Article
Guillaume Rongier · Juin 20, 2022 5m de lecture

Sécurisation de vos API avec OAuth 2.0 dans le cadre de la gestion des API d'InterSystems - Partie 1

Introduction

Aujourd'hui, de nombreuses applications utilisent le cadre d'autorisation ouvert (OAuth) pour accéder aux ressources de toutes sortes de services de manière sûre, fiable et efficace. InterSystems IRIS est déjà compatible avec le cadre OAuth 2.0, en fait, il y a un excellent article dans la communauté concernant OAuth 2.0 et InterSystems IRIS dans le lien suivant ici.

Toutefois, avec l'avènement des outils de gestion des API, certaines organisations l'utilisent comme point unique d'authentification, empêchant les demandes non autorisées d'arriver aux services descendants et découplant les complexités d'autorisation/authentification du service lui-même.

Comme vous le savez peut-être, InterSystems a lancé son outil de gestion des API, appelé InterSystems API Management (IAM), qui est disponible avec la licence IRIS Enterprise (et non IRIS Community Edition). Vous trouverez ici un autre excellent article de la communauté présentant InterSystems API Management.

Il s'agit de la première partie d'une série d'articles en trois parties qui montrent comment vous pouvez utiliser IAM pour ajouter simplement de la sécurité, selon les normes OAuth 2.0, à un service précédemment non authentifié déployé dans IRIS.

Dans cette première partie, vous trouverez des informations sur OAuth 2.0 ainsi que des définitions et des configurations initiales d'IRIS et d'IAM afin de faciliter la compréhension de l'ensemble du processus de sécurisation de vos services.

Suite à la première partie, cette série d'articles abordera deux scénarios possibles pour sécuriser vos services avec IAM. Dans le premier scénario, IAM validera uniquement le jeton d'accès présent dans la requête entrante et transmettra la requête au backend si la validation réussit. Dans le second scénario, IAM va à la fois générer un jeton d'accès (en agissant comme un serveur d'autorisation) et le valider.

Par conséquent, la deuxième partie abordera et montrera en détail les étapes nécessaires pour configurer le scénario 1, et la troisième partie abordera et démontrera les configurations pour le scénario 2, ainsi que quelques considérations finales.

Si vous voulez essayer IAM, veuillez contacter votre représentant commercial InterSystems.

OAuth 2.0 : contexte

Chaque flux d'autorisation OAuth 2.0 se compose essentiellement de 4 parties :

1. Utilisateur
2. Client
3. Serveur d'autorisation
4. Propriétaire de la ressource

Pour des raisons de simplicité, cet article utilisera le flux OAuth "Resource Owner Password Credentials" (Identifiants du mot de passe du propriétaire de la ressource), mais vous pouvez utiliser n'importe quel flux OAuth dans IAM. De même, cet article ne spécifiera aucune portée.

Note: Vous ne devez utiliser le flux d'informations d'identification du mot de passe du propriétaire des ressources que lorsque l'application cliente est hautement fiable, car elle traite directement les informations d'identification des utilisateurs. Dans la plupart des cas, le client doit être une application de première partie.

En général, le flux "Resource Owner Password Credentials" (Identifiants du mot de passe du propriétaire de la ressource) suit les étapes suivantes :

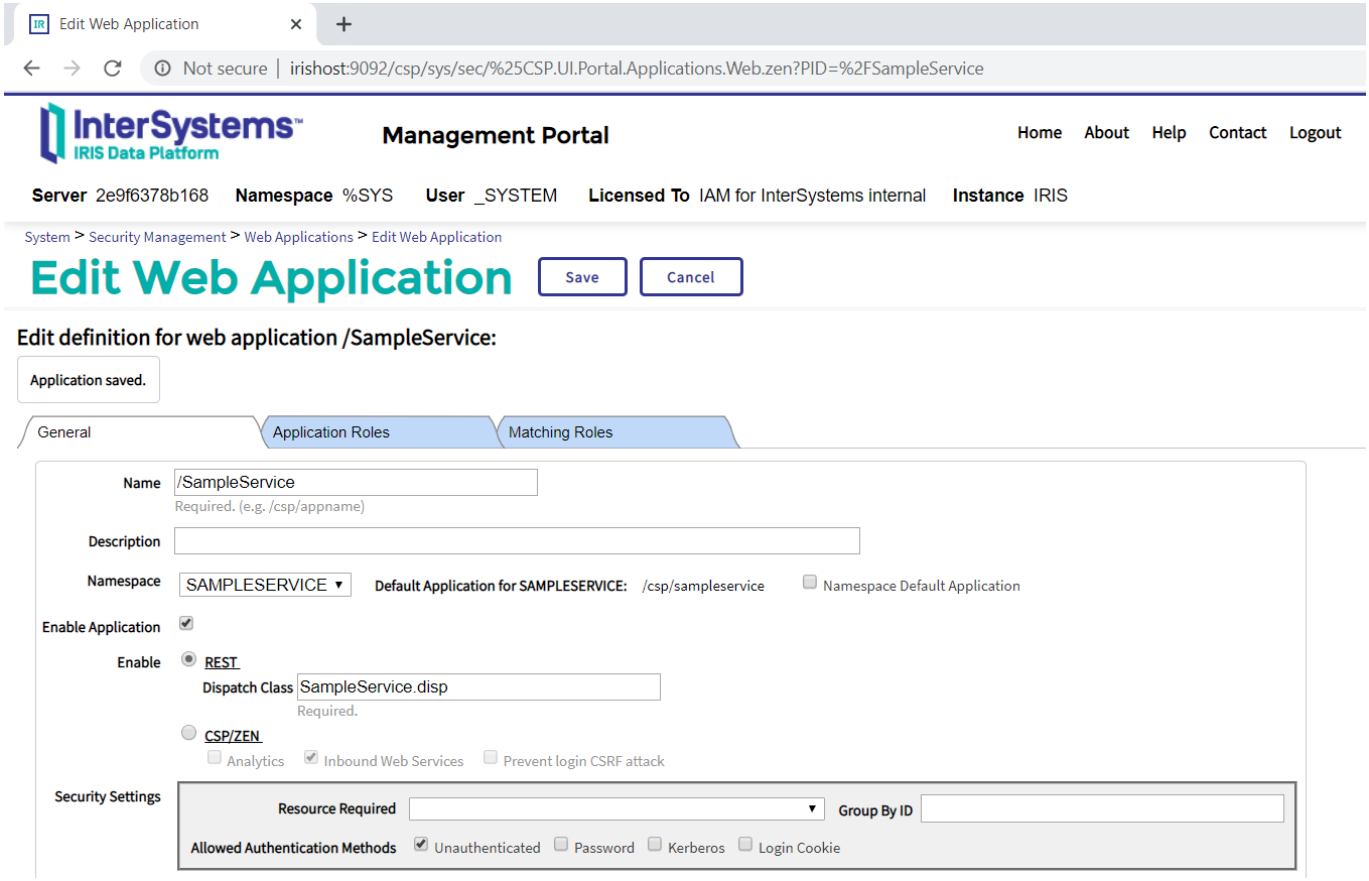
1. L'utilisateur saisit ses identifiants (par exemple le nom d'utilisateur et le mot de passe) dans l'application client.
2. L'application client envoie les identifiants de l'utilisateur ainsi que sa propre identification (identifiant et secret du client, par exemple) au serveur d'autorisation. Le serveur d'autorisation valide les identifiants de l'utilisateur et l'identification du client et renvoie un jeton d'accès.
3. Le client utilise le jeton pour accéder aux ressources du serveur de ressources.
4. Le serveur de ressources valide le jeton d'accès reçu avant de renvoyer toute information au client.

Dans cette optique, il existe deux scénarios dans lesquels vous pouvez utiliser IAM pour traiter OAuth 2.0 :

1. IAM agit comme un validateur, vérifiant le jeton d'accès fourni par l'application cliente, transmettant la demande au serveur de ressources uniquement si le jeton d'accès est valide ; dans ce cas, le jeton d'accès serait généré par un serveur d'autorisation tiers.
2. IAM agissant à la fois comme un serveur d'autorisation, fournissant un jeton d'accès au client, et comme un validateur de jeton d'accès, vérifiant le jeton d'accès avant de rediriger la demande vers le serveur de ressources.

Définitions d'IRIS et d'IAM

Dans ce post, il sera utilisé une application Web IRIS appelée "/SampleService". Comme vous pouvez le voir sur la capture d'écran ci-dessous, il s'agit d'un service REST non authentifié déployé dans IRIS :



En outre, dans le côté IAM est configuré un service appelé "SampleIRISService" contenant un itinéraire, comme vous pouvez le voir dans la capture d'écran ci-dessous :

En outre, dans IAM est configuré un consommateur appelé "ClientApp", initialement sans aucun justificatif d'identité, pour identifier celui qui appelle l'API dans IAM :

Avec les configurations ci-dessus, IAM transmet chaque requête GET envoyée à l'URL suivante à IRIS :

http://iamhost:8000/event

À ce stade, aucune authentification n'est encore utilisée. Par conséquent, si nous envoyons une simple requête GET, sans authentification, à l'URL

http://iamhost:8000/event/1

nous obtenons la réponse recherchée.

Dans cet article, nous allons utiliser une application appelée "PostMan" pour envoyer des requêtes et vérifier les réponses. Dans la capture d'écran de PostMan ci-dessous, vous pouvez voir une simple requête GET ainsi que sa réponse.

Passez à la deuxième partie de cette série pour comprendre comment configurer IAM pour valider les jetons d'accès présents dans les demandes entrantes.

#API #OAuth2 #REST API #Sécurité #InterSystems IRIS

URL de la source:https://fr.community.intersystems.com/post/s%C3%A9curisation-de-vos-api-avec-oauth-20-dans-le-cadre-de-la-gestion-des-api-dintersystems-partie