Control: no-cache

例(推奨される設定): "Cache-Control: private"

ProxyHeadersProtected="Cache-Control: max-age=60"

注: このパラメータ名には 'proxy' という単語が含まれていますが、このパラメータの設定は、Web ブラウザ、またはこのパラメータ設定を使用する SiteMinder エージェントが動作する Web サーバに接続するすべてのクライアントの動作にも影響があります。

詳細情報:

プロキシ サーバの背後にあるエージェントの設定 (P. 213)
Cache-Control ヘッダ設定と ExpireForProxy ヘッダ設定のカスタマイズ (P. 216)

ProxyHeadersProtected10

Web エージェント設定の ExpireForProxy パラメータが yes に設定されている場合、Web エージェントがクライアントへの HTTP レスポンスに挿入する HTTP 1.0 ヘッダの値を指定します。このヘッダの値によって、保護されているリソースがキャッシュされるかどうか、またはキャッシュされる期間が決定します。

デフォルト: Expires: Thu, 01 Dec 1994 16:00:00 GMT

Cache-Control: no-cache

例(推奨される設定): "Expires: Thu, 01 Dec 1994 16:00:00 GMT"

注: このパラメータ名には 'proxy' という単語が含まれていますが、このパラメータの設定は、Web ブラウザ、またはこのパラメータ設定を使用する SiteMinder エージェントが動作する Web サーバに接続するすべてのクライアントの動作にも影響があります。

詳細情報:

プロキシ サーバの背後にあるエージェントの設定 (P. 213)
Cache-Control ヘッダ設定と ExpireForProxy ヘッダ設定のカスタマイズ (P. 216)

ProxyHeadersUnprotected

Web エージェント設定の ExpireForProxy パラメータが yes に設定されている場合、Web エージェントがクライアントへの HTTP レスポンスに挿入する HTTP 1.1 ヘッダの値を指定します。このヘッダの値によって、保護されていないリソースがキャッシュされるかどうか、またはキャッシュされる期間が決定します。

デフォルト: Expires: Thu, 01 Dec 1994 16:00:00 GMT

Cache-Control: no-cache

例(推奨される設定): ProxyHeadersUnprotected="Cache-Control: private"

ProxyHeadersUnprotected="Cache-Control: max-age=60"

注: このパラメータ名には 'proxy' という単語が含まれていますが、このパラメータの設定は、Web ブラウザ、またはこのパラメータ設定を使用する SiteMinder エージェントが動作する Web サーバに接続するすべてのクライアントの動作にも影響があります。

詳細情報:

プロキシ サーバの背後にあるエージェントの設定 (P. 213)
Cache-Control ヘッダ設定と ExpireForProxy ヘッダ設定のカスタマイズ (P. 216)

ProxyHeadersUnprotected10

Web エージェント設定の ExpireForProxy パラメータが yes に設定されている場合、Web エージェントがクライアントへの HTTP レスポンスに挿入する HTTP 1.0 ヘッダの値を指定します。このヘッダの値によって、保護されていないリソースがキャッシュされるかどうか、またはキャッシュされる期間が決定します。

デフォルト: Expires: Thu, 01 Dec 1994 16:00:00 GMT

Cache-Control: no-cache

例(推奨される設定): "Expires: Thu, 01 Dec 1994 16:00:00 GMT"

注: このパラメータ名には 'proxy' という単語が含まれていますが、このパラメータの設定は、Web ブラウザ、またはこのパラメータ設定を使用する SiteMinder エージェントが動作する Web サーバに接続するすべてのクライアントの動作にも影響があります。

詳細情報:

プロキシ サーバの背後にあるエージェントの設定 (P. 213)
Cache-Control ヘッダ設定と ExpireForProxy ヘッダ設定のカスタマイズ (P. 216)

ProxyTimeout

要求に応答するためにリバース プロキシの背後に展開された Web エージェントをリバース プロキシが待機する秒数を指定します。

デフォルト: デフォルトなし

注: このパラメータは Apache Web エージェントにのみ適用されます。

詳細情報:

SiteMinder リバース プロキシ展開の考慮事項 (P. 224)

ProxyTrust

プロキシ サーバが実行した許可を信頼するよう、アクセス先サーバに対応する Web エージェントに指示します。アクセス先サーバに対応する Web エージェントがユーザを再度許可する必要がないため、これはより効率的です。

デフォルト: no

詳細情報:

プロキシ サーバの背後にあるエージェントの設定 (P. 213)
SiteMinder リバース プロキシ展開の考慮事項 (P. 224)

PSPollInterval

ポリシー変更に関する情報または動的に更新されたキーを取得するために Web エージェントがポリシー サーバと通信する間隔(秒単位)を指定します。数値が大きい(間隔が長い)ほど、ネットワークトラフィックは減少します。数値が小さい(間隔が短い)ほど、ネットワークトラフィックは増加します。

デフォルト: 30

制限: 1

詳細情報:

エージェントがポリシーまたはキーの更新をチェックする頻度の変更 (P. 60)
Web エージェントとダイナミック キーのロールオーバー (P. 23)

RemoteUserVar

Web エージェントに対し、HTTP-WebAgent-Header-Variable レスポンス属性の値に基づいて REMOTE_USER 変数の値を設定するように指示します。このパラメータは、レガシー アプリケーションと統合するために使用します。レスポンス変数の名前のみを入力します。

例: 「user=aperson」のような HTTP-WebAgent-Header-Variable を返すには、RemoteUserVar パラメータを「user」に設定します。

デフォルト: デフォルトなし

詳細情報:

REMOTE_USER 変数を設定するように Web エージェントを設定する (P. 134)

ReqCookieErrorFile

RequireCookies パラメータが yes に設定されているときに基本的な認証情報を備えた cookie がブラウザによって返されない場合に、ユーザがリダイレクトされるカスタマイズされたエラー ページを指定します。

例: http://yourcompany.com/need_cookies.htm

詳細情報:

カスタム エラー処理の指定 (P. 158)
エラー処理をセットアップする方法 (P. 160)

RequireCookies

SiteMinder が cookie を必要とするかどうかを指定します。SiteMinder では以下を実行するために cookie を使用します。

- シングル サインオン環境を保護する
- セッション タイムアウトを追跡する
- アイドル タイムアウトを追跡する

重要: cookie を要求するように Web エージェントを設定する場合、ユーザ側では、Web ブラウザに HTTP cookie を受け取るように設定することが必要です。ブラウザが cookie を受け取らない場合は、エージェントからエラーメッセージが返され、すべての保護されたリソースに対するユーザのアクセスが拒否されます。

デフォルト: yes

詳細情報:

シングル サインオンの設定方法 (P. 93)
基本認証用の cookie が必要です。 (P. 94)
カスタム エラー処理の指定 (P. 158)

ResourceCacheTimeout

リソース エントリがキャッシュに保存される秒数を指定します。時間間隔の値を超えると、Web エージェントはキャッシュされたエントリを削除します。その後、保護されているリソースにユーザがアクセスしようとすると、Web エージェントはポリシー サーバに問い合わせます。

デフォルト: 600(10 分)

注: このパラメータの値を変更した場合は、変更を適用するために Web サーバを再起動する必要があります。

詳細情報:

Web エージェント キャッシュ (P. 205)
リソース エントリをキャッシュに保存しておく時間の制御 (P. 206)
変更時にサーバの再起動を必要とするパラメータ (P. 25)

SaveCredsTimeout

ユーザ認証情報が含まれている永続的な cookie が保存される時間数を指定します。この時間中に、Web エージェントは、cookie 内に保存されたデータでユーザを認証します。この時間を過ぎると、cookie は削除され、Web エージェントは再度ユーザ認証を試みます。

デフォルト: 720(30 日)

詳細情報:

保存された認証情報のタイムアウトの設定 (P. 107)

SCCExt

SSL 認証情報コレクタの MIME タイプを指定します。

デフォルト: .scc

詳細情報:

IIS と Domino の各 Web サーバでのクレデンシャル コレクタのセットアップ (P. 266)

SecureApps

エージェントが、権限のないユーザからの URL を許可することを防ぎます。Web エージェントが、特定の拡張子で終わるファイルに対するリクエストを無視するように設定されている場合は、偽の URL を作成してリソースにアクセスしようとする攻撃を受ける可能性があります。

たとえば、以下の URL を持つリソースがあるとします。

```
/scripts/myapp
```

以下の例のような偽の URL を作成して、アクセス権を取得しようとする攻撃を受ける可能性があります。

```
/scripts/myapp/junk.jpg
```

SecureApps パラメータの値が no の場合に、Web エージェントが .jpg ファイルのリクエストを無視するように設定されていると、/scripts/myapp/junk.jpg のリクエストは自動的に許可されます。

SecureApps パラメータの値が yes の場合は、Web エージェントは、リソースが正当であるか、URL が偽であるかの検出を試みます。

デフォルト: No

詳細情報:

アプリケーションのセキュリティ保護 (P. 247)

SecureURLs

Web エージェントがリダイレクト URL 内の SiteMinder クエリ パラメータを暗号化するかどうかを指定します。この設定を使用して、高度な認証方式であるパスワード サービスによって保護されている要求されたリソースのセキュリティを強化したり、要求が cookie プロバイダを呼び出すときのセキュリティを強化したりすることができます。

重要: Web エージェントは、SiteMinder コンポーネント間で送信されたデータを暗号化するだけです。リダイレクトのために SiteMinder 以外のアプリケーションに送信されるデータは暗号化されません。

以下の SiteMinder 認証情報コレクタおよびアプリケーションは SecureUrls 機能をサポートします。

- HTML フォーム認証
- 証明書およびフォーム認証
- SSL 認証
- 証明書またはフォーム認証
- NTLM 認証
- ACE 認証
- SafeWord 認証
- ユーザによる自己登録
- cookie プロバイダによるマルチドメイン シングル サインオン
- FCC ベースのパスワード サービス(CGI または JSP ベースではない)

デフォルト: No

詳細情報:

シングル サインオンと併用する場合の SecureUrls の設定 (P. 118)
リダイレクト URL のクエリ文字列暗号化 (P. 236)
リダイレクト URL のクエリ文字列暗号化と FCC ベースのパスワード サービス (P. 238)
リダイレクト URL 内のクエリ文字列パラメータの暗号化 (P. 239)

ServerErrorFile

サーバ エラーに遭遇したユーザに対してカスタム エラー ページを表示するように Web エージェントに指示します。このパラメータのファイル パスまたは URL を指定します。

デフォルト: デフォルトなし

詳細情報:

カスタム エラー処理の指定 (P. 158)
エラー処理をセットアップする方法 (P. 160)
カスタム エラー ページではなくサーバ エラー 500 が表示される (P. 344)
カスタム エラー ページが表示されない (P. 356)

ServerPath

Web エージェントが Web サーバの複数インスタンスを使用するように設定されている場合に、各 Web サーバ インスタンスの一意のパスを指定します。ServerPath は、Web エージェントのキャッシュ、ロギング、および状態監視のリソースに関して、一意の識別情報を作成します。

デフォルト: 空白

例: 4 つの Web サーバ インスタンスがあり、それぞれが Web エージェントをロードする場合、各サーバの WebAgent.conf ファイルの ServerPath パラメータには一意の値を設定する必要があります。ServerPath パラメータは *server_instance_root*/logs など、Web サーバのログ ファイルが保管されるディレクトリに設定できます。

注: このパラメータは Apache と Sun Java System エージェントにのみ適用されます。

詳細情報:

フレームワーク エージェントの WebAgent.conf ファイル (P. 33)
複数の Web サーバ インスタンスを持つ Web エージェントの管理 (P. 62)
Windows システムに関する ServerPath パラメータの設定 (P. 63)
UNIX システムに関する ServerPath パラメータの設定 (P. 63)
ServerPath パラメータを必要とする追加設定 (P. 64)

SessionGracePeriod

SiteMinder セッション（SMSESSION） cookie が再生成されない秒数を指定します。以下の条件が すべて満たされる場合、cookie は再生成されません。

- URL SMSESSION cookie が存在しない。
- 現在の時刻と受け取った SMSESSION cookie の最終アクセス時刻の差が SessionGracePeriod 以下である。
- 現在の時刻と受け取った cookie がアイドルになった時刻の間隔が 2 つの猶予期間を超えている。たとえば猶予期間が 25 分でアイドル タイムアウトが 60 分である場合、セッションがアイドルになる前に残っている時間が 2 つの猶予期間（50 分）に満たないので、SiteMinder は 10 分後にセッション cookie を再生成します。

デフォルト: 30

詳細情報:

セッション猶予期間の変更 (P. 106)

SessionUpdatePeriod

新しい cookie を設定する目的で、Web エージェントが要求を cookie プロバイダにリダイレクトする頻度（秒単位）を指定します。マスタ cookie を更新すると、SiteMinder セッションのアイドル タイムアウトが原因でその cookie が期限切れになる確率を低下させることができます。

デフォルト: 60

詳細情報:

cookie プロバイダの指定 (P. 97)
セッション更新期間の変更 (P. 98)

SetRemoteUser

いくつかのレガシー アプリケーションが必要とする場合がある REMOTE_USER 変数の値を指定します。

デフォルト: No

詳細情報:

IIS Web エージェントで REMOTE_USER 変数を取り込む方法 (P. 133)
REMOTE_USER 変数を設定するように Web エージェントを設定する (P. 134)
IIS 6.0 サーバ ログでのユーザ名およびトランザクション ID の記録 (P. 181)

SFCCExt

SSL フォーム認証情報コレクタの MIME タイプを指定します。

デフォルト: .sfcc

詳細情報:

IIS と Domino の各 Web サーバでのクレデンシャル コレクタのセットアップ (P. 266)

SkipDominoAuth

Domino の認証メカニズムを使用する代わりにユーザを認証するように SiteMinder Web エージェントに指示します。ユーザが Domino ディレクトリに保存されていない *保存されていない* 場合にも、このパラメータを yes に設定する必要があります。

このパラメータは以下のパラメータに影響します。

- DominoSuperUser

デフォルト: yes

注: このパラメータは Domino Web エージェントのみ適用されます。

詳細情報:

Domino サーバによるユーザ認証 (P. 307)
SiteMinder によるユーザ認証 (P. 309)
Domino スーパー ユーザとしての認証 (P. 310)
実ユーザまたはデフォルト ユーザとしての認証 (P. 310)
Domino サーバに関するポリシーを作成する場合の考慮事項 (P. 323)

SSOTrustedZone

シングル サインオン セキュリティ ゾーンの信頼の信頼された SSOZoneNames の順序づけられた(大文字と小文字を区別する)リストを定義します。必要に応じて、SM を使用してデフォルト ゾーンを追加します。エージェントは常に、その他すべてのトラステッド シングル サインオン ゾーンより、自身の SSOZoneName を信頼します。

デフォルト: 空(提供されている場合は SM または SSOZoneName)

制限: 複数値

詳細情報:

信頼とフェールオーバの順序 (P. 340)

SSOZoneName

Web エージェントがサポートするシングル サインオン セキュリティー ゾーンの(大文字と小文字を区別する)名前を指定します。このパラメータの値は Web エージェントが作成する cookie の名前に先頭に付けられます。これは、それぞれの cookie ドメインと cookie を関連付けるのに役立ちます。このパラメータが空でない場合、SiteMinder は以下の規則を使用して、cookie を生成します。

ZonenameCookiename

デフォルト: 空(ゾーン名として SM を使用します。これは cookie に以下のデフォルトの名前を与えます)

- SMSESSION
- SMIDENTITY
- SMDATA
- SMTRYNO
- SMCHALLENGE
- SMONDENIEDREDIR

制限: 単一値

例: Z1 に値を設定すると以下の cookie が作成されます。

- Z1SESSION
- Z1IDENTITY
- Z1DATA
- Z1TRYNO
- Z1CHALLENGE
- Z1ONDENIEDREDIR

詳細情報:

トラステッド ゾーンの順序 (P. 331)
セキュリティ ゾーンの設定 (P. 336)
エージェントのシングル サインオン ゾーンの指定 (P. 338)

StoreSessioninServer

使い捨てのセッション cookie を使用するかどうか指定します。StoreSessioninServer パラメータの値が yes の場合は、使い捨てのセッション cookie が作成され、セッション サーバに格納されます。cookie プロバイダおよび Web エージェントは、セッション サーバの cookie にアクセスします。

cookie プロバイダおよび Web エージェントは、URL 内のセッション cookie を、セッション サーバ上に格納された使い捨てのセッション cookie に対応する GUID に置き換えます。

StoreSessioninServer パラメータの値が no の場合、セッション cookie は URL で直接渡されます。

デフォルト: No

SuppressServerHeader

IIS Web エージェントがレスポンスでサーバ HTTP ヘッダを返すことを防ぎます。このパラメータの値が no の場合、Web エージェントはレスポンスと一緒にサーバ ヘッダを送信し、IIS Web サーバはそれをクライアントに渡します。このパラメータの値が yes の場合、Web エージェントは、レスポンスでサーバ ヘッダを送信しません。

デフォルト: No

注: このパラメータは IIS Web エージェントにのみ適用されます。

詳細情報:

URLScan ユーティリティを使用する場合のサーバ HTTP ヘッダの削除 (P. 68)

TargetAsRelativeURI

要求を認証情報コレクタとターゲット リソースへ振り向ける際に、完全修飾 URL の代わりに相対 URI を使用するよう Web エージェントに指示します。相対 URI を使用すると、Web エージェントと共に他のシステム上に存在している認証情報コレクタがリクエストを処理することを防止できます。このパラメータを有効にすると、Web エージェントは、スラッシュ (/) 以外の文字で始まるすべてのターゲットを拒否します。

注: この設定項目は、cookie 認証情報コレクタ (CCC) を*除く*、他のすべての認証情報コレクタに適用されます。CCC は、このパラメータの完全修飾ドメイン名を使用する必要があります。相対 URI を使用した場合、OnAuthAccept レスポンスは CCC で適切に動作しません。

デフォルト: No

詳細情報:

認証情報コレクタのリダイレクトでの相対ターゲットの使用 (P. 277)

TraceAppend

ロギングが有効になるたびにファイル全体を書き直す代わりに、既存のログ ファイルの最後に新しいログ情報を追加します。

デフォルト: No

詳細情報:

トレース ロギングの設定 (P. 191)

TraceConfigFile

監視するコンポーネントとイベントを決定する WebAgentTrace.conf 設定ファイルの場所を指定します。

デフォルト: デフォルトなし

例: C:¥Program Files¥ca¥webagent¥config¥WebAgentTrace.conf

詳細情報:

トレース ロギングの設定 (P. 191)

TraceDelimiter

トレース ファイル内のフィールドを区切るカスタム文字を指定します。

デフォルト: デフォルトなし

例: |

詳細情報:

トレース ロギングの設定 (P. 191)

TraceFile

トレース ロギングを有効にします。

デフォルト: No

詳細情報:

トレース ロギングの設定 (P. 191)

TraceFileName

トレース ログ ファイルの完全パスを指定します。

デフォルト: デフォルトなし
例: C:¥Program Files¥ca¥webagent¥log¥trace.log

詳細情報:

トレース ロギングの設定 (P. 191)

TraceFileSize

トレース ファイルの最大サイズを指定します(メガバイト単位)。この制限に到達すると、Web エージェントは新しいファイルを作成します。

デフォルト: 0(新しいログ ファイルは作成されません)

例: 20(MB)

詳細情報:

トレース ロギングの設定 (P. 191)

TraceFilesToKeep

保持する Web エージェント トレース ログ ファイルの数を指定します。以下の場合に新しいトレース ログが作成されます。

- Web エージェントが起動したとき。
- トレース ログのサイズ制限(TraceFileSize パラメータの値で指定)に達したとき。

このパラメータの値を変更しても、保持数を超える既存のトレース ログは自動的に削除*されません*。たとえば、システムに 500 個のトレース ログが格納されているときに、それらのファイルのうち 50 個のみを保持することを指定しても、Web エージェントは、残りの 450 個のトレース ログを削除*しません*。

このパラメータの値を 0 に設定すると、すべてのトレース ログが保持されます。

デフォルト: 0

詳細情報:

保存されるトレース ログ ファイルの数の制限 (P. 202)

TraceFormat

トレース ファイルがメッセージを表示する方法を指定します。以下の*いずれかの*オプションを選択します。

- default - 角かっこ[]を使用してフィールドを囲みます。
- fixed - 固定幅のフィールドに使用します。
- delim - 選択した文字を使用してフィールドを区切ります。
- xml - XML-like タグを使用します。Web エージェントには、DTD (Document Type Definition、文書タイプ定義)や、他のスタイルシートは*付属していません*。

デフォルト: default(角かっこ)

詳細情報:

トレース ロギングの設定 (P. 191)

TrackSessionDomain

セッション cookie の中にセッション cookie の対象ドメインを暗号化して格納するように Web エージェントに指示します。後続のリクエストでセッション cookie が提示されると、Web エージェントは、セッション cookie 内にある対象ドメインを、要求されたリソースのドメインと比較します。ドメインが一致*しない* 場合、Web エージェントはリクエストを拒否します。

たとえば、このパラメータの値が yes に設定されているときに、operations.example.com での使用を目的とするセッション cookie が finance.example.com で提示された場合、Web エージェントはその cookie を拒否します。

デフォルト: no

詳細情報:

セッション cookie ドメインの検証 (P. 100)

TransientIDCookies

エージェント ID（SMIDENTITY）cookie が一時的か永続的かを指定します。複数のブラウザ セッションにわたってユーザにシングル サインオン機能を与えるためには永続的な cookie を使用します。SiteMinder セッションが期限切れになっていない限り、ユーザは再認証する必要がありません。

個別のブラウザ セッションごとにシングル サインオン環境に対して再認証する場合は、一時的な cookie を使用します。

このパラメータは以下のパラメータに影響します。

- TransientIPCheck
- PersistentIPCheck

デフォルト: No

詳細情報:

識別 cookie の制御 (P. 105)
永続的 cookie の設定 (P. 95)

TransientIPCheck

最後の要求から受信した IP アドレス(一時的な cookie に保存されている)と現在の要求の IP アドレスとを比較して、その 2 つが一致するかどうかを確認するように Web エージェントに指示します。IP アドレスが一致しない場合、Web エージェントは要求を拒否します。

注: SiteMinder ID cookie は IP チェックの影響を受けません。

このパラメータは以下のパラメータに影響します。

- TransientIDCookies

デフォルト: No

詳細情報:

セキュリティ侵害を防止するための IP アドレスの比較 (P. 258)

UseAnonAccess

プロキシ ユーザの認証情報を使用するのではなく、匿名ユーザとして Web アプリケーションを実行するように IIS Web エージェントに指示します。

デフォルト: No

注: このパラメータは IIS Web エージェントにのみ適用されます。

詳細情報:

IIS Web エージェントで REMOTE_USER 変数を取り込む方法 (P. 133)
匿名ユーザ アクセスの有効化 (P. 164)

UseHTTPOnlyCookies

Web エージェントが作成する cookie で HTTP のみの属性を設定するように Web エージェントに指示します。Web エージェントが、ユーザのブラウザにこの属性を持つ cookie を返すと、その cookie の内容はスクリプトで読み取ることができなくなります。これは、cookie を最初に設定した Web サイトのスクリプトにも当てはまります。これにより、cookie 内の機密情報を、権限のないサード パーティがスクリプトを使用して読み取ることを防ぐことができます。

デフォルト: No

詳細情報:

HTTP 専用属性を備えた cookie での情報の保護 (P. 258)

UseNetBIOSforIISAuth

IIS 6.0 Web エージェントが IIS ユーザ認証のために、ユーザ プリンシパル名（UPN）と NetBIOS 名のどちらを IIS 6.0 Web サーバに送信するかを指定します。

注: このパラメータは、Active Directory ユーザ ストアがポリシー サーバに関連付けられている場合のみ有効です。

このパラメータを有効にした場合は、SiteMinder の認証時にポリシー サーバが Active Directory からユーザ DN、UPN、および NetBIOS 名を抽出し、このデータを IIS 6.0 Web エージェントに送り返します。

管理 UI でユーザ ディレクトリに対して[Use Authenticated User's Security Context]オプションを選択したかどうか、および UseNetBIOSforIIAuth パラメータをどのように設定したかに応じて、ユーザのログオン認証情報は以下のように送信されます。

- UseNetBIOSforIISAuth パラメータが no に設定されている場合、IIS 6.0 Web エージェントは UPN 名を送信します。
- UseNetBIOSforIISAuth パラメータが yes に設定されている場合、Web エージェントは NetBIOS 名を送信します。

IIS Web サーバは、Web エージェントから受け取った認証情報を使用してユーザを認証します。

デフォルト: No

注: このパラメータは IIS Web エージェントにのみ適用されます。

詳細情報:

IIS 認証での NetBIOS 名または UPN の使用 (P. 165)

UseSecureCookies

安全な(HTTPS)接続を使用して、Web サーバに cookie を送信します。このパラメータを使用することで、ブラウザと Web サーバの間のセキュリティを向上させることができます。

この設定が有効な場合、シングル サインオン環境のユーザは、SSL Web サーバから非 SSL Web サーバに移動するときに再認証する必要があります。セキュア cookie は、従来の HTTP 接続を介して渡すことはできません。

デフォルト: No

詳細情報:

安全な cookie の設定 (P. 103)

UseSecureCPCookies

UseSecureCPCookies が Yes に設定されていると、cookie プロバイダは、セキュア cookie の使用も設定されている(つまり、UseSecureCookies も有効である)別の cookie ドメイン内の Web エージェントにのみ cookie を送信します。

この設定と UseSecureCookies が両方とも有効な場合、複数ドメインのシングル サインオン環境内のユーザは、SSL の Web サーバから別の cookie ドメインの非 SSL の Web サーバに移動するときに、再認証する必要があります。セキュア cookie は、従来の HTTP 接続を介して渡すことはできません。

デフォルト: No

詳細情報:

複数ドメインにわたる安全な cookie の設定 (P. 104)

UseServerRequestIp

仮想 Web サーバの物理 IP アドレスに従って AgentName を解決するように Web エージェントに指示します。Web サーバが仮想サーバ マッピングに IP アドレスを使用する場合は、このパラメータを使用してセキュリティを向上させます。このパラメータが no の場合、Web エージェントは、クライアントのリクエストの HTTP ホスト ヘッダ内のホスト名に従って AgentName を解決します。

Domino サーバでは、このパラメータは、Domino 6.x でのみサポートされています。他の Domino バージョン上のエージェントに対してこのパラメータを有効にすると、Web エージェントはデフォルトのエージェント名を使用します。

SSL 通信と仮想ホストを使用するように IIS Web エージェントが設定されている場合は、このパラメータを yes に設定する必要があります。IIS では、SSL を有効にした状態で、ホスト名を使用して仮想ホスト マッピングを行うことはできません。

デフォルト: No

ValidFedTargetDomain

（Federation のみ-SAML 2.0）Identity Provider Discovery を実装した場合に、フェデレーション環境の有効なドメインをすべてリスト表示します。

SiteMinder Identity Provider Discovery（IPD）サービスでリクエストを受信すると、リクエストの IPDTarget クエリ パラメータを調べます。このクエリ パラメータは、Discovery サービスでリクエストを処理した後にリダイレクトする URL をリスト表示します。IdP の場合、IPDTarget は SAML 2.0 シングル サインオン サービスです。SP の場合、ターゲットは共通ドメイン cookie を使用するリクエスト アプリケーションです。

フェデレーション Web サービスでは、IPDTarget URL のドメインを、ValidFedTargetDomain パラメータに指定されたドメインのリストと比較します。URL ドメインが ValidFedTargetDomain に設定されたドメインの 1 つと一致する場合、IPD サービスは IPDTarget パラメータに示された URL にユーザをリダイレクトします。このリダイレクトは SP の URL に対して行われます。

ドメインが一致しない場合、IPD サービスはユーザ リクエストを拒否し、ブラウザに 403 Forbidden が返されます。また、FWS トレース ログおよび affwebservices ログにエラーが報告されます。 これらのメッセージは、IPDTarget のドメインが有効なフェデレーション ターゲット ドメインとして定義されないことを示します。

ValidFedTargetDomain を設定しない場合、検証は行われず、ユーザはターゲット URL にリダイレクトされます。

制限: フェデレーション ネットワーク内の有効なドメイン

デフォルト: デフォルトなし

詳細情報:

有効なフェデレーション ターゲット ドメインの定義 (P. 279)

ValidTargetDomain

クレデンシャル コレクタがユーザをリダイレクトすることを許可されるドメインを指定します。URL 内のドメインがこのパラメータ内で設定されたドメインに一致しない場合は、リダイレクトが拒否されます。

デフォルト: デフォルトなし

詳細情報:

有効なターゲットドメインの定義 (P. 278)

Apache サーバにのみ使用されるエージェント設定パラメータ

以下に、Apache Web エージェントにのみ使用される設定パラメータをアルファベット順に示します。

DeleteCerts

Web エージェントで使用されなくなったときに Stronghold サーバに格納されている証明書を削除するかどうかを指定します。

デフォルト: No

GetPortFromHeaders

Web サーバ サービス構造からポート番号を取得する代わりに、HTTP HOST リクエスト ヘッダからポート番号を取得するように Web エージェントに指示します。

デフォルト: No

HttpsPorts

ユーザが Web サーバへの SSL 接続を使用しているかどうかを Web エージェントがリスンする安全なポートを指定します。このパラメータの値を指定する場合、安全な要求を提供するすべての Web サーバの対象となるすべてのポートを含める必要があります。値を指定しなかった場合、Web エージェントは HTTP スキームをサーバのコンテキストから読み取ります。

サーバが、(HTTPS を HTTP へ変換する) HTTPS アクセラレータの背後にある場合、ブラウザはその要求を SSL 接続として扱います。

デフォルト: 空白

例: 80

例: (複数のポート) 80,8080,8083

LegacyTransferEncodingBehavior

Web エージェントが使用するメッセージ エンコーディングのタイプを指定します。このパラメータの値が no の場合、転送エンコーディング（transfer-encoding）がサポートされます。

このパラメータの値が yes の場合、コンテンツ エンコーディングがサポートされます。transfer-encoding ヘッダは無視され、content-length ヘッダのみがサポートされます。

デフォルト: No

ProxyAgent

Web エージェントがリバース プロキシ エージェントとして動作するかどうかを指定します。

このパラメータの値が yes である場合、フロントエンド サーバ上の SiteMinder Web エージェントは SM_PROXYREQUEST HTTP ヘッダ内のユーザによって要求された元の URL を維持します。保護されているリソースと保護されていないリソースが要求された場合は常に、このヘッダが作成されます。バックエンド サーバは、元の URL に関する情報を取得するためにこのヘッダを読み取ることができます。

デフォルト: No

ProxyTimeout

要求に応答するためにリバース プロキシの背後に展開された Web エージェントをリバース プロキシが待機する秒数を指定します。

デフォルト: デフォルトなし

ServerPath

Web エージェントが Web サーバの複数インスタンスを使用するように設定されている場合に、各 Web サーバ インスタンスの一意のパスを指定します。ServerPath は、Web エージェントのキャッシュ、ロギング、および状態監視のリソースに関して、一意の識別情報を作成します。

デフォルト: 空白

例: 4 つの Web サーバ インスタンスがあり、それぞれが Web エージェントをロードする場合、各サーバの WebAgent.conf ファイルの ServerPath パラメータには一意の値を設定する必要があります。ServerPath パラメータは *server_instance_root*/logs など、Web サーバのログ ファイルが保管されるディレクトリに設定できます。

Domino サーバでのみ使用されるエージェント設定パラメータ

以下のリストは、Domino Web エージェントでのみ使用される設定パラメータをアルファベット順で示したものです。

DominoDefaultUser

Domino Web エージェントが SiteMinder が事前に Domino サーバへの別のディレクトリに対して認証したユーザを識別する名前を指定します。

重要: このパラメータがローカル設定ファイルに格納される場合は、暗号化する必要があります。このパラメータを暗号化するには、encryptkey ツールを使用します。パラメータを変更する場合は、ローカル設定ファイルを直接編集すしないでください。

デフォルト: デフォルトなし

DominoLegacyDocumentSupport

Web エージェントが Domino 環境内の保護されている Lotus Notes ドキュメントに対するユーザ要求を処理する方法を指定します。このパラメータを yes に設定すると、要求されたドキュメントに対してのみ、ユーザに ReadForm 許可が与えられます。

デフォルト: No

DominoLookUpHeaderForLogin

ユーザがリソースへのアクセスを要求した場合に、Domino Web エージェントはそのユーザが Domino ユーザ ディレクトリ内で一意であるかあいまいであるかを、Domino Web サーバに問い合わせます。これは、リソースへのアクセスを要求するユーザの名前がユーザ ディレクトリ内の他のユーザと同じである場合に役立ちます。

デフォルト: No

DominoMapUrlForRedirect

フォーム認証情報コレクタ(FCC)にリダイレクトするために、URL を Domino サーバの表現から URL フレンドリ名にマップ(正規化)するように Web エージェントに指示します。FCC は、要求された Domino リソースに対する要求を処理することができます。このパラメータがない場合は、デフォルトの動作が発生します。このパラメータが no の場合、Web エージェントは URL をマップせず、元の Domino サーバの表現を使用して FCC リダイレクトを実行します。

また、DominoNormalizeUrls パラメータも yes に設定する必要があります。そうしないと、URL は正規化されません。

デフォルト: yes

DominoNormalizeUrls

SiteMinder Web エージェントが、フォーム認証情報コレクタにリダイレクトする前に、Domino の URL を URL フレンドリ名に変換するかどうかを指定します。

Domino の URL を変換するためには、MapUrlsForRedirect パラメータも yes に設定する必要があります。

DominoNormalizeUrls パラメータが no の場合は、MapUrlsForRedirect パラメータが yes に設定されていても、URL は正規化されません。

重要: DominoNormalizeUrls パラメータを no に設定した場合、Notes データベース内の個々のドキュメントを保護することはできません。Domino Web サーバのデータベース全体またはサブディレクトリのみを保護することができます。

デフォルト: yes

DominoSuperUser

Domino サーバ上のすべてのリソースへのアクセス権があるユーザを識別し、SiteMinder に正常にログインしたすべてのユーザが Domino SuperUser として Domino にログインすることを確認します。

この値は暗号化できます。

このパラメータは以下のパラメータに影響します。

- SkipDominoAuth

デフォルト: デフォルトなし

DominoUseHeaderForLogin

SiteMinder のヘッダの値を Domino Web サーバに渡すよう Domino Web エージェントに指示します。Domino サーバはそのヘッダのデータを使用して、自らのユーザ ディレクトリの中に存在しているユーザを識別します。

デフォルト: デフォルトなし

SkipDominoAuth

Domino の認証メカニズムを使用する代わりにユーザを認証するように SiteMinder Web エージェントに指示します。ユーザが Domino ディレクトリに保存されていない *保存されていない* 場合にも、このパラメータを yes に設定する必要があります。

このパラメータは以下のパラメータに影響します。

- DominoSuperUser

デフォルト: yes

IIS サーバでのみ使用されるエージェント設定パラメータ

以下のリストは、IIS Web エージェントでのみ使用される設定パラメータをアルファベット順で示したものです。

AppendIISServerLog

認証されたユーザ名と IIS サーバ ログに対する SiteMinder トランザクション ID を個別の行に追加するように Web エージェントに指示します。

デフォルト: No

注: このパラメータは IIS 6.0 Web エージェントにのみ適用されます。

DefaultPassword

プロキシ ユーザとして IIS リソースにアクセスするために使用される関連する Windows ユーザのためのデフォルトのパスワードを指定します。

重要: このパラメータを暗号化する場合は、エージェント設定オブジェクトの中心に設定します。このパラメータがローカル設定ファイルに設定されると、それは暗号化されず、安全性が低下します。

デフォルト: デフォルトなし

DefaultUserName

プロキシ ユーザとして IIS リソースにアクセスするために使用される Windows ユーザの名前を指定します。SiteMinder で保護されている IIS Web サーバのリソースにアクセスするユーザが、必要なサーバ アクセス権限を持つユーザではないことがあります。たとえば、UNIX システム上の LDAP ユーザ ディレクトリに格納されているユーザは、IIS Web サーバの Windows システムへのアクセス権限がない場合があります。

Web エージェントが SiteMinder によってアクセスを許可されたユーザのプロキシユーザ アカウントとして機能するためには、NT 管理者によって割り当てられたこの NT ユーザ アカウントを使用する必要があります。

デフォルト: デフォルトなし

ForceIISProxyUser

Web エージェントが IIS プロキシ アカウントを使用して、IIS Web サーバ上の要求されたリソースへのアクセスを、IIS Web サーバにアクセスするのに十分な権限が通常は不足しているユーザに許可するかどうかを指定します。

このパラメータは以下のパラメータに影響します。

- DefaultUserName
- DefaultPassword

デフォルト: No

SuppressServerHeader

IIS Web エージェントがレスポンスでサーバ HTTP ヘッダを返すことを防ぎます。このパラメータの値が no の場合、Web エージェントはレスポンスと一緒にサーバ ヘッダを送信し、IIS Web サーバはそれをクライアントに渡します。このパラメータの値が yes の場合、Web エージェントは、レスポンスでサーバ ヘッダを送信しません。

デフォルト: No

UseAnonAccess

プロキシ ユーザの認証情報を使用するのではなく、匿名ユーザとして Web アプリケーションを実行するように IIS Web エージェントに指示します。

デフォルト: No

UseNetBIOSforIISAuth

IIS 6.0 Web エージェントが IIS ユーザ認証のために、ユーザ プリンシパル名（UPN）と NetBIOS 名のどちらを IIS 6.0 Web サーバに送信するかを指定します。

注: このパラメータは、Active Directory ユーザ ストアがポリシー サーバに関連付けられている場合のみ有効です。

このパラメータを有効にした場合は、SiteMinder の認証時にポリシー サーバが Active Directory からユーザ DN、UPN、および NetBIOS 名を抽出し、このデータを IIS 6.0 Web エージェントに送り返します。

管理 UI でユーザ ディレクトリに対して[Use Authenticated User's Security Context]オプションを選択したかどうか、および UseNetBIOSforIIAuth パラメータをどのように設定したかに応じて、ユーザのログオン認証情報は以下のように送信されます。

- UseNetBIOSforIISAuth パラメータが no に設定されている場合、IIS 6.0 Web エージェントは UPN 名を送信します。
- UseNetBIOSforIISAuth パラメータが yes に設定されている場合、Web エージェントは NetBIOS 名を送信します。

IIS Web サーバは、Web エージェントから受け取った認証情報を使用してユーザを認証します。

デフォルト: No

Oracle iPlanet Web サーバのみで使用されるエージェント設定パラメータ

以下に、Oracle iPlanet Web エージェントにのみ使用される設定パラメータをアルファベット順に示します。

DisableDirectoryList

最初に認証情報を要求せずに、ユーザがディレクトリの内容を表示または参照することを Web エージェントが認めるかどうかを指定します。これは、以下の条件がすべて当てはまる場合に発生します。

- レルムがルートリソース(/)を保護するように設定されている。
- ディレクトリのデフォルト Web ページ(index.html など)が名前変更または削除されている。

デフォルト: No

EnableOtherAuthTrans

SiteMinder で他の AuthTrans 機能の使用を許可します。

デフォルト: No

ServerPath

Web エージェントが Web サーバの複数インスタンスを使用するように設定されている場合に、各 Web サーバ インスタンスの一意のパスを指定します。ServerPath は、Web エージェントのキャッシュ、ロギング、および状態監視のリソースに関して、一意の識別情報を作成します。

デフォルト: 空白

例: 4 つの Web サーバ インスタンスがあり、それぞれが Web エージェントをロードする場合、各サーバの WebAgent.conf ファイルの ServerPath パラメータには一意の値を設定する必要があります。ServerPath パラメータは *server_instance_root*/logs など、Web サーバのログ ファイルが保管されるディレクトリに設定できます。

付録 C: エラー コード

Web エージェントは問題が発生したときにエラーコードを生成します。これにより、SiteMinder 操作における問題の診断が可能になります。たとえば、Web エージェントが SiteMinder 認証サーバに接続できない場合、エラー コード 20-0002 が表示されます。

Web エージェント エラー コードは、以下の方法で表示されます。

- Web ブラウザ
- カスタム サーバ エラー ページ

カスタム エラー処理を設定すると、Web エージェントは、指定の URL または（カスタム エラー ページを表示するように）カスタマイズした HTML ファイルに対して、エラー コードを渡します。

このセクションには、以下のトピックが含まれています。

さまざまな HTTP 500 サーバ エラー コード

このセクションでは、その他のサーバ エラー コードをリストします。

00-0001

原因:

IP アドレスからエージェント名を特定できない

処置:

エージェント設定を参照し、Web サーバが提供する各 HOST アドレスに対応して AgentName がマップされていること、または DefaultAgentName が正しく設定されていることを確認します。

00-0002

原因:

問題のある文字が URL 内に存在するか、BadUrlChars パラメータ内で定義された文字が URL 内で検出されました。

処置:

以下のいずれかの操作を行います。

- 問題のある文字を URL から削除します
- その文字を BadUrlChars パラメータのリストから削除します。その結果、その URL はブロックされなくなります。

00-0004

原因:

SSLCRED cookie がエラーのステータスを示している。

処置:

SCC（安全な認証情報コレクタ）として動作している Web エージェントを調査し、その設定を確認します。

通常、このエラーが発生するのは、SCC エージェントが自らの環境から認証情報を取得できない場合のみです。これは設定エラーの可能性を示しています。

00-0005

原因:

FORMCRED cookie がエラーのステータスを示している。

処置:

FCC（フォーム認証情報コレクタ）として動作している Web エージェントを調査し、その設定を確認します。

通常、このエラーが発生するのは、FCC エージェントが自らの環境から認証情報を取得できない場合のみです。これは設定エラーの可能性を示しています。

00-0006

原因:

NTLM 保護されたリソースが、期待されているリソース キャッシュの中で見つからなかった。

処置:

Windows 認証方式のセットアップを調査し、設定を確認します。

00-0007

原因:

ASCII エンコード エラーが存在する。これは Web エージェントの内部エラーです。

処置:

Web サーバと Web エージェントを調査し、不安定であることが疑われるサービスを診断します。

Web エージェントのログ ファイルと設定ファイルを参照できるように用意して、カスタマ サポートに問い合わせます。

00-0008

原因:

SSL 認証が失敗した。このエラーは、不適切な証明書が存在すること、またはそのユーザが認証されていないことを示します。

処置:

他の証明書を使用するか、SSL 認証方式の設定を調査し、疑わしい原因を探します。

00-0009

原因:

SSL 認証情報が不適切または見つからない。

処置:

他の証明書、またはユーザ名とパスワードのペアを使用します。SSL 認証方式の設定を調査し、疑わしい原因を探します。

00-0010

原因:

アクセスが拒否されました。このエラーは、アクセスがブロックされたことに起因する、一般的な障害を示しています。

処置:

Web エージェントとポリシー サーバのログを調査し、障害の根本的な原因を判断します。

00-0011

原因:

認証情報コレクタのエラー。このエラーは、アクセスがブロックされたことに起因する、フォームベースまたは SSL ベースの高度な認証に関する一般的な障害を示しています。

処置:

以下の手順を実行します。

- Web エージェントとポリシー サーバのログを確認し、障害の根本的な原因を判断します。
- 高度な認証方式のセットアップを調査し、原因を調べます。

00-0012

原因:

暗号化エラー。これは Web エージェントの内部エラーを示唆しています。

処置:

以下の手順を実行します。

- Web サーバと Web エージェントを調査し、不安定であることが疑われるサービスを診断します。
- キー ストアのセットアップを参照し、適切なエージェント キーが使用されていることを確認します。
- カスタマ サポートに問い合わせ、Web エージェントのログ ファイルと設定ファイルを参照用に送ります。

00-0013

原因:

エージェント設定エラー。起動時に 1 つ以上のエラーが発生し、Web エージェントの有効な設定を行うことができませんでした。

処置:

以下の手順を実行します。

- Windows 環境では、[アプリケーション]イベント ログを参照し、詳細を把握します。
- Apache エージェントでは、Apache エラー ログを参照して、詳細を把握します。
- Oracle iPlanet UNIX エージェントでは、シェルのプロンプトから Oracle iPlanet を起動し、STDERR によって表示される中から、疑わしいエラーを探します。
- SmHost.conf ファイルが存在している(ホストが正しく登録された)こと、および正しいエントリが記録されていることを確認します。
- エージェント設定ファイル内に、有効な SmHost.conf ファイルを指す、1 つの有効な HostConfigFile エントリが存在していることを確認します。
- AgentConfigObject が、1 つの有効な値を保持していることを確認します。

00-0014

原因:

ユーザをログ アウトできなかった。

処置:

詳細については以下のファイルを確認してください。

- Web エージェント ログ ファイル
- Web エージェント トレース ファイル
- ポリシー サーバ ログ ファイル
- ポリシー サーバ トレース ファイル

00-0015

原因:

SiteMinder アカウンティング サービスは監査要求に SM_AGENTAPI_NO で応答しました。

処置:

詳細については以下のファイルを確認してください。

- ポリシー サーバ ログ ファイル
- ポリシー サーバ トレース ファイル

00-0016

原因:

FQ ホスト名を解決できない。

処置:

Web エージェントのログを確認して、エージェントが解決しようとしているホスト名を特定します。ホスト名が正しい場合は、エージェントが実行される Web サーバの DNS 設定を確認します。

00-0017

原因:

無効なリダイレクト ターゲットが見つかった。

処置:

このメッセージをレポートしている Web エージェントのログ ファイルを調べて、処理されている URL（通常は FCC または他の高度な認証 URL）を特定し、TARGET CGI パラメータの値が有効であると考えられるかどうかを判断します。

HTTP ヘッダ解析エラー コード

このセクションでは、HTTP ヘッダの解析に関連するエラー コードをリストします。

10-0001

原因:

'SERVER_NAME' HTTP 変数を読み込むことができない。

処置:

Web ブラウザおよび Web サーバが HTTP 1.0 に準拠していることを確認します。

10-0002

原因:

'URL' HTTP 変数を読み込むことができない。

処置:

Web ブラウザおよび Web サーバが HTTP 1.0 に準拠していることを確認します。

10-0003

原因:

'method' HTTP 変数を読みこむことができない。

処置:

Web ブラウザおよび Web サーバが HTTP 1.0 に準拠していることを確認します。

10-0004

原因:

'host' HTTP 変数を読み込むことができない。

処置:

Web ブラウザおよび Web サーバが HTTP 1.0 に準拠していることを確認します。

10-0005

原因:

'URI' HTTP 変数を読み込むことができない。

処置:

Web ブラウザおよび Web サーバが HTTP 1.0 に準拠していることを確認します。

10-0007

原因:

URL が長すぎる。

処置:

MaxUrlSize パラメータの設定を大きくします。デフォルトの設定値は 4,096 バイトです。

SiteMinder 通信エラーコード

このセクションでは、通信エラーに関連するエラー コードをリストします。

20-0001

原因:

SiteMinder アカウンティング サーバに接続できないか、ポリシー サーバの予期できないエラーが発生した。

処置:

以下の手順を実行します。

- ポリシー サーバのログを参照し、エラーの詳細を調べます。
- ポリシー サーバに ping を行い、Web エージェントとポリシー サーバの接続状態を確認します。エージェントとポリシー サーバ間にファイアウォールが構築されている場合、以下のサービス ポートがブロックされていないことを確認します。
  - 44441(アカウンティング)
  - 44442(認証)
  - 44443(許可)

20-0002

原因:

SiteMinder 認証サーバに接続できないか、ポリシー サーバの予期できないエラーが発生した。

処置:

以下の手順を実行します。

- ポリシー サーバのログを参照し、エラーの詳細を調べます。
- ポリシー サーバに ping を行い、Web エージェントとポリシー サーバの接続状態を確認します。エージェントとポリシー サーバ間にファイアウォールが構築されている場合、以下のサービス ポートがブロックされていないことを確認します。
  - 44441(アカウンティング)
  - 44442(認証)
  - 44443(許可)

20-0003

原因:

SiteMinder 許可サーバに接続できないか、ポリシー サーバの予期できないエラーが発生した。

処置:

以下の手順を実行します。

- ポリシー サーバのログを参照し、エラーの詳細を調べます。
- ポリシー サーバに ping を行い、Web エージェントとポリシー サーバの接続状態を確認します。エージェントとポリシー サーバ間にファイアウォールが構築されている場合、以下のサービス ポートがブロックされていないことを確認します。
  - 44441(アカウンティング)
  - 44442(認証)
  - 44443(許可)

SiteMinder パスワード サービス エラーコード

このセクションでは、パスワード サービスに関連するエラー コードをリストします。

30-0026

原因:

パスワード サービスのリダイレクト URL が使用できない。

処置:

パスワード サービスのリダイレクト URL が設定されていることを確認します。

付録 D: 暗号化キーのロールオーバー メッセージ

このセクションには、以下のトピックが含まれています。

[情報]管理者:キー更新属性 'KEY_UPDATE_LAST' の受信

原因:

Web エージェントで、ポリシー サーバから LAST キー属性を更新するコマンドを受信しました。

処置:

なし

[情報]管理者:キー更新属性 'KEY_UPDATE_CURRENT' の受信

原因:

Web エージェントで、ポリシー サーバから CURRENT キー属性を更新するコマンドを受信しました。

処置:

なし

[情報]管理者:キー更新属性 'KEY_UPDATE_NEXT' の受信

原因:

Web エージェントで、ポリシー サーバから NEXT キー属性を更新するコマンドを受信しました。

処置:

なし

[情報]管理者:キー更新属性 'KEY_UPDATE_PERSISTENT' の受信

原因:

Web エージェントで、ポリシー サーバから PERSISTENT キー属性を更新するコマンドを受信しました。

処置:

なし

[情報]管理者: キー更新属性 'LAST' の処理完了

原因:

Web エージェントで指定されたキーがロールオーバされました。

処置:

なし

[情報]管理者: キー更新属性 'CURRENT' の処理完了

原因:

Web エージェントで指定されたキーがロールオーバされました。

処置:

なし

[情報]管理者: キー更新属性 'NEXT' の処理完了

原因:

Web エージェントで指定されたキーがロールオーバされました。

処置:

なし

[情報]管理者: キー更新属性 'PERSISTENT' の処理完了

原因:

Web エージェントで指定されたキーがロールオーバされました。

処置:

なし

管理者: キー更新属性 'LAST' を処理できない

原因:

Web エージェントで指定されたキーをロールオーバできませんでした。

処置:

なし

管理者: キー更新属性 'CURRENT' を処理できない

原因:

Web エージェントで指定されたキーをロールオーバできませんでした。

処置:

なし

管理者: キー更新属性 'NEXT' を処理できない

原因:

Web エージェントで指定されたキーをロールオーバできませんでした。

処置:

なし

管理者: キー更新属性 'PERSISTENT' を処理できない

原因:

Web エージェントで指定されたキーをロールオーバできませんでした。

処置:

なし

索引

D

E

F

H

I

L

N

O

P

R

S

T

U

W

あ

か

さ

た

な

は

ま

や

ら