Article
Guillaume Rongier · Juin 24, 2022 7m de lecture

Sécurisation de vos API avec OAuth 2.0 dans le cadre de la gestion des API d'InterSystems - Partie 3

Dans cette série d'articles en trois parties, il est montré 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 la première partie, nous avons fourni 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.

La deuxième partie discute et montre en détail les étapes nécessaires pour configurer IAM pour valider le jeton d'accès présent dans la demande entrante et transmettre la demande au backend si la validation réussit.

Cette dernière partie de cette série abordera et démontrera les configurations nécessaires pour que IAM génère un jeton d'accès (agissant comme un serveur d'autorisation) et le valide, ainsi que quelques considérations finales importantes.

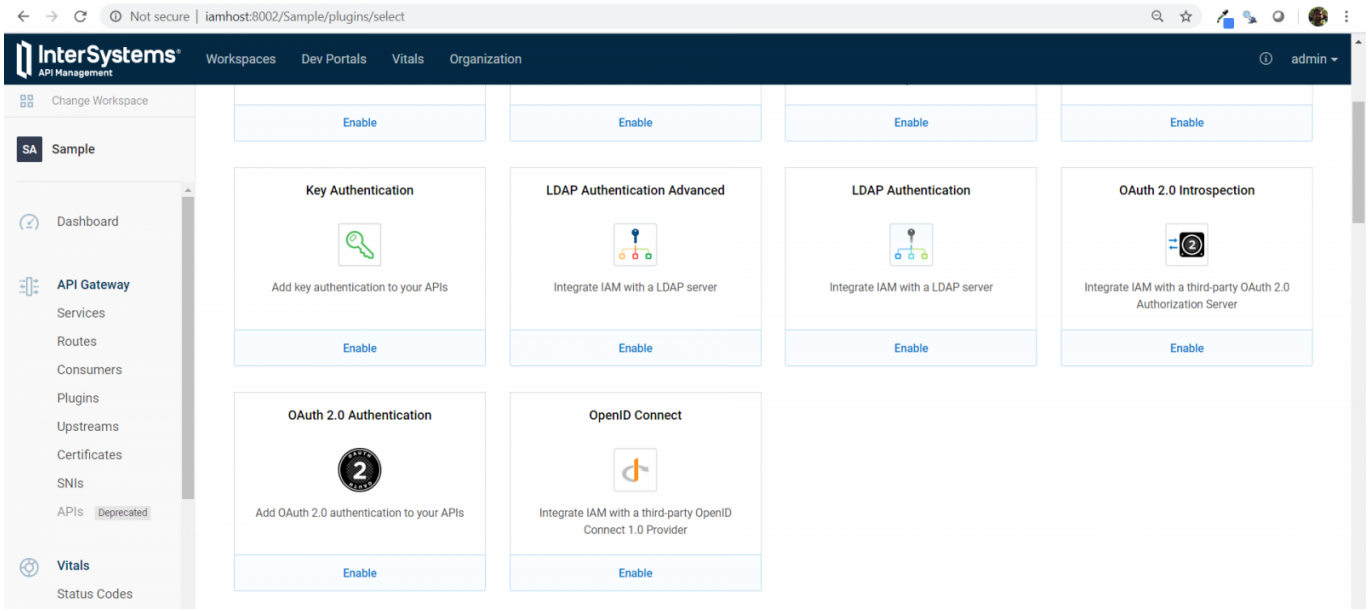
Si vous voulez essayer IAM, veuillez contacter votre représentant commercial d'InterSystems.

Scénario 2 : IAM comme serveur d'autorisation et validateur de jetons d'accès

Dans ce scénario, contrairement au premier scénario, nous allons utiliser un plugin appelé "OAuth 2.0 Authentication".

Afin d'utiliser IAM comme serveur d'autorisation dans ce flux d'informations d'identification du propriétaire des ressources et du mot de passe, le nom d'utilisateur et le mot de passe doivent être authentifiés par l'application cliente. La demande d'obtention du jeton d'accès auprès d'IAM ne doit être effectuée que si l'authentification est réussie.

Commençons par l'ajouter à notre "SampleIRISService". Comme vous pouvez le voir dans la capture d'écran ci-dessous, nous avons différents champs à remplir afin de configurer ce plugin.



Tout d'abord, nous allons copier l'identifiant de notre service "SampleIRISService" dans le champ "serviceid" pour activer ce plugin à notre service.

Dans le champ "config.authheadername", nous allons spécifier le nom de l'en-tête qui portera le jeton d'autorisation. Dans ce cas, je vais laisser la valeur par défaut comme "authorization".

Le plugin "OAuth 2.0 Authentication" prend en charge les flux OAuth 2.0 Authorization Code Grant, Client Credentials, Implicit Grant ou Resource Owner Password Credentials Grant (respectivement, Attribution de codes d'autorisation, Identifiants client, Attribution implicite et Attribution des informations d'identification du mot de passe du propriétaire de la ressource OAuth 2.0). Comme nous utilisons dans cet article le flux "Resource Owner Password Credentials" (Identifiants du mot de passe du propriétaire de la ressource), nous allons cocher la case "config.enablepasswordgrant"..

Dans le champ "config.provisionkey", saisissez une chaîne à utiliser comme clé de provision. Cette valeur sera utilisée pour demander un jeton d'accès à IAM.

Dans ce cas, j'ai laissé tous les autres champs avec la valeur par défaut. Vous pouvez vérifier la référence complète de chaque champ dans la documentation du plugin disponible ici.

Voici à quoi ressemble la configuration du plugin à la fin :

Une fois le plugin créé, nous devons créer les informations d'identification de notre consommateur "ClientApp".

Pour ce faire, allez dans "Consumers" dans le menu de gauche et cliquez sur "ClientApp". Ensuite, cliquez sur l'onglet "Credentials" et ensuite sur le bouton "New OAuth 2.0 Application".

Sur la page suivante, entrez un nom quelconque pour identifier votre application dans le champ "name" (nom), définissez un identifiant et un secret de client, respectivement, dans les champs "clientid" et "clientsecret" et enfin, entrez l'URL dans votre application où les utilisateurs seront envoyés après autorisation dans le champ "redirecturi". Ensuite, cliquez sur "Create" (créer).

Maintenant, vous êtes prêt à envoyer des demandes.

La première requête que nous devons faire est d'obtenir le jeton d'accès depuis IAM. Le plugin "OAuth 2.0 Authentication" crée automatiquement un point de terminaison en ajoutant le chemin "/oauth2/token" à l'itinéraire déjà créé.
Note: Assurez-vous que vous utilisez le protocole HTTPS et que le port proxy d'IAM prête attention aux requêtes TLS/SSL (le port par défaut est 8443). Il s'agit d'une exigence d'OAuth 2.0.

Par conséquent, dans ce cas, nous devrions faire une demande POST à l'URL :

https://iamhost:8443/event/oauth2/token

Dans le corps de la requête, vous devez inclure le JSON suivant :

```
{
"clientid": "clientid",
"clientsecret": "clientsecret",
"granttype": "password",
"provisionkey": "provisionkey",
"authenticateduserid": "1"
}
```

Comme vous pouvez le constater, ce JSON contient des valeurs définies à la fois lors de la création du plugin "OAuth 2.0 Authentication", telles que "granttype" et "provisionkey", et lors de la création des informations d'identification du Consommateur, telles que "clientid" et "clientsecret".

Le paramètre "authenticateduserid" doit également être ajouté par l'application client lorsque le nom d'utilisateur et le mot de passe fournis sont authentifiés avec succès. Sa valeur doit être utilisée pour identifier de manière unique l'utilisateur authentifié.

La demande et sa réponse respective devraient ressembler à ceci :

Avec cela, nous pouvons maintenant faire une requête pour obtenir les données de l'événement en incluant la valeur "accesstoken" de la réponse ci-dessus comme "Bearer Token" (jeton de porteur) dans une requête GET vers l'URL

https://iamhost:8443/event/1

Si votre jeton d'accès expire, vous pouvez générer un nouveau jeton d'accès en utilisant le jeton de renouvellement que vous avez reçu avec le jeton d'accès expiré en effectuant une requête POST vers le même point de terminaison que celui utilisé pour obtenir un jeton d'accès, avec un corps quelque peu différent :

```
{
"clientid": "clientid",
"clientsecret": "clientsecret",
"granttype": "refreshtoken",
"refreshtoken": "E50m6Yd9xWy6lybgo3DOvu5ktZTjzkwF"
}
```

La demande et sa réponse respective devraient ressembler à ceci :

Une fonctionnalité intéressante du plugin "OAuth 2.0 Authentication" est la possibilité d'afficher et d'invalider les jetons d'accès.

Pour répertorier les tokens, envoyez une requête GET au point de terminaison suivant de l'API d'administration d'IAM :

https://iamhost:8444/**{workspacename}/oauth2tokens**

où {workspacename} est le nom de l'espace de travail IAM utilisé. Veillez à saisir les informations d'identification nécessaires pour appeler l'API d'administration d'IAM si vous avez activé RBAC.

Notez que "credentialid" est l'identifiant de l'application OAuth que nous avons créée dans le consommateur ClientApp (dans ce cas, elle s'appelle SampleApp), et "serviceid" est l'identifiant de notre "SampleIRISService" auquel ce plugin est appliqué.

Pour invalider un jeton, vous pouvez envoyer une demande DELETE au point de terminaison suivant

https://iamhost:8444/Sample/oauth2tokens/ **{tokenid}**

où {tokenid} est l'identifiant du jeton à invalider.

Si nous essayons d'utiliser le jeton invalidé, nous obtenons un message indiquant que le jeton est invalide ou a expiré si nous envoyons une requête GET contenant ce jeton invalidé comme jeton porteur à l'URL :

https://iamhost:8443/event/1

Considérations finales

Dans cet article, nous avons montré comment vous pouvez ajouter l'authentification OAuth 2.0 dans IAM à un service non authentifié déployé dans IRIS. Vous devez tenir compte du fait que le service lui-même continuera à être non authentifié dans IRIS. Par conséquent, si quelqu'un appelle directement le point de terminaison du service IRIS, en contournant la couche IAM, il pourra voir les informations sans aucune authentification. Pour cette raison, il est important d'avoir des règles de sécurité au niveau du réseau pour empêcher les demandes non désirées de contourner la couche IAM.

Vous pouvez en savoir plus sur IAM ici.

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

#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-1