



# **Conceitos**

## **ONTAP Select**

NetApp  
February 03, 2026

# Índice

Conceitos .....	1
Base de serviços web REST .....	1
Arquitetura e restrições clássicas .....	1
Recursos e representação do Estado .....	1
Pontos de extremidade URI .....	1
Mensagens HTTP .....	1
Formatação JSON .....	2
Como acessar a API Deploy .....	2
Implantar interface de usuário nativa do utilitário .....	2
Página de documentação on-line do ONTAP Select Deploy .....	2
Programa personalizado .....	2
Implantar o controle de versão da API .....	2
Caraterísticas operacionais básicas .....	3
Host de hipervisor versus nó ONTAP Select .....	3
Identificadores de objeto .....	3
Solicitar identificadores .....	3
Chamadas síncronas e assíncronas .....	3
Confirme a conclusão de um trabalho de longa duração .....	4
Segurança .....	4
Transação de API de solicitação e resposta .....	4
Variáveis de entrada que controlam uma solicitação de API .....	4
Interpretar uma resposta da API .....	6
Processamento assíncrono usando o objeto de tarefa .....	7
Solicitações assíncronas descritas usando o objeto Job .....	7
Consulte o objeto Job associado a uma solicitação de API .....	8
Procedimento geral para emitir uma solicitação assíncrona .....	8

# Conceitos

## Base de serviços web REST

Representational State Transfer (REST) é um estilo para a criação de aplicações web distribuídas. Quando aplicada ao design de uma API de serviços da Web, ela estabelece um conjunto de tecnologias e melhores práticas para expor recursos baseados em servidor e gerenciar seus estados. Ele usa protocolos e padrões convencionais para fornecer uma base flexível para implantar e gerenciar clusters ONTAP Select.

### Arquitetura e restrições clássicas

REST foi formalmente articulado por Roy Fielding em seu doutorado "[dissertação](#)" na UC Irvine em 2000. Ele define um estilo arquitetônico por meio de um conjunto de restrições, que coletivamente melhoraram as aplicações baseadas na Web e os protocolos subjacentes. As restrições estabelecem um aplicativo de serviços da Web RESTful com base em uma arquitetura cliente/servidor usando um protocolo de comunicação sem estado.

### Recursos e representação do Estado

Os recursos são os componentes básicos de um sistema baseado na Web. Ao criar um aplicativo REST de serviços da Web, as tarefas iniciais de design incluem:

- Identificação de recursos baseados em sistema ou servidor que cada sistema usa e mantém recursos. Um recurso pode ser um arquivo, transação comercial, processo ou entidade administrativa. Uma das primeiras tarefas no projeto de um aplicativo baseado em serviços web REST é identificar os recursos.
- Definição de estados de recursos e recursos de operações estatais associadas estão sempre em um de um número finito de estados. Os estados, bem como as operações associadas usadas para afetar as mudanças de estado, devem ser claramente definidos.

As mensagens são trocadas entre o cliente e o servidor para acessar e alterar o estado dos recursos de acordo com o modelo genérico CRUD (criar, ler, atualizar e excluir).

### Pontos de extremidade URI

Todos os recursos REST devem ser definidos e disponibilizados usando um esquema de endereçamento bem definido. Os endpoints onde os recursos estão localizados e identificados usam um URI (Uniform Resource Identifier). O URI fornece uma estrutura geral para criar um nome exclusivo para cada recurso na rede. O Uniform Resource Locator (URL) é um tipo de URI usado com serviços da Web para identificar e acessar recursos. Os recursos são normalmente expostos em uma estrutura hierárquica semelhante a um diretório de arquivos.

### Mensagens HTTP

O Hypertext Transfer Protocol (HTTP) é o protocolo usado pelo cliente e servidor de serviços da Web para trocar mensagens de solicitação e resposta sobre os recursos. Como parte do projeto de um aplicativo de serviços da Web, os verbos HTTP (como GET e POST) são mapeados para os recursos e ações de gerenciamento de estado correspondentes.

HTTP está sem estado. Portanto, para associar um conjunto de solicitações e respostas relacionadas em uma transação, informações adicionais devem ser incluídas nos cabeçalhos HTTP carregados com os fluxos de

dados de solicitação/resposta.

## Formatação JSON

Embora as informações possam ser estruturadas e transferidas entre um cliente e um servidor de várias maneiras, a opção mais popular (e a usada com a API REST de implantação) é JavaScript Object Notation (JSON). JSON é um padrão da indústria para representar estruturas de dados simples em texto simples e é usado para transferir informações de estado descrevendo os recursos.

## Como acessar a API Deploy

Devido à flexibilidade inerente dos serviços da Web REST, a API de implantação do ONTAP Select pode ser acessada de várias maneiras diferentes.

### Implantar interface de usuário nativa do utilitário

A principal maneira de acessar a API é por meio da interface de usuário da Web do ONTAP Select Deploy. O navegador faz chamadas para a API e reformata os dados de acordo com o design da interface do usuário. Você também acessa a API por meio da interface de linha de comando Deploy Utility.

### Página de documentação on-line do ONTAP Select Deploy

A página de documentação on-line do ONTAP Select Deploy fornece um ponto de acesso alternativo ao usar um navegador. Além de fornecer uma maneira de executar chamadas individuais de API diretamente, a página também inclui uma descrição detalhada da API, incluindo parâmetros de entrada e outras opções para cada chamada. As chamadas de API são organizadas em várias áreas ou categorias funcionais diferentes.

### Programa personalizado

Você pode acessar a API Deploy usando qualquer uma das várias linguagens e ferramentas de programação diferentes. As escolhas populares incluem Python, Java e curl. Um programa, script ou ferramenta que usa a API atua como um cliente de serviços da Web REST. O uso de uma linguagem de programação permite que você entenda melhor a API e oferece a oportunidade de automatizar as implantações do ONTAP Select.

## Implantar o controle de versão da API

A API REST incluída no ONTAP Select Deploy recebe um número de versão. O número da versão da API é independente do número da versão de implantação. Você deve estar ciente da versão da API incluída na versão de implantação e como isso pode afetar o uso da API.

A versão atual do utilitário de administração implantar inclui a versão 3 da API REST. As versões anteriores do utilitário deploy incluem as seguintes versões de API:

### Implantar o 2,8 e posterior

O ONTAP Select Deploy 2,8 e todas as versões posteriores incluem a versão 3 da API REST.

### Implantar o 2.7.2 e versões anteriores

O ONTAP Select Deploy 2.7.2 e todas as versões anteriores incluem a versão 2 da API REST.



As versões 2 e 3 da API REST não são compatíveis. Se você atualizar para implantar o 2,8 ou posterior a partir de uma versão anterior que incluía a versão 2 da API, você deve atualizar qualquer código existente que acesse diretamente a API, bem como quaisquer scripts usando a interface de linha de comando.

## Caraterísticas operacionais básicas

Embora O REST estabeleça um conjunto comum de tecnologias e práticas recomendadas, os detalhes de cada API podem variar com base nas escolhas de design. Você deve estar ciente dos detalhes e das caraterísticas operacionais da API de implantação do ONTAP Select antes de usar a API.

### Host de hipervisor versus nó ONTAP Select

Um *hypervisor host* é a plataforma de hardware principal que hospeda uma máquina virtual ONTAP Select. Quando uma máquina virtual ONTAP Select é implantada e ativa em um host de hipervisor, a máquina virtual é considerada um nó *ONTAP Select*. Com a versão 3 da API REST de implantação, os objetos de host e nó são separados e distintos. Isso permite uma relação um-para-muitos, em que um ou mais nós de ONTAP Select podem ser executados no mesmo host de hipervisor.

### Identificadores de objeto

Cada instância ou objeto de recurso recebe um identificador exclusivo quando é criado. Esses identificadores são globalmente exclusivos em uma instância específica do ONTAP Select Deploy. Depois de emitir uma chamada de API que cria uma nova instância de objeto, o valor de id associado é retornado ao chamador no `location` cabeçalho da resposta HTTP. Você pode extrair o identificador e usá-lo em chamadas subsequentes quando se refere à instância de recurso.



O conteúdo e a estrutura interna dos identificadores de objeto podem mudar a qualquer momento. Você só deve usar os identificadores nas chamadas de API aplicáveis conforme necessário ao se referir aos objetos associados.

### Solicitar identificadores

Cada solicitação de API bem-sucedida é atribuído um identificador exclusivo. O identificador é retornado no `request-id` cabeçalho da resposta HTTP associada. Você pode usar um identificador de solicitação para se referir coletivamente às atividades de uma única transação de solicitação-resposta de API específica. Por exemplo, você pode recuperar todas as mensagens de evento de uma transação com base no ID de solicitação.

### Chamadas síncronas e assíncronas

Há duas maneiras principais pelas quais um servidor executa uma solicitação HTTP recebida de um cliente:

- Síncrono o servidor executa a solicitação imediatamente e responde com um código de status 200, 201 ou 204.
- Assíncrono o servidor aceita a solicitação e responde com um código de status 202. Isso indica que o servidor aceitou a solicitação do cliente e iniciou uma tarefa em segundo plano para concluir a solicitação. O sucesso ou falha final não está disponível imediatamente e deve ser determinado por meio de chamadas de API adicionais.

## Confirme a conclusão de um trabalho de longa duração

Geralmente, qualquer operação que possa levar muito tempo para ser concluída é processada assincronamente usando uma tarefa em segundo plano no servidor. Com a API Deploy REST, cada tarefa em segundo plano é ancorada por um objeto Job que rastreia a tarefa e fornece informações, como o estado atual. Um objeto Job, incluindo seu identificador exclusivo, é retornado na resposta HTTP depois que uma tarefa em segundo plano é criada.

Você pode consultar o objeto Job diretamente para determinar o sucesso ou falha da chamada API associada. Consulte *processamento assíncrono usando o objeto Job* para obter informações adicionais.

Além de usar o objeto Job, existem outras maneiras de determinar o sucesso ou falha de uma solicitação, incluindo:

- Você pode recuperar todas as mensagens de evento associadas a uma chamada de API específica usando o ID de solicitação retornado com a resposta original. As mensagens de evento geralmente contêm uma indicação de sucesso ou falha, e também podem ser úteis ao depurar uma condição de erro.
- Estado ou status do recurso vários dos recursos mantêm um estado ou valor de status que você pode consultar para determinar indiretamente o sucesso ou falha de uma solicitação.

## Segurança

A API Deploy usa as seguintes tecnologias de segurança:

- Segurança da camada de transporte todo o tráfego enviado pela rede entre o servidor de implantação e o cliente é criptografado por meio do TLS. O uso do protocolo HTTP em um canal não criptografado não é suportado. O TLS versão 1,2 é suportado.
- Autenticação HTTP a autenticação básica é usada para cada transação de API. Um cabeçalho HTTP, que inclui o nome de usuário e senha em uma cadeia de caracteres base64, é adicionado a cada solicitação.

## Transação de API de solicitação e resposta

Cada chamada de API de implantação é executada como uma solicitação HTTP para a máquina virtual de implantação que gera uma resposta associada ao cliente. Esse par de solicitação/resposta é considerado uma transação de API. Antes de usar a API Deploy, você deve estar familiarizado com as variáveis de entrada disponíveis para controlar uma solicitação e o conteúdo da saída de resposta.

## Variáveis de entrada que controlam uma solicitação de API

Você pode controlar como uma chamada de API é processada por meio de parâmetros definidos na solicitação HTTP.

### Cabeçalhos de solicitação

Você deve incluir vários cabeçalhos na solicitação HTTP, incluindo:

- Se o corpo da solicitação incluir JSON, esse cabeçalho deve ser definido como `application/json`.
- Aceitar se o corpo da resposta incluir JSON, esse cabeçalho deve ser definido como `application/json`.
- A autenticação básica de autorização deve ser definida com o nome de usuário e senha codificados em uma cadeia de caracteres base64.

## Corpo do pedido

O conteúdo do corpo da solicitação varia de acordo com a chamada específica. O corpo da solicitação HTTP consiste em um dos seguintes:

- Objeto JSON com variáveis de entrada (como o nome de um novo cluster)
- Vazio

## Filtrar objetos

Ao emitir uma chamada de API que usa GET, você pode limitar ou filtrar os objetos retornados com base em qualquer atributo. Por exemplo, você pode especificar um valor exato para corresponder:

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

Além de uma correspondência exata, há outros operadores disponíveis para retornar um conjunto de objetos em uma faixa de valores. O ONTAP Select suporta os operadores de filtragem mostrados abaixo.

Operador	Descrição
.	Igual a.
*	Menos de
>	Superior a.
O que é que eu tenho	Inferior ou igual a
>	Maior ou igual a
	Ou
!	Não é igual a
*	Wildcard ganancioso

Você também pode retornar um conjunto de objetos com base se um campo específico está definido ou não usando a palavra-chave null ou sua negação (!null) como parte da consulta.

## Selecionar campos de objeto

Por padrão, a emissão de uma chamada de API usando O GET retorna apenas os atributos que identificam exclusivamente o objeto ou objetos. Este conjunto mínimo de campos atua como uma chave para cada objeto e varia de acordo com o tipo de objeto. Você pode selecionar propriedades de objeto adicionais usando o parâmetro de consulta campos das seguintes maneiras:

- Campos de baixo custo especificam `fields=*` para recuperar os campos de objeto que são mantidos na memória do servidor local ou requerem pouco processamento para acessar.
- Campos caros especificam `fields=**` para recuperar todos os campos de objeto, incluindo aqueles que exigem processamento adicional de servidor para acessar.
- Seleção de campo personalizada Use `fields=FIELDNAME` para especificar o campo exato desejado. Ao solicitar vários campos, os valores devem ser separados usando vírgulas sem espaços.



Como prática recomendada, você deve sempre identificar os campos específicos que deseja. Você só deve recuperar o conjunto de campos baratos ou caros quando necessário. A classificação barata e cara é determinada pelo NetApp com base na análise interna de desempenho. A classificação de um determinado campo pode mudar a qualquer momento.

## Classificar objetos no conjunto de saída

Os Registros em uma coleção de recursos são retornados na ordem padrão definida pelo objeto. Você pode alterar a ordem usando o parâmetro de consulta `order_by` com o nome do campo e a direção de ordenação da seguinte forma:

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

Por exemplo, você pode classificar o campo `tipo` em ordem decrescente seguido de `id` em ordem crescente:

```
order_by=tipo desc, id asc
```

Ao incluir vários parâmetros, você deve separar os campos com uma vírgula.

## Paginação

Ao emitir uma chamada de API usando GET para acessar uma coleção de objetos do mesmo tipo, todos os objetos correspondentes são retornados por padrão. Se necessário, você pode limitar o número de Registros retornados usando o parâmetro de consulta `Max_Records` com a solicitação. Por exemplo:

```
max_records=20
```

Se necessário, você pode combinar este parâmetro com outros parâmetros de consulta para restringir o conjunto de resultados. Por exemplo, o seguinte retorna até 10 eventos do sistema gerados após o tempo especificado:

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

Você pode emitir várias solicitações para percorrer os eventos (ou qualquer tipo de objeto). Cada chamada de API subsequente deve usar um novo valor de tempo com base no evento mais recente no último conjunto de resultados.

## Interpretar uma resposta da API

Cada solicitação de API gera uma resposta de volta ao cliente. Você pode examinar a resposta para determinar se ela foi bem-sucedida e recuperar dados adicionais, conforme necessário.

## Código de status HTTP

Os códigos de status HTTP usados pela API REST de implantação são descritos abaixo.

Código	Significado	Descrição
200	OK	Indica sucesso para chamadas que não criam um novo objeto.
201	Criado	Um objeto é criado com sucesso; o cabeçalho de resposta de localização inclui o identificador exclusivo para o objeto.
202	Aceito	Foi iniciado um trabalho em segundo plano de longa execução para executar a solicitação, mas a operação ainda não foi concluída.
400	Pedido incorreto	A entrada de solicitação não é reconhecida ou é inadequada.
403	Proibido	O acesso é negado devido a um erro de autorização.



Código	Significado	Descrição
404	Não encontrado	O recurso referido na solicitação não existe.
405	Método não permitido	O verbo HTTP na solicitação não é suportado para o recurso.
409	Conflito	Uma tentativa de criar um objeto falhou porque o objeto já existe.
500	Erro interno	Ocorreu um erro interno geral no servidor.
501	Não implementado	O URI é conhecido, mas não é capaz de executar a solicitação.

## Cabeçalhos de resposta

Vários cabeçalhos estão incluídos na resposta HTTP gerada pelo servidor de implantação, incluindo:

- Cada solicitação de API bem-sucedida é atribuída a um identificador de solicitação exclusivo.
- Localização quando um objeto é criado, o cabeçalho do local inclui o URL completo para o novo objeto, incluindo o identificador de objeto exclusivo.

## Corpo de resposta

O conteúdo da resposta associada a uma solicitação de API difere com base no objeto, no tipo de processamento e no sucesso ou falha da solicitação. O corpo de resposta é renderizado em JSON.

- Um único objeto pode ser retornado com um conjunto de campos com base na solicitação. Por exemplo, você pode usar GET para recuperar propriedades selecionadas de um cluster usando o identificador exclusivo.
- Vários objetos de uma coleção de recursos podem ser retornados. Em todos os casos, há um formato consistente usado, com `num_records` a indicação do número de Registros e Registros contendo um array das instâncias do objeto. Por exemplo, você pode recuperar todos os nós definidos em um cluster específico.
- Objeto de tarefa se uma chamada de API for processada de forma assíncrona, um objeto Job será retornado que ancora a tarefa em segundo plano. Por exemplo, a solicitação POST usada para implantar um cluster é processada de forma assíncrona e retorna um objeto Job.
- Se ocorrer um erro, um objeto de erro é sempre retornado. Por exemplo, você receberá um erro ao tentar criar um cluster com um nome que já existe.
- Vazio em certos casos, nenhum dado é retornado e o corpo de resposta está vazio. Por exemplo, o corpo da resposta está vazio depois de usar DELETE para excluir um host existente.

# Processamento assíncrono usando o objeto de tarefa

Algumas das chamadas da API Deploy, particularmente aquelas que criam ou modificam um recurso, podem levar mais tempo para serem concluídas do que outras chamadas. O ONTAP Select Deploy processa essas solicitações de longa duração de forma assíncrona.

## Solicitações assíncronas descritas usando o objeto Job

Depois de fazer uma chamada de API que é executada de forma assíncrona, o código de resposta HTTP 202 indica que a solicitação foi validada e aceita com sucesso, mas ainda não foi concluída. A solicitação é

processada como uma tarefa em segundo plano que continua a ser executada após a resposta HTTP inicial ao cliente. A resposta inclui o objeto Job ancorando a solicitação, incluindo seu identificador exclusivo.



Você deve consultar a página de documentação on-line do ONTAP Select Deploy para determinar quais chamadas de API operam assincronamente.

## Consulte o objeto Job associado a uma solicitação de API

O objeto Job retornado na resposta HTTP contém várias propriedades. Você pode consultar a propriedade State para determinar se a solicitação foi concluída com sucesso. Um objeto Job pode estar num dos seguintes estados:

- Em fila de espera
- Em execução
- Sucesso
- Falha

Há duas técnicas que você pode usar ao fazer polling de um objeto Job para detectar um estado terminal para a tarefa, seja com sucesso ou falha:

- O estado da tarefa atual da solicitação de polling padrão é retornado imediatamente
- O estado da tarefa de solicitação de polling longa é retornado somente quando ocorre uma das seguintes situações:
  - O estado mudou mais recentemente do que o valor de data-hora fornecido na solicitação de enquete
  - O valor de tempo limite expirou (1 a 120 segundos)

Polling padrão e polling longo usam a mesma chamada de API para consultar um objeto Job. No entanto, uma solicitação de polling longa inclui dois parâmetros de consulta: `poll_timeout` e `last_modified`.



Você deve sempre usar polling longo para reduzir a carga de trabalho na máquina virtual de implantação.

## Procedimento geral para emitir uma solicitação assíncrona

Você pode usar o seguinte procedimento de alto nível para concluir uma chamada assíncrona de API:

1. Emita a chamada assíncrona da API.
2. Receber uma resposta HTTP 202 indicando aceitação bem-sucedida da solicitação.
3. Extraia o identificador do objeto Job do corpo de resposta.
4. Dentro de um loop, execute o seguinte em cada ciclo:
  - a. Obtenha o estado atual do trabalho com uma solicitação de poll longa
  - b. Se o trabalho estiver em um estado não terminal (em fila de espera, em execução), execute o loop novamente.
5. Pare quando o trabalho atingir um estado terminal (sucesso, falha).

## **Informações sobre direitos autorais**

Copyright © 2026 NetApp, Inc. Todos os direitos reservados. Impresso nos EUA. Nenhuma parte deste documento protegida por direitos autorais pode ser reproduzida de qualquer forma ou por qualquer meio — gráfico, eletrônico ou mecânico, incluindo fotocópia, gravação, gravação em fita ou storage em um sistema de recuperação eletrônica — sem permissão prévia, por escrito, do proprietário dos direitos autorais.

O software derivado do material da NetApp protegido por direitos autorais está sujeito à seguinte licença e isenção de responsabilidade:

ESTE SOFTWARE É FORNECIDO PELA NETAPP "NO PRESENTE ESTADO" E SEM QUAISQUER GARANTIAS EXPRESSAS OU IMPLÍCITAS, INCLUINDO, SEM LIMITAÇÕES, GARANTIAS IMPLÍCITAS DE COMERCIALIZAÇÃO E ADEQUAÇÃO A UM DETERMINADO PROPÓSITO, CONFORME A ISENÇÃO DE RESPONSABILIDADE DESTES DOCUMENTOS. EM HIPÓTESE ALGUMA A NETAPP SERÁ RESPONSÁVEL POR QUALQUER DANO DIRETO, INDIRETO, INCIDENTAL, ESPECIAL, EXEMPLAR OU CONSEQUENCIAL (INCLUINDO, SEM LIMITAÇÕES, AQUISIÇÃO DE PRODUTOS OU SERVIÇOS SOBRESSAIENTES; PERDA DE USO, DADOS OU LUCROS; OU INTERRUPÇÃO DOS NEGÓCIOS), INDEPENDENTEMENTE DA CAUSA E DO PRINCÍPIO DE RESPONSABILIDADE, SEJA EM CONTRATO, POR RESPONSABILIDADE OBJETIVA OU PREJUÍZO (INCLUINDO NEGLIGÊNCIA OU DE OUTRO MODO), RESULTANTE DO USO DESTES DOCUMENTOS, MESMO SE ADVERTIDA DA RESPONSABILIDADE DE TAL DANO.

A NetApp reserva-se o direito de alterar quaisquer produtos descritos neste documento, a qualquer momento e sem aviso. A NetApp não assume nenhuma responsabilidade nem obrigação decorrentes do uso dos produtos descritos neste documento, exceto conforme expressamente acordado por escrito pela NetApp. O uso ou a compra deste produto não representam uma licença sob quaisquer direitos de patente, direitos de marca comercial ou quaisquer outros direitos de propriedade intelectual da NetApp.

O produto descrito neste manual pode estar protegido por uma ou mais patentes dos EUA, patentes estrangeiras ou pedidos pendentes.

LEGENDA DE DIREITOS LIMITADOS: o uso, a duplicação ou a divulgação pelo governo estão sujeitos a restrições conforme estabelecido no subparágrafo (b)(3) dos Direitos em Dados Técnicos - Itens Não Comerciais no DFARS 252.227-7013 (fevereiro de 2014) e no FAR 52.227- 19 (dezembro de 2007).

Os dados aqui contidos pertencem a um produto comercial e/ou serviço comercial (conforme definido no FAR 2.101) e são de propriedade da NetApp, Inc. Todos os dados técnicos e software de computador da NetApp fornecidos sob este Contrato são de natureza comercial e desenvolvidos exclusivamente com despesas privadas. O Governo dos EUA tem uma licença mundial limitada, irrevogável, não exclusiva, intransferível e não sublicenciável para usar os Dados que estão relacionados apenas com o suporte e para cumprir os contratos governamentais desse país que determinam o fornecimento de tais Dados. Salvo disposição em contrário no presente documento, não é permitido usar, divulgar, reproduzir, modificar, executar ou exibir os dados sem a aprovação prévia por escrito da NetApp, Inc. Os direitos de licença pertencentes ao governo dos Estados Unidos para o Departamento de Defesa estão limitados aos direitos identificados na cláusula 252.227-7015(b) (fevereiro de 2014) do DFARS.

## **Informações sobre marcas comerciais**

NETAPP, o logotipo NETAPP e as marcas listadas em <http://www.netapp.com/TM> são marcas comerciais da NetApp, Inc. Outros nomes de produtos e empresas podem ser marcas comerciais de seus respectivos proprietários.