

Guide de l'utilisateur BPA Distributed Tracing Framework v5.1

- [Introduction](#)
 - [Zipkin](#)
 - [@opentelemetry/api](#)
- [Composants clés](#)
- [Conditions préalables](#)
 - [Besoins réseau](#)
- [Docker compose le déploiement](#)
 - [Activation du profil de surveillance](#)
 - [Vérification du conteneur Zipkin](#)
- [Activation du suivi dans les services](#)
 - [Hiérarchie de chargement de configuration](#)
 - [Configuration spécifique au pod](#)
 - [Scénario](#)
 - [Exemple de scénario](#)
 - [Configuration globale](#)
 - [Scénario](#)
 - [Exemple de scénario](#)
 - [Configuration de secours](#)
 - [Scénario](#)
 - [Définition de la configuration de suivi](#)
 - [Propriétés de configuration](#)
 - [Application des modifications de configuration](#)
 - [Commande De Mise À Niveau Helm](#)
- [déploiement de Kubernetes et Helm](#)
 - [Activation de la surveillance dans les valeurs](#)
 - [Déploiement du service Zipkin](#)
 - [Vérification du déploiement](#)
 - [Vérification du statut du service](#)
- [Accès et surveillance](#)
 - [Accès à 1 noeud au tableau de bord Zipkin](#)
 - [Accès Kubernetes à 3 noeuds au tableau de bord Zipkin](#)
 - [Utilisation de l'interface Zipkin](#)
- [Traceurs et étendues](#)

Introduction

Zipkin

Zipkin est un système de suivi distribué qui permet de collecter les données de synchronisation nécessaires pour résoudre les problèmes de latence dans l'architecture de service. Ce guide fournit des instructions pour déployer Zipkin et activer le traçage sur les services de la plate-forme BPA.

@opentelemetry/api

Le package `@opentelemetry/api` est l'API principale pour OpenTelemetry dans Node.js. Il fournit des interfaces et des classes pour la création et la gestion des suivis, des étendues et de la propagation de contexte. Ce paquet permet aux développeurs d'instrumenter leurs applications pour collecter des données de télémétrie telles que des traces et des métriques, qui peuvent ensuite être exportées vers des backends comme Zipkin pour analyse.

Composants clés

Les composants clés de Zipkin incluent :

- Serveur Zipkin : Collecteur de suivi central et interface utilisateur (UI)
- OpenTelemetry : Bibliothèque d'instrumentation pour le traçage
- Recherche élastique : Back-end de stockage pour données de suivi
- SSL (Secure Sockets Layer) et TLS (Transport Layer Security) : Assure une communication sécurisée

Conditions préalables

Besoins réseau

Pour déployer Zipkin et activer le traçage, les exigences réseau suivantes sont requises :

- Port 9412 : Point de terminaison de l'API HTTPS (Hypertext Transfer Protocol Secure) Zipkin
- Port 9411 : Point de terminaison de gestion HTTP (Hypertext Transfer Protocol) Zipkin
- Port 9200 : Accès au cluster Elasticsearch

Docker compose le déploiement

Activation du profil de surveillance

1. Démarrez le docker BPA à 1 noeud à partir du chemin suivant :

```
cd /opt/bpa/bpa-{build_version}/scripts
```

2. Exécutez le script bash avec l'option de surveillance à l'aide de la commande suivante :

```
./startbpa.1node.sh monitoring
```

Vérification du conteneur Zipkin

1. Pour vérifier l'état du conteneur, exécutez la commande suivante :

```
docker ps | grep tracers-zipkin
```

2. Pour afficher les journaux d'un conteneur Zipkin. exécutez la commande suivante :

```
docker logs tracers-zipkin -f
```

Activation du suivi dans les services

L'application BPA prend en charge une configuration de traçage flexible via les fichiers « tracingConfig.json ». Le système met en oeuvre un mécanisme de chargement de configuration hiérarchique avec trois (3) niveaux de priorité pour s'adapter à différents scénarios de déploiement.

Hiérarchie de chargement de configuration

L'application charge les configurations de suivi dans l'ordre suivant en fonction de la priorité :

Configuration spécifique au pod

Chemin : bpa-helm-chart/charts/<nom du service>/public_conf

Scénario

- Configuration de traçage individuelle, spécifique à la zone
- Autorise différents paramètres de suivi pour des services ou des modules spécifiques
- Positionnement manuel requis : Les utilisateurs doivent placer manuellement ce fichier dans le chemin d'accès au dossier du pod : bpa-helm-chart/charts/<nom du service>/public_conf

Exemple de scénario

- Activer le suivi détaillé pour les services critiques uniquement
- Différents taux d'échantillonnage pour différents microservices
- Suivi spécifique au débogage pour le dépannage des modules

Configuration globale

Chemin : bpa-helm-chart/bpa/conf/common/globals/tracingConfig.json

Scénario

- Configuration du suivi global pour tous les conteneurs et pods
- Gestion centralisée du suivi
- Impact mondial : Les modifications apportées à ce fichier affectent tous les pods du déploiement

Exemple de scénario

- Activer et désactiver le suivi sur l'ensemble du déploiement BPA
- Définir des stratégies de suivi uniformes pour l'environnement de production
- Exigences de conformité et de surveillance centralisées

Configuration de secours

Chemin : ../conf/tracingConfig.json (secours)

Scénario

- Configuration de suivi par défaut fournie avec l'application
- Environnement de développement par défaut
- Garantit que l'application n'échoue jamais en raison d'une configuration de suivi manquante

Si un fichier de configuration est mal formé ou illisible, le système :

- Consigne l'erreur
- La valeur par défaut est {enable: false} pour empêcher les échecs de suivi
- Poursuit le démarrage des applications

Définition de la configuration de suivi

Créez ou mettez à jour le fichier « tracingConfig.json » avec le contenu suivant dans le chemin d'accès au dossier du pod :

bpa-helm-chart/charts/<nom du service>/public_conf/

Exemples:

- Pour activer le traçage

```
{
    "enable": true
}
```

- Pour désactiver le suivi :

```
{
    "enable": false
}
```

Propriétés de configuration

- enable (booléen)(obligatoire) : Contrôle si le suivi est activé ou désactivé pour l'application

- vrai : Active le suivi distribué
 - L'application initialise l'assistant de traçage
 - Les traces sont collectées et envoyées au serveur principal configuré
 - Fonctionnalités de surveillance et de débogage des performances activées
- false : Désactive le suivi distribué
 - Aucune surcharge de suivi sur les performances des applications
 - Aucune collecte ou transmission de données de suivi
 - L'application s'exécute sans instrumentation de traçage

Application des modifications de configuration

Commande De Mise À Niveau Helm

Après avoir placé les fichiers « tracingConfig.json » aux emplacements appropriés, appliquez les modifications à l'aide de la commande Helm upgrade :

```
helm upgrade bpa-rel --namespace bpa-ns /opt/bpa-helm-chart
```

Ventilation des commandes :

- mise à niveau helm : Met à jour une version Helm existante
- bpa-rel : Nom de la version de la barre BPA
- —espace de noms bpa-ns : Spécifie l'espace de noms Kubernetes où BPA est déployé
- /opt/bpa-helm-chart: Chemin d'accès au répertoire du graphique Helm BPA

déploiement de Kubernetes et Helm

Activation de la surveillance dans les valeurs

Dans le chemin de graphique Helm suivant, modifiez les valeurs Helm dans le fichier « values.yaml » pour activer la surveillance :

```
yaml
global:
  enableMonitoring: true
```

Déploiement du service Zipkin

Déployez le service Zipkin à l'aide de la commande Helm suivante :

```
helm install bpa-rel --create-namespace --namespace bpa-ns
```

Vérification du déploiement

Utilisez les commandes suivantes pour vérifier le déploiement :

```
kubectl get pods -n bpa-ns | grep tracers-zipkin  
kubectl get svc -n bpa-ns | grep tracers-zipkin
```

Vérification du statut du service

Pour afficher les journaux pod, utilisez la commande suivante :

```
kubectl logs -n bpa-ns deployment/tracers-zipkin -f
```

Pour vérifier les points de terminaison de service, utilisez la commande suivante :

```
kubectl describe svc tracers-zipkin -n bpa-ns
```

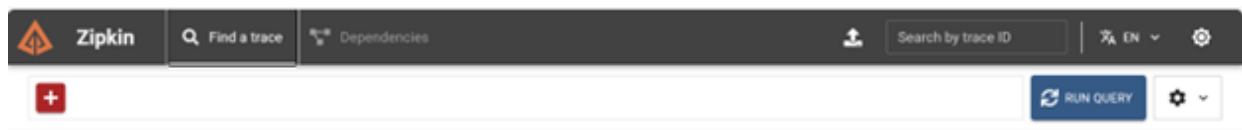
Accès et surveillance

Accès à 1 noeud au tableau de bord Zipkin

URL: <https://<IP DU SERVEUR>:9412/zipkin/>

Accès Kubernetes à 3 noeuds au tableau de bord Zipkin

URL: https://<ip_cluster>:30900/zipkin/



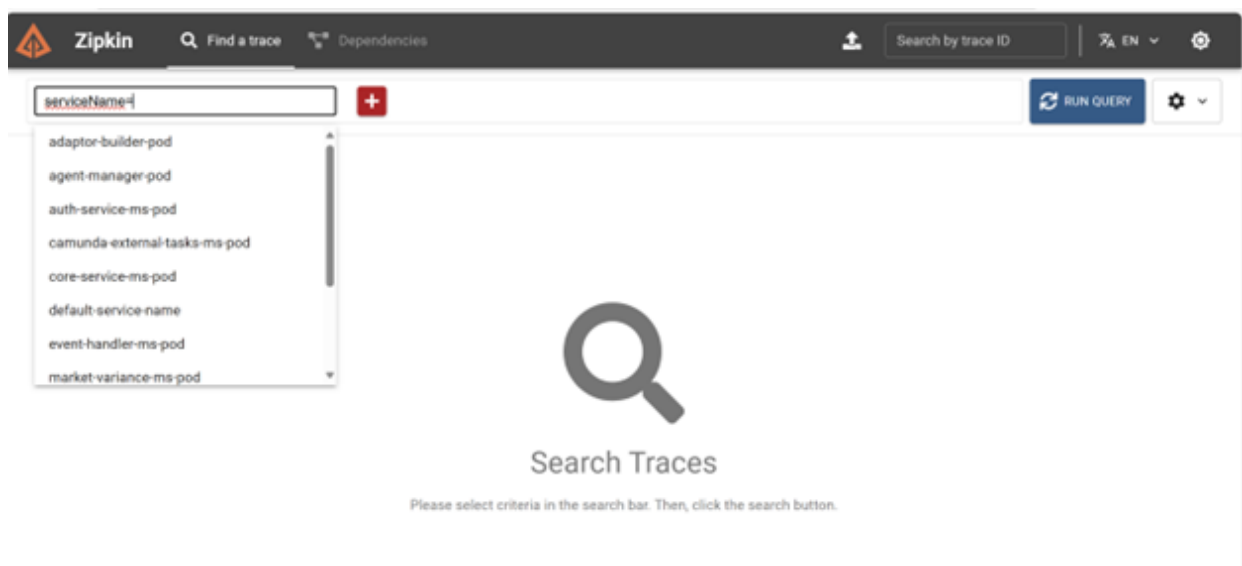
Please select criteria in the search bar. Then, click the search button.

Tableau de bord Zipkin

Utilisation de l'interface Zipkin

Le tableau de bord Zipkin fournit une interface utilisateur pour rechercher et visualiser les traces. Les composants clés sont les suivants :

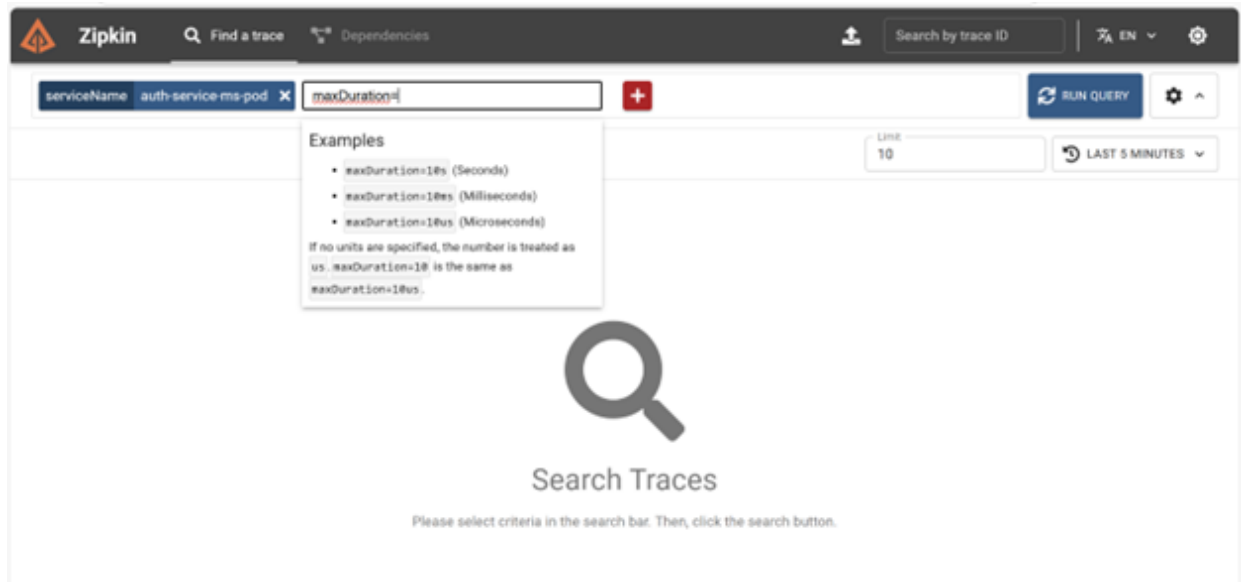
- Liste déroulante Nom du service : Filtrer les suivis par nom de service



Nom du service

Nom du service

- Sélecteur de plage horaire : Définir la fenêtre de temps pour la recherche de trace



Sélecteur De Plage De Temps

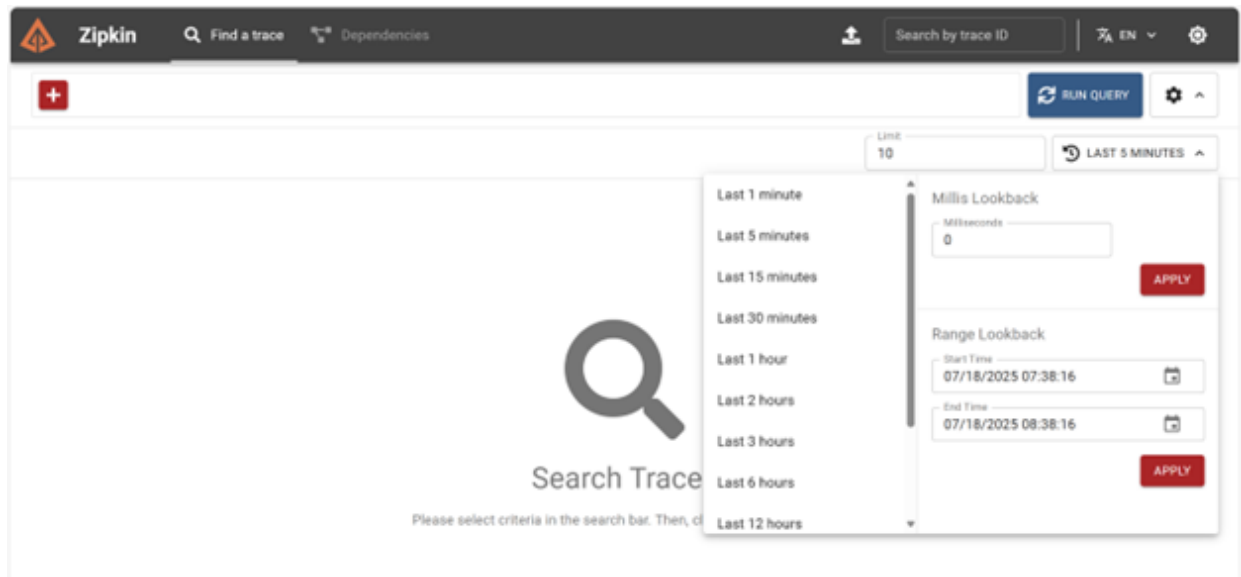
- Liste de suivi : Affiche une liste de suivis correspondant aux critères de recherche

The screenshot shows the Zipkin web interface with search results. The search bar has 'serviceName' and 'auth-service-ms-pod' selected. Below the search bar, there's a 'Limit' dropdown set to '10' and a 'LAST 7 DAYS' button. Below the search bar, there's a table with search results. The table has columns: 'Root', 'Start Time', 'Spans', 'Duration', and 'SHOW'. The table contains 10 rows of search results.

Root	Start Time	Spans	Duration	SHOW
auth-service-ms-pod: pg connect	a minute ago (07/18 08:38:33:184)	1	13.373ms	SHOW
auth-service-ms-pod: pg connect	2 minutes ago (07/18 08:37:49:631)	1	13.353ms	SHOW
auth-service-ms-pod: get /api/v0-9+({0-9}+)(0,2)/internal/tenantsinheaders	2 minutes ago (07/18 08:37:45:107)	26	5.936ms	SHOW
auth-service-ms-pod: get /	a minute ago (07/18 08:38:25:240)	23	2.556ms	SHOW
auth-service-ms-pod: pg.query/select kong	2 minutes ago (07/18 08:37:33:198)	1	740.000us	SHOW
auth-service-ms-pod: pg.query/set kong	2 minutes ago (07/18 08:37:49:644)	1	704.000us	SHOW
auth-service-ms-pod: pg.query/set kong	2 minutes ago (07/18 08:37:33:198)	1	670.000us	SHOW
auth-service-ms-pod: pg.query/set kong	a minute ago (07/18 08:38:33:197)	1	576.000us	SHOW
auth-service-ms-pod: pg.query/select kong	a minute ago (07/18 08:38:33:198)	1	468.000us	SHOW
auth-service-ms-pod: pg.query/select kong	2 minutes ago (07/18 08:37:49:645)	1	443.000us	SHOW

Liste de suivi

- Chronologie du suivi : Représentation visuelle des intervalles de temps (c.-à-d. l'intervalle) dans une trace



Chronologie du suivi

- Détails de la portée : Vue détaillée de chaque plage, y compris la durée, les balises et les journaux.

Span ID	Service name	Span name	Start time	Duration
48df1e037faa526	auth-service-ms-pod	get /api/v[0-9]+/[0-9]+/...	07/18 08:37:45.107	5.936ms
0e0cf964630c0f2	auth-service-ms-pod	middleware - <anonymo...	07/18 08:37:45.107	12.000µs
49900fc36337736	auth-service-ms-pod	middleware - <anonymo...	07/18 08:37:45.107	467.000µs
533ab670a89b1b0	auth-service-ms-pod	middleware - <anonymo...	07/18 08:37:45.107	8.000µs
2a280d577647910f	auth-service-ms-pod	middleware - cookieparser	07/18 08:37:45.107	26.000µs
c83798ab02f1b956	auth-service-ms-pod	middleware - consmiddle...	07/18 08:37:45.107	70.000µs
0c7fa359b5e749b4	auth-service-ms-pod	middleware - expressint	07/18 08:37:45.107	57.000µs
da6798c5dbaa0444	auth-service-ms-pod	middleware - jsonparser	07/18 08:37:45.107	15.000µs
50a2619ac283995	auth-service-ms-pod	middleware - middleware	07/18 08:37:45.107	15.000µs
c75eab2679b57c	auth-service-ms-pod	middleware - query	07/18 08:37:45.107	43.000µs
b697d95823f2e5c	auth-service-ms-pod	middleware - urlencoded...	07/18 08:37:45.107	40.000µs
0803dae4fb0a89ef	auth-service-ms-pod	middleware - <anonymo...	07/18 08:37:45.108	31.000µs
4a04cb7104b66807	auth-service-ms-pod	middleware - <anonymo...	07/18 08:37:45.108	121.000µs
6b6d7396d43d13	auth-service-ms-pod	middleware - <anonymo...	07/18 08:37:45.108	50.000µs
960de40cc8036e0	auth-service-ms-pod	middleware - <anonymo...	07/18 08:37:45.108	19.000µs
b05ed4820ca0288	auth-service-ms-pod	middleware - <anonymo...	07/18 08:37:45.108	24.000µs
c639a16b2633b90	auth-service-ms-pod	middleware - <anonymo...	07/18 08:37:45.108	53.000µs
7e19ee5040e2179	auth-service-ms-pod	middleware - initialize	07/18 08:37:45.108	74.000µs
a2d6a26a2650c47	auth-service-ms-pod	router - /about/cisco-bp...	07/18 08:37:45.108	3.000µs
c0e72796a456b7a	auth-service-ms-pod	router - /about/cisco-bp...	07/18 08:37:45.108	4.000µs

Détails de la portée

Traceurs et étendues

Les traceurs sont responsables de la création et de la gestion des portées. Un traceur est associé à un composant de service ou d'application spécifique. Les portées représentent une unité de travail ou d'opération unique au sein d'une trace. Chaque plage contient des informations telles que le nom de l'opération, les heures de début et de fin, les attributs et les relations parent-enfant

avec d'autres plages.

À propos de cette traduction

Cisco a traduit ce document en traduction automatisée vérifiée par une personne dans le cadre d'un service mondial permettant à nos utilisateurs d'obtenir le contenu d'assistance dans leur propre langue.

Il convient cependant de noter que même la meilleure traduction automatisée ne sera pas aussi précise que celle fournie par un traducteur professionnel.