記事
Toshihiko Minamoto · 2021年7月19日 5m read

InterSystems API Managementで OAuth 2.0 を使って API を保護する - パート 1

はじめに

近年、オープン認証フレームワーク（OAuth）を使って、あらゆる種類のサービスから信頼性のある方法で安全かつ効率的にリソースにアクセスするアプリケーションが増えています。 InterSystems IRIS はすでに OAuth 2.0 フレームワークに対応しており、事実コミュニティには、OAuth 2.0 と InterSystems IRIS に関する素晴らしい記事が掲載されています。

しかし、API 管理ツールの出現により、一部の組織はそのツールを単一の認証ポイントとして使用し、不正なリクエストが下流のサービスに到達するのを防ぎ、サービスそのものから承認/認証の複雑さを取り除いています。

ご存知かもしれませんが、InterSystems は、IRIS Enterprise ライセンス（IRIS Community Edition ではありません）で利用できる InterSystems API Management（IAM）という API 管理ツールを公開しています。 こちらには、InterSystems API Management を紹介する素晴らしい別のコミュニティ記事が掲載されています。

これは、IAM を使って、以前に IRIS にデプロイされた認証されていないサービスに OAuth 2.0 標準に従ったセキュリティを追加する方法を説明した 3 部構成記事の最初の記事です。

サービスを保護するプロセス全体を理解しやすくするために、最初の記事では、IRIS と IAM の基本的な定義と構成を示しながら OAuth 2.0 の背景を説明します。

この連載記事のパート 2 以降では、IAM によってサービスを保護する上で考えられる 2 つのシナリオを説明します。 最初のシナリオでは、IAM は着信リクエストに存在するアクセストークンを検証し、検証が成功した場合にのみリクエストを 転送します。 2 番目のシナリオでは、IAM がアクセストークンを生成し（承認サーバーとして機能します）、それを検証します。

従って、パート 2 では、シナリオ 1 を構成するために必要な手順を詳しく説明し、パート 3 ではシナリオ 2 の構成を説明した上で、最終的な考慮事項を示します。

IAM をお試しになりたい方は、InterSystems 営業担当者にお問い合わせください。

OAuth 2.0 の背景

すべての OAuth 2.0 承認フローには基本的に以下の 4 つのグループが関わっています。

1. ユーザー
2. クライアント
3. 承認サーバー
4. リソース所有者

分かりやすくするために、この記事では「リソース所有者パスワード資格情報」OAuth フローを使用しますが、IAM ではあらゆる OAuth フローを使用できます。

また、この記事では範囲を指定しません。
注意: クライアントアプリはユーザー資格情報を直接処理するため、クライアントアプリの信頼性が非常に高い場合にのみリソース所有者パスワード資格情報フローを使用することをお勧めします。
ほとんどの場合、クライアントはファーストパーティアプリである必要があります。

以下は、一般的なリソース所有者パスワード資格情報フローの手順です。

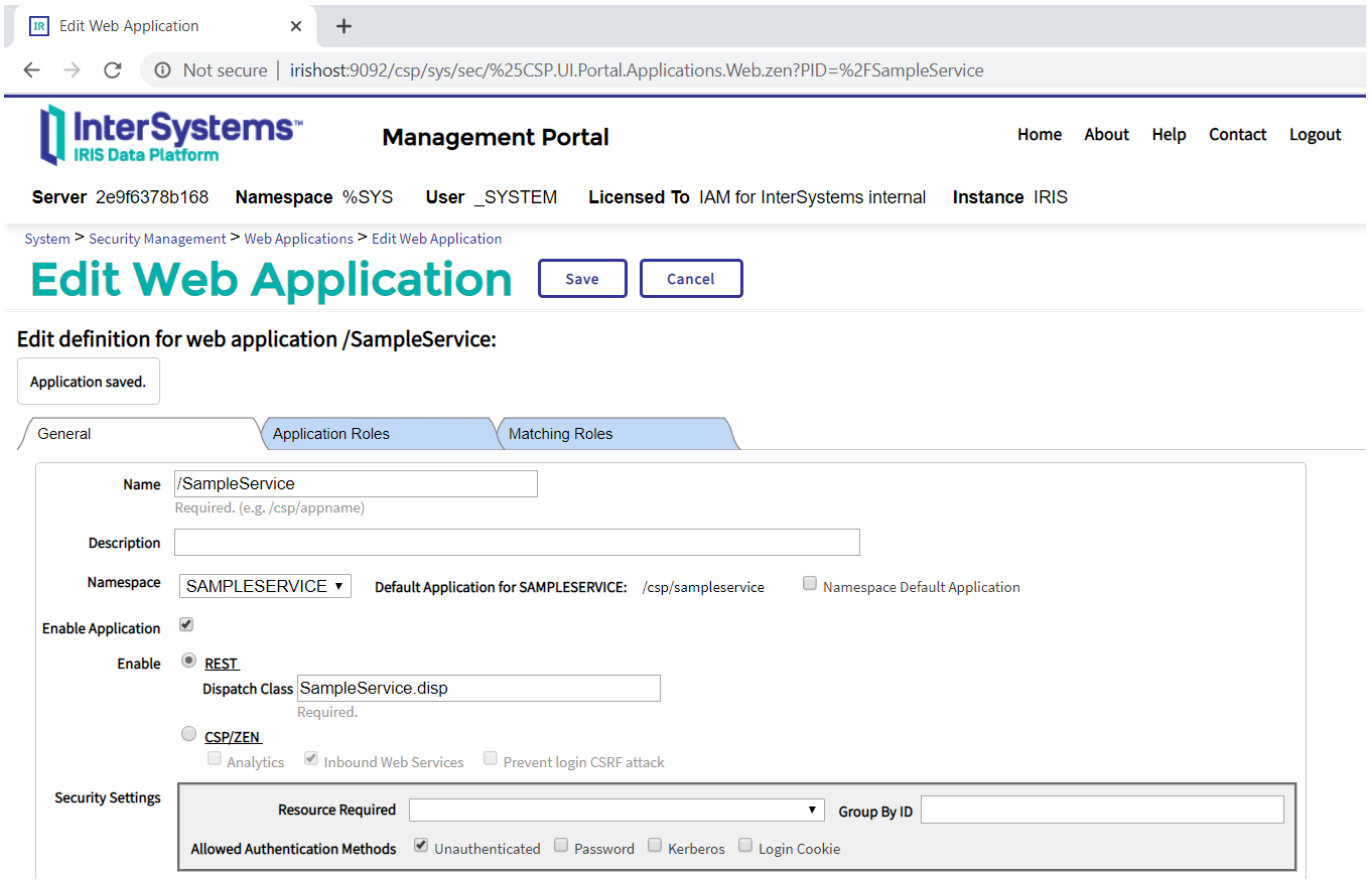
1. ユーザーはクライアントアプリに資格情報（ユーザー名とパスワードなど）を入力します。
2. クライアントアプリは承認サーバーにユーザー資格情報と独自の ID（クライアント ID とクライアントシークレットなど）を送信します。 承認サーバーはユーザー資格情報とクライアント ID を検証し、アクセストークンを返します。
3. クライアントはトークンを使用して、リソースサーバーにあるリソースにアクセスします。
4. リソースサーバーは受け取ったアクセストークンを検証してから、クライアントに情報を返します。

これを踏まえ、IAM を使用して OAuth 2.0 を処理できるシナリオが 2 つあります。

1. IAM はバリデーターをして機能し、クライアントアプリが提供するアクセストークンを検証し、アクセストークンが有効である場合にのみリソースサーバーにリクエストを転送します。この場合、アクセストークンはサードパーティの承認サーバーによって生成されます。
2. IAM は承認サーバーとしてクライアントにアクセストークンを提供し、アクセストークンのバリデーターとしてアクセストークンを検証してから、リソースサーバーにリクエストをリダイレクトします。

IRIS と IAM の定義

この記事では、「/SampleService」という IRIS Web アプリケーションを使用します。
以下のスクリーンショットからわかるように、これは IRIS にデプロイされた認証されていない REST サービスです。



さらに、以下のスクリーンショットのとおり、IAM 側では 1
つのルートを含む「SampleIRISService」というサービスが構成されています。

また、IAM では、IAM で API を呼び出しているユーザーを識別するために、最初に資格情報の無い「CliantApp」というコンシューマーが構成されています。

上記の構成により、IAM は次の URL に送信されるすべての GET リクエストを IRIS にプロキシしています。

http://iamhost:8000/event

この時点では、認証は使用されていません。 したがって、認証無しで単純な GET リクエストを次の URL
に送信する場合、

http://iamhost:8000/event/1

必要なレスポンスを得られます。

この記事では、「PostMan」というアプリを使用してリクエストを送信し、レスポンスを確認します。 以下の PostMan のスクリーンショットでは、単純な GET リクエストとそのレスポンスを確認できます。

着信リクエストに存在するアクセストークンを検証するように IAM
を構成する方法を理解するには、この連載のパート 2 をお読みください。

#API #OAuth2 #REST API #セキュリティ #InterSystems IRIS

ソースURL:https://jp.community.intersystems.com/post/intersystems-api-management%E3%81%A7-oauth-20-%E3%82%92%E4%BD%BF%E3%81%A3%E3%81%A6-api-%E3%82%92%E4%BF%9D%E8%AD%B7%E3%81%99%E3%82%8B-%E3%83%91%E3%83%BC%E3%83%88-1