Concepts

ONTAP Select

NetApp
January 29, 2026

Sommaire

Concepts

Base de services Web REST pour le déploiement et la gestion des clusters ONTAP Select

Le transfert d'état représentatif (REST) est un style de création d'applications web distribuées. Appliqué à la conception d'une API de services web, il établit un ensemble de technologies et de bonnes pratiques pour exposer les ressources serveur et gérer leurs états. Il utilise les protocoles et normes les plus courants pour fournir une base flexible au déploiement et à la gestion des clusters ONTAP Select .

Architecture et contraintes classiques

REST a été formellement articulé par Roy Fielding dans son doctorat "thèse" Créé à l'Université de Californie à Irvine en 2000, il définit un style architectural grâce à un ensemble de contraintes qui, collectivement, améliorent les applications web et les protocoles sous-jacents. Ces contraintes établissent une application de services web RESTful basée sur une architecture client/serveur utilisant un protocole de communication sans état.

Ressources et représentation de l'État

Les ressources sont les composants de base d'un système web. Lors de la création d'une application de services web REST, les premières tâches de conception incluent :

- Identification des ressources système ou serveur. Chaque système utilise et gère des ressources. Une ressource peut être un fichier, une transaction métier, un processus ou une entité administrative. L'une des premières tâches de la conception d'une application basée sur les services Web REST est d'identifier les ressources.
- Définition des états des ressources et des opérations associées. Les ressources sont toujours dans un état parmi un nombre fini. Ces états, ainsi que les opérations associées utilisées pour modifier les états, doivent être clairement définis.

Des messages sont échangés entre le client et le serveur pour accéder et modifier l'état des ressources selon le modèle générique CRUD (Créer, Lire, Mettre à jour et Supprimer).

Points de terminaison URI

Chaque ressource REST doit être définie et rendue disponible selon un schéma d'adressage précis. Les points de terminaison où les ressources sont localisées et identifiées utilisent un identifiant de ressource uniforme (URI). Cet URI fournit un cadre général pour créer un nom unique pour chaque ressource du réseau. L'URL (Uniform Resource Locator) est un type d'URI utilisé par les services web pour identifier et accéder aux ressources. Les ressources sont généralement présentées dans une structure hiérarchique similaire à un répertoire de fichiers.

messages HTTP

Le protocole HTTP (Hypertext Transfer Protocol) est utilisé par le client et le serveur de services Web pour échanger des messages de requête et de réponse concernant les ressources. Lors de la conception d'une application de services Web, les verbes HTTP (tels que GET et POST) sont associés aux ressources et aux actions de gestion d'état correspondantes.

HTTP est un protocole sans état. Par conséquent, pour associer un ensemble de requêtes et de réponses connexes dans une même transaction, des informations supplémentaires doivent être incluses dans les en-têtes HTTP accompagnant les flux de données de requêtes/réponses.

Formatage JSON

Bien que les informations puissent être structurées et transférées entre un client et un serveur de plusieurs manières, l'option la plus courante (et celle utilisée avec l'API REST Deploy) est JavaScript Object Notation (JSON). JSON est une norme industrielle pour la représentation de structures de données simples en texte brut et permet de transférer des informations d'état décrivant les ressources.

Comment accéder à l'API ONTAP Select Deploy

En raison de la flexibilité inhérente aux services Web REST, l'API ONTAP Select Deploy est accessible de plusieurs manières différentes.

Déployer l'interface utilisateur native de l'utilitaire

L'accès à l'API se fait principalement via l'interface utilisateur Web ONTAP Select Deploy. Le navigateur appelle l'API et reformate les données conformément à la conception de l'interface utilisateur. L'accès à l'API se fait également via l'interface de ligne de commande de l'utilitaire Deploy.

Page de documentation en ligne ONTAP Select Deploy

La page de documentation en ligne ONTAP Select Deploy offre un point d'accès alternatif pour l'utilisation d'un navigateur. Outre la possibilité d'exécuter directement des appels d'API individuels, cette page inclut une description détaillée de l'API, incluant les paramètres d'entrée et d'autres options pour chaque appel. Les appels d'API sont organisés en plusieurs domaines ou catégories fonctionnels.

Programme personnalisé

Vous pouvez accéder à l'API Deploy à l'aide de différents langages et outils de programmation. Parmi les plus courants, on trouve Python, Java et cURL. Un programme, un script ou un outil utilisant l'API agit comme un client de services Web REST. L'utilisation d'un langage de programmation vous permet de mieux comprendre l'API et d'automatiser les déploiements ONTAP Select .

Gestion des versions de l'API ONTAP Select Deploy

L'API REST incluse dans ONTAP Select Deploy possède un numéro de version. Ce numéro est indépendant de celui de Deploy. Veuillez connaître la version de l'API incluse dans votre version de Deploy et son impact potentiel sur son utilisation.

La version actuelle de l'utilitaire d'administration Deploy inclut la version 3 de l'API REST. Les versions précédentes de l'utilitaire Deploy incluent les versions d'API suivantes :

Déployer 2.8 et versions ultérieures
: ONTAP Select Deploy 2.8 et toutes les versions ultérieures incluent la version 3 de l'API REST.

Déployer 2.7.2 et versions antérieures
: ONTAP Select Deploy 2.7.2 et toutes les versions antérieures incluent la version 2 de l'API REST.

Les versions 2 et 3 de l'API REST ne sont pas compatibles. Si vous effectuez une mise à niveau vers Deploy 2.8 ou une version ultérieure à partir d'une version antérieure incluant la version 2 de l'API, vous devez mettre à jour tout code existant accédant directement à l'API, ainsi que tous les scripts utilisant l'interface de ligne de commande.

Caractéristiques opérationnelles de base de l'API ONTAP Select Deploy

Bien que REST établisse un ensemble commun de technologies et de bonnes pratiques, les détails de chaque API peuvent varier en fonction des choix de conception. Il est important de connaître les détails et les caractéristiques opérationnelles de l'API ONTAP Select Deploy avant de l'utiliser.

Hôte hyperviseur par rapport au nœud ONTAP Select

Un hôte hyperviseur est la plateforme matérielle principale qui héberge une machine virtuelle ONTAP Select . Lorsqu'une machine virtuelle ONTAP Select est déployée et active sur un hôte hyperviseur, elle est considérée comme un nœud ONTAP Select. Avec la version 3 de l'API REST Deploy, les objets hôte et nœud sont distincts. Cela permet une relation un-à-plusieurs, où un ou plusieurs nœuds ONTAP Select peuvent s'exécuter sur le même hôte hyperviseur.

Identifiants d'objet

Chaque instance ou objet de ressource se voit attribuer un identifiant unique lors de sa création. Ces identifiants sont uniques au sein d'une instance spécifique d' ONTAP Select Deploy. Après un appel d'API créant une instance d'objet, la valeur d'identifiant associée est renvoyée à l'appelant dans le `location` En-tête de la réponse HTTP. Vous pouvez extraire l'identifiant et l'utiliser lors des appels ultérieurs pour faire référence à l'instance de ressource.

Le contenu et la structure interne des identifiants d'objet peuvent changer à tout moment. Vous ne devez utiliser les identifiants que dans les appels d'API applicables, lorsque cela est nécessaire, pour faire référence aux objets associés.

Identifiants de demande

Chaque requête API réussie se voit attribuer un identifiant unique. Cet identifiant est renvoyé dans le `request-id` En-tête de la réponse HTTP associée. Vous pouvez utiliser un identifiant de requête pour désigner collectivement les activités d'une transaction API spécifique. Par exemple, vous pouvez récupérer tous les messages d'événement d'une transaction en fonction de l'identifiant de requête.

Appels synchrones et asynchrones

Il existe deux manières principales par lesquelles un serveur exécute une requête HTTP reçue d'un client :

- Synchrone Le serveur exécute la demande immédiatement et répond avec un code d'état 200, 201 ou 204.
- Asynchrone : le serveur accepte la requête et répond avec le code d'état 202. Cela indique que le serveur a accepté la requête du client et a lancé une tâche en arrière-plan pour la traiter. La réussite ou l'échec final n'est pas immédiatement disponible et doit être déterminé par des appels d'API supplémentaires.

Confirmer l’achèvement d’une tâche de longue durée

En général, toute opération longue est traitée de manière asynchrone via une tâche d’arrière-plan sur le serveur. Avec l’API REST Deploy, chaque tâche d’arrière-plan est ancrée par un objet Job qui la suit et fournit des informations, telles que son état actuel. Un objet Job, avec son identifiant unique, est renvoyé dans la réponse HTTP après la création d’une tâche d’arrière-plan.

Vous pouvez interroger directement l’objet Job pour déterminer la réussite ou l’échec de l’appel d’API associé. Pour plus d’informations, consultez la section « Traitement asynchrone avec l’objet Job ».

Outre l’utilisation de l’objet Job, il existe d’autres moyens de déterminer le succès ou l’échec d’une demande, notamment :

- Messages d’événement : vous pouvez récupérer tous les messages d’événement associés à un appel d’API spécifique à l’aide de l’ID de requête renvoyé avec la réponse d’origine. Ces messages contiennent généralement une indication de réussite ou d’échec et peuvent également être utiles lors du débogage d’une condition d’erreur.
- État ou statut des ressources Plusieurs ressources conservent une valeur d’état ou de statut que vous pouvez interroger pour déterminer indirectement la réussite ou l’échec d’une demande.

Sécurité

L’API de déploiement utilise les technologies de sécurité suivantes :

- Sécurité de la couche transport : tout le trafic envoyé sur le réseau entre le serveur et le client Deploy est chiffré via TLS. L’utilisation du protocole HTTP sur un canal non chiffré n’est pas prise en charge. La version 1.2 de TLS est prise en charge.
- Authentification HTTP : l’authentification de base est utilisée pour chaque transaction API. Un en-tête HTTP, contenant le nom d’utilisateur et le mot de passe dans une chaîne base64, est ajouté à chaque requête.

Transaction API de demande et de réponse pour ONTAP Select

Chaque appel à l’API Deploy est exécuté sous forme de requête HTTP adressée à la machine virtuelle Deploy, qui génère une réponse associée au client. Cette paire requête/réponse est considérée comme une transaction API. Avant d’utiliser l’API Deploy, vous devez vous familiariser avec les variables d’entrée disponibles pour contrôler une requête et le contenu de la réponse.

Variables d’entrée contrôlant une requête API

Vous pouvez contrôler la manière dont un appel API est traité via des paramètres définis dans la requête HTTP.

En-têtes de requête

Vous devez inclure plusieurs en-têtes dans la requête HTTP, notamment :

- type de contenu Si le corps de la demande inclut JSON, cet en-tête doit être défini sur application/json.

- accepter Si le corps de la réponse doit inclure du JSON, cet en-tête doit être défini sur application/json.
- L'authentification de base doit être définie avec le nom d'utilisateur et le mot de passe codés dans une chaîne base64.

Corps de la requête

Le contenu du corps de la requête varie selon l'appel. Le corps de la requête HTTP se compose de l'un des éléments suivants :

- Objet JSON avec variables d'entrée (comme le nom d'un nouveau cluster)
- Vide

Filtrer les objets

Lors d'un appel d'API utilisant GET, vous pouvez limiter ou filtrer les objets renvoyés en fonction de n'importe quel attribut. Par exemple, vous pouvez spécifier une valeur exacte à laquelle correspondre :

```
<field>=<query value>
```

Outre une correspondance exacte, d'autres opérateurs permettent de renvoyer un ensemble d'objets sur une plage de valeurs. ONTAP Select prend en charge les opérateurs de filtrage présentés ci-dessous.

Opérateur	Description
=	Égal à
<	Moins que
>	Plus grand que
⇐	Inférieur ou égal à
>=	Supérieur ou égal à
	Ou
!	Pas égal à
*	Caractère générique gourmand

Vous pouvez également renvoyer un ensemble d'objets selon qu'un champ spécifique est défini ou non en utilisant le mot-clé null ou sa négation (!null) dans le cadre de la requête.

Sélection des champs d'objet

Par défaut, un appel d'API via GET ne renvoie que les attributs identifiant de manière unique le ou les objets. Cet ensemble minimal de champs sert de clé pour chaque objet et varie selon son type. Vous pouvez sélectionner des propriétés d'objet supplémentaires à l'aide du paramètre de requête fields, comme suit :

- Champs peu coûteux Spécifier `fields=*` pour récupérer les champs d'objet qui sont conservés dans la mémoire du serveur local ou qui nécessitent peu de traitement pour y accéder.
- Champs coûteux Spécifier `fields=**` pour récupérer tous les champs de l'objet, y compris ceux nécessitant un traitement serveur supplémentaire pour y accéder.
- Sélection de champ personnalisé Utiliser `fields=FIELDNAME` pour spécifier le champ exact souhaité. Lorsque vous demandez plusieurs champs, les valeurs doivent être séparées par des virgules sans

espaces.

> Il est recommandé de toujours identifier les champs spécifiques souhaités. Ne récupérez les champs les moins chers ou les plus coûteux que lorsque cela est nécessaire. La classification des champs les moins chers et les plus coûteux est déterminée par NetApp sur la base d'une analyse interne des performances. La classification d'un champ donné peut changer à tout moment.

Trier les objets dans l'ensemble de sortie

Les enregistrements d'une collection de ressources sont renvoyés dans l'ordre par défaut défini par l'objet. Vous pouvez modifier cet ordre à l'aide du paramètre de requête order_by, en indiquant le nom du champ et le sens de tri comme suit :

```
order_by=<field name> asc|desc
```

Par exemple, vous pouvez trier le champ type par ordre décroissant suivi de l'ID par ordre croissant :

```
order_by=type desc, id asc
```

Lorsque vous incluez plusieurs paramètres, vous devez séparer les champs par une virgule.

Pagination

Lors d'un appel API via GET pour accéder à une collection d'objets de même type, tous les objets correspondants sont renvoyés par défaut. Si nécessaire, vous pouvez limiter le nombre d'enregistrements renvoyés en utilisant le paramètre de requête max_records. Par exemple :

```
max_records=20
```

Si nécessaire, vous pouvez combiner ce paramètre avec d'autres paramètres de requête pour affiner les résultats. Par exemple, la requête suivante renvoie jusqu'à 10 événements système générés après l'heure spécifiée :

```
time⇒ 2019-04-04T15:41:29.140265Z&max_records=10
```

Vous pouvez émettre plusieurs requêtes pour parcourir les événements (ou tout type d'objet). Chaque appel d'API ultérieur doit utiliser une nouvelle valeur temporelle basée sur le dernier événement du dernier ensemble de résultats.

Interpréter une réponse API

Chaque requête API génère une réponse au client. Vous pouvez examiner cette réponse pour déterminer si elle a abouti et récupérer des données supplémentaires si nécessaire.

Code d'état HTTP

Les codes d'état HTTP utilisés par l'API REST Deploy sont décrits ci-dessous.

Code	Signification	Description
200	OK	Indique le succès des appels qui ne créent pas de nouvel objet.
201	Créé	Un objet est créé avec succès ; l'en-tête de réponse d'emplacement inclut l'identifiant unique de l'objet.
202	Accepté	Une tâche d'arrière-plan de longue durée a été démarrée pour exécuter la demande, mais l'opération n'est pas encore terminée.

Code	Signification	Description
400	Mauvaise demande	La demande d'entrée n'est pas reconnue ou est inappropriée.
403	Interdit	L'accès est refusé en raison d'une erreur d'autorisation.
404	Non trouvé	La ressource référencée dans la demande n'existe pas.
405	Méthode non autorisée	Le verbe HTTP dans la requête n'est pas pris en charge pour la ressource.
409	Conflit	Une tentative de création d'un objet a échoué car l'objet existe déjà.
500	Erreur interne	Une erreur interne générale s'est produite sur le serveur.
501	Non implémenté	L'URI est connu mais n'est pas capable d'exécuter la requête.

En-têtes de réponse

Plusieurs en-têtes sont inclus dans la réponse HTTP générée par le serveur de déploiement, notamment :

- request-id Chaque demande d'API réussie se voit attribuer un identifiant de demande unique.
- emplacement Lorsqu'un objet est créé, l'en-tête d'emplacement inclut l'URL complète du nouvel objet, y compris l'identifiant d'objet unique.

Corps de la réponse

Le contenu de la réponse associée à une requête API varie selon l'objet, le type de traitement et la réussite ou l'échec de la requête. Le corps de la réponse est affiché au format JSON.

- Objet unique : un objet unique peut être renvoyé avec un ensemble de champs en fonction de la requête. Par exemple, vous pouvez utiliser GET pour récupérer les propriétés sélectionnées d'un cluster à l'aide de son identifiant unique.
- Objets multiples : plusieurs objets d'une collection de ressources peuvent être renvoyés. Dans tous les cas, un format cohérent est utilisé, avec `num_records` indiquant le nombre d'enregistrements et les enregistrements contenant un tableau d'instances d'objet. Par exemple, vous pouvez récupérer tous les nœuds définis dans un cluster spécifique.
- Objet Job : si un appel d'API est traité de manière asynchrone, un objet Job est renvoyé, qui ancre la tâche en arrière-plan. Par exemple, la requête POST utilisée pour déployer un cluster est traitée de manière asynchrone et renvoie un objet Job.
- Objet d'erreur : si une erreur se produit, un objet d'erreur est toujours renvoyé. Par exemple, vous recevrez une erreur lorsque vous tenterez de créer un cluster avec un nom qui existe déjà.
- Vide : Dans certains cas, aucune donnée n'est renvoyée et le corps de la réponse est vide. Par exemple, le corps de la réponse est vide après l'utilisation de DELETE pour supprimer un hôte existant.

Traitement asynchrone à l'aide de l'objet Job pour ONTAP Select

Certains appels d'API Deploy, notamment ceux qui créent ou modifient une ressource, peuvent prendre plus de temps que d'autres. ONTAP Select Deploy traite ces requêtes de longue durée de manière asynchrone.

Requêtes asynchrones décrites à l'aide de l'objet Job

Après un appel d'API exécuté de manière asynchrone, le code de réponse HTTP 202 indique que la requête a été validée et acceptée, mais pas encore terminée. La requête est traitée en tâche d'arrière-plan et continue de s'exécuter après la réponse HTTP initiale au client. La réponse inclut l'objet Job ancrant la requête, ainsi que son identifiant unique.

Vous devez vous référer à la page de documentation en ligne ONTAP Select Deploy pour déterminer quels appels d'API fonctionnent de manière asynchrone.

Interroger l'objet Job associé à une requête API

L'objet Job renvoyé dans la réponse HTTP contient plusieurs propriétés. Vous pouvez interroger la propriété « state » pour déterminer si la requête a abouti. Un objet Job peut être dans l'un des états suivants :

- En file d'attente
- Exécution
- Succès
- Échec

Il existe deux techniques que vous pouvez utiliser lors de l'interrogation d'un objet Job pour détecter un état terminal pour la tâche, soit un succès, soit un échec :

- Demande d'interrogation standard L'état actuel du travail est renvoyé immédiatement
- Demande d'interrogation longue L'état du travail est renvoyé uniquement lorsque l'un des événements suivants se produit :
  - L'état a changé plus récemment que la valeur de date et d'heure fournie dans la demande de sondage
  - La valeur du délai d'attente a expiré (1 à 120 secondes)

Les interrogations standard et longue utilisent le même appel d'API pour interroger un objet Job. Cependant, une requête longue inclut deux paramètres de requête : `poll_timeout` et `last_modified` .

Vous devez toujours utiliser une interrogation longue pour réduire la charge de travail sur la machine virtuelle Deploy.

Procédure générale pour émettre une requête asynchrone

Vous pouvez utiliser la procédure de haut niveau suivante pour terminer un appel d'API asynchrone :

1. Émettez l'appel API asynchrone.
2. Recevez une réponse HTTP 202 indiquant l'acceptation réussie de la demande.
3. Extraire l'identifiant de l'objet Job du corps de la réponse.
4. Dans une boucle, effectuez les opérations suivantes à chaque cycle :
   a. Obtenez l'état actuel du travail avec une requête longue durée
   b. Si le travail est dans un état non terminal (en file d'attente, en cours d'exécution), exécutez à nouveau la boucle.
5. Arrêtez-vous lorsque le travail atteint un état terminal (succès, échec).