swagger-api.json

Explore

Note: This section applies to v2 Clusters.

Cisco Container Platform Control Plane API Documentation

```
[ Base URL: https://Cisco Container Platform Control Plane IP/2/ ]
```

swagger-api.json

Schemes

HTTP

/v3 CCP v3 API

DELETE `/v3/{resource}` forwards v3 API requests to the v3 API service

GET `/v3/{resource}` forwards v3 API requests to the v3 API service

HEAD `/v3/{resource}` forwards v3 API requests to the v3 API service

PATCH `/v3/{resource}` forwards v3 API requests to the v3 API service

POST `/v3/{resource}` forwards v3 API requests to the v3 API service

PUT `/v3/{resource}` forwards v3 API requests to the v3 API service

2/aci_api accessing ACI api

POST `/2/aci_api/login` ACI login

2/aci_profiles List of ACI profile endpoints

GET `/2/aci_profiles` Get all ACI profiles

POST `/2/aci_profiles` Create an ACI profile with the given configuration

GET `/2/aci_profiles/{aciProfileName}` Get an ACI profile by name

DELETE `/2/aci_profiles/{aciProfileUUID}` Delete an ACI profile

PATCH `/2/aci_profiles/{aciProfileUUID}` Update an ACI profile

2/clusters List of cluster endpoints

GET `/2/clusters` Get all clusters

POST `/2/clusters` Create a cluster with the given specification

GET `/2/clusters/{clusterID}/authz` List authorizations for a cluster

POST `/2/clusters/{clusterID}/authz` Add authorization for a cluster

DELETE `/2/clusters/{clusterID}/authz/{authID}` Delete authorization for a cluster

GET `/2/clusters/{clusterName}` Get a cluster by name

DELETE `/2/clusters/{clusterUUID}` Delete a cluster

PATCH `/2/clusters/{clusterUUID}` Patch a cluster

PUT `/2/clusters/{clusterUUID}` Update a cluster

GET `/2/clusters/{clusterUUID}/dashboard` Get dashboard

GET `/2/clusters/{clusterUUID}/env` Get cluster environment

GET `/2/clusters/{clusterUUID}/helmcharts` Get HelmCharts object for a given cluster

POST `/2/clusters/{clusterUUID}/helmcharts` Create a helmChart for cluster with the given specification

DELETE `/2/clusters/{clusterUUID}/helmcharts/{HelmChartUUID}` Delete helm chart for cluster

POST `/2/clusters/{clusterUUID}/nodepools` Create a node pool for a cluster

DELETE `/2/clusters/{clusterUUID}/nodepools/{nodePoolID}` Delete a node pool from a cluster

PATCH `/2/clusters/{clusterUUID}/nodepools/{nodePoolID}` Update a node pool in a cluster

PATCH `/2/clusters/{clusterUUID}/upgrade` Upgrade a cluster

2/keyvalues List of endpoints for key values

GET `/2/keyvalues/{key}`

POST `/2/keyvalues/{key}`

2/ldap List of ldap endpoints

GET `/2/ldap/groups` Get CX LDAP Groups

POST `/2/ldap/groups` Create CX LDAP Group

PUT `/2/ldap/groups` Update a CX LDAP Group.

GET `/2/ldap/groups/authz` Get CX the cluster authorizations for a CX LDAP group

DELETE `/2/ldap/groups/{ldapDN}` Delete CX LDAP Group specified by LDAP DN

GET `/2/ldap/setup` Get LDAP parameters

PUT `/2/ldap/setup` Setup/update LDAP parameters

2/license List of licensing endpoints

DELETE `/2/license/{resource}` Refer to the smart licensing documentation

GET `/2/license/{resource}` Refer to the smart licensing documentation

DELETE `/2/license/{resource}/{agentID}` Refer to the smart licensing documentation

GET `/2/license/{resource}/{agentID}` Refer to the smart licensing documentation

POST `/2/license/{resource}/{agentID}` Refer to the smart licensing documentation

2/localusers

GET `/2/localusers` Get CX local users

POST `/2/localusers` Create CX local user

DELETE `/2/localusers/{username}` Delete a local user

PATCH `/2/localusers/{username}` Update a local user. Can provide either or both parameters.

PATCH `/2/localusers/{username}/password` Update

2/providerclientconfigs List of provider client config endpoints

GET `/2/providerclientconfigs` Get provider client configuration list

POST `/2/providerclientconfigs` Add provider client configuration

DELETE `/2/providerclientconfigs/{clientconfigUUID}` Delete provider client configuration

GET `/2/providerclientconfigs/{clientconfigUUID}` Get provider client configuration

PATCH `/2/providerclientconfigs/{clientconfigUUID}` Update provider client configuration

GET `/2/providerclientconfigs/{clientconfigUUID}/clusters` Get list of clusters who are using providerclientconfig

GET `/2/providerclientconfigs/{clientconfigUUID}/vsphere/datacenter` Gets the list of vSphere Data Centers.

GET `/2/providerclientconfigs/{clientconfigUUID}/vsphere/datacenter/{datacenterName}/cluster` Gets the list of vSphere Clusters in a datacenter.

GET `/2/providerclientconfigs/{clientconfigUUID}/vsphere/datacenter/{datacenterName}/cluster/{clusterName}/gpu` Gets the list of VSphere GPUs.

GET `/2/providerclientconfigs/{clientconfigUUID}/vsphere/datacenter/{datacenterName}/cluster/{clusterName}/pool` Gets the list of vSphere Pools.

GET `/2/providerclientconfigs/{clientconfigUUID}/vsphere/datacenter/{datacenterName}/datastore` Gets the list of vSphere Datastores.

GET `/2/providerclientconfigs/{clientconfigUUID}/vsphere/datacenter/{datacenterName}/network` Gets the list of vSphere Networks.

GET `/2/providerclientconfigs/{clientconfigUUID}/vsphere/datacenter/{datacenterName}/vm` Gets the list of vSphere Virtual Machines.

2/rbac

GET `/2/rbac` get the role of the current user

2/system List of system endpoints

GET `/2/system/CorcHealth` Get corc health

GET `/2/system/health` Returns the health of the system

GET `/2/system/livenessHealth` Returns a string representing the health of the system

POST `/2/system/login` Management server login

Models

```
api.ACILoginReply {
    token*                  string
}
```

```
api.ACILoginRequest {
    apic_ips*               string
    apic_password*          string
    apic_username*          string
}
```

```
api.AddAuthorization {
    Local*                  boolean
    Name*                   string
}
```

```
api.AddAuthorizationReply {
    AuthID*                 string
    Local*                  boolean
    Name*                   string
}
```

```
api.CorcHealthReply {
}
```

```
api.CorcHealthRequest {
}
```

```
api.CreateLocalUserRequest    {
    Disable*              boolean
    FirstName*            string
    LastName*             string
    Password*             string
    Role*                 string
    Token*                string
    UserName*             string
}
```

```
api.CreateLocalUserResponse    {
}
```

```
api.CreateNodePoolReply    {
    NodePool*             api.CreateNodePoolReply.NodePool    {...}
}
```

```
api.CreateNodePoolReply.NodePool    {
}
```

```
api.DeleteNodePoolReply    {
}
```

```
api.GetVSphereClustersReply    {
    Clusters*                 [...]
}
```

```
api.GetVSphereDatacentersReply    {
    Datacenters*              [...]
}
```

```
api.GetVSphereDatastoresReply    {
    Datastores*               [...]
}
```

```
api.GetVSphereGpusReply {
    gpus*                         [...]
}
```

```
api.GetVSphereNetworksReply {
    Networks*                     [...]
}
```

```
api.GetVSpherePoolsReply {
    Pools*                        [...]
}
```

```
api.GetVSphereVMsReply {
    VMs*                          [...]
}
```

```
api.GpuHostIndex {
    gpu_type*                 string
    hosts*                        [...]
}
```

```
api.HostGpuCount {
    count*                    integer($int32)
    hostname*                 string
}
```

```
api.LdapGroup {
    LdapDN*                   string
    Role*                     string
}
```

api.NodePoolRequest {

```
    gpus*
                            [...]
    labels*                 string
    memory*                 integer($int64)
    name*                   string
    node_ip_pool_uuid*      string
    size*                   integer($int32)
    taints*                 string
    template*               string
    vcpus*                  integer($int32)
}
```

api.ResizeNodePoolRequest {

```
    size*                   integer($int32)
}
```

api.UpdateLocalUserPasswordRequest {

```
    logged_in_user_password* string
    new_password*            string
}
```

api.UpdateLocalUserRequest {

```
    Disable*                boolean
    FirstName*              string
    LastName*               string
    Role*                   string
}
```

ipam.IPInfo {

```
    gateway*                string
    id*                     integer
    ip*                     string
    mtu*                    integer($int32)
    nameservers*
                            [...]
    netmask*                string
    subnet                  string
    uuid*                   string
}
```

ipam.LoadBalancerIPInfo {

```
    IPInfo*
                            ipam.IPInfo {...}
    never_release*          boolean
}
```

```
ipam.NodeIPInfo    {
    IPInfo*                  ipam.IPInfo    {...}
    if_name*                 string
    type*                        {...}
}
```

```
main.GetRoleResonse    {
    role*                    string
}
```

```
types.ACIProfile    {
    aaep_name*                     string
    aci_allocator                  types.ACIProfileAllocatorConfig    {...}
    aci_infra_vlan_id*             integer
    aci_tenant*                    string
    aci_vmm_domain_name*           string
    apic_hosts*                    string
    apic_password*                 string
    apic_username*                 string
    control_plane_contract_name*   string
    l3_outside_network_name*       string
    l3_outside_policy_name*        string
    name*                          string
    nameservers*
                                       [...]
    uuid*                          string
    vrf_name*                      string
}
```

```
types.ACIProfileAllocatorConfig    {
    multicast_range*         string
    node_vlan_end*           integer
    node_vlan_start*         integer
    pod_subnet_start*        string
    service_subnet_start*    string
}
```

```
types.Cluster    {
    Infra*
                                  types.Cluster.Infra    {...}
    aci_profile_uuid*             string
    aws_iam_enabled*              boolean
    aws_iam_role_arn*             string
    ccp_private_ssh_key*          string
    ccp_public_ssh_key*           string
    cluster_dashboard_url*        string
    cluster_env_url*              string
    deployer*
                                  types.Kubeadm    {...}
    description*                  string
    etcd_encrypted*               boolean
    harbor_admin_server_password* string
    harbor_registry_size*         string
    helm_charts*
                                      [...]
    ingress_vip_pool_id*          string
    ingress_vips*
                                      [...]
    is_adopt*                     boolean
    is_control_cluster*           boolean
    is_harbor_enabled*            boolean
    is_istio_enabled*             boolean
    kubernetes_version*           string
    labels*
                                      [...]
    load_balancer_ip_info_list*
                                      [...]
    load_balancer_ip_num*         integer($int32)
    master_mac_addresses*
                                      [...]
    master_node_pool
                                  types.Cluster.master_node_pool    {...}
    master_vip*                   string
    master_vip_addr_id*           string
    masters*                      integer($int32)
    name*                         string
    network_plugin*
                                  types.NetworkPluginProfile    {...}
    node_ip_pool_uuid             string
    node_pools*
                                      [...]
    nodes*
                                      [...]
    ntp_pools*
                                      [...]
    ntp_servers*
                                      [...]
    provider_client_config_uuid*  string
    registries_insecure*
                                      [...]
    registries_root_ca*
                                      [...]
    registries_self_signed*
                                      [...]
    secure_multitenancy_enabled*  boolean
    ssh_key*                      string
    ssh_user*                     string
    state*                        string
    storage_class*                string
    template*                     string
    tsig_key*                     string
    type*
                                      {...}
    uuid*                         string
    worker_node_pool
                                  types.Cluster.worker_node_pool    {...}
    workers*                      integer($int32)
}
```

```
types.Cluster.Infra    {
}
```

```
types.Cluster.master_node_pool    {
}
```

```
types.Cluster.node_pools    {
}
```

```
types.Cluster.worker_node_pool    {
}
```

```
types.GpuTypeCount    {
    count*                 integer($int32)
    gpu_type*              string
}
```

```
types.HelmChart    {
    chart_url*             string
    cluster_UUID*          string
    helmchart_uuid*        string
    name*                  string
    options*               string
}
```

```
types.K8SNodeStatus    {
    LastTransitionTime*    string
    NodeCondition*         string
    NodeName*              string
    NodeStatus*            string
}
```

```
types.K8SPodStatus    {
    LastTransitionTime*    string
    PodCondition*          string
    PodName*               string
    PodStatus*             string
}
```

```
types.Kubeadm    {
    provider*           types.VsphereCloudProvider    {...}
    provider_type*      string
}
```

```
types.Label    {
    key*                string
    value*              string
}
```

```
types.LdapSetup    {
    BaseDN*                    string
    InsecureSkipVerify*        boolean
    Port*                      integer
    Server*                    string
    ServiceAccountDN*          string
    ServiceAccountPassword*    string
    StartTLS*                  boolean
}
```

```
types.LoginStatus    {
    from_host*          string
    last_fail*          string($date-time)
    last_success*       string($date-time)
    login_id*           string
    proto*              string
    status*             string
    to_host*            string
    total_fail*         integer($int32)
}
```

```
types.NetworkPluginProfile    {
    details*            string
    name*               string
    status*             string
}
```

```
types.Node    {
    cloud_init_data*       string
    error_log*             string
    ip_info*
                               [...]
    is_master*             boolean
    kubernetes_version*    string
    mac_addresses*
                               [...]
    name*                  string
    node_pool_id*          integer
    node_pool_type*        string
    private_ip*            string
    public_ip*             string
    state*                 string
    template*              string
    uuid*                  string
}
```

```
types.ProviderClientConfig    {
    config*
                           types.ProviderClientConfig.config    {...}
    name*                  string
    type*
                               {...}
    uuid*                  string
}
```

```
types.ProviderClientConfig.config    {
}
```

```
types.SystemHealth    {
    CurrentNodes*          integer($int32)
    ExpectedNodes*         integer($int32)
    NodesStatus*
                               [...]
    PodStatusList*
                               [...]
    TotalSystemHealth*     string
}
```

```
types.VsphereClientConfig    {
    ip*                    string
    password               string
    port*                  integer
    username*              string
}
```

```
types.VsphereCloudProvider    {
    client_config;omitempty*    types.VsphereClientConfig    {...}
    vsphere_client_config_uuid*  string
    vsphere_datacenter*          string
    vsphere_datastore*           string
    vsphere_scsi_controller_type* string
    vsphere_working_dir*         string
}
```

ERROR {...}