CLOUD DEVELOPER GUIDE: GETTING STARTED WITH VCLOUD AIR

Discover how vCloud Air can improve your agility and performance with these code samples and common use cases. We've provided APIs and automation tools to make it easier for you to access and automate your cloud environment using your preferred language bindings and third-party solutions.

Programmatic Foundations of vCloud Air

vCloud Air is a public cloud service built on VMware's industry-leading vSphere technology. Designed to enable organizations to quickly and securely take advantage of the benefits of the cloud, the vCloud Air architecture is both modular and compatible to our wide network of service provider partners. Familiarizing yourself with the underlying vCloud Air architecture will provide you with a deeper understanding of how to consume vCloud Air via APIs, or through higher level constructs such as CLIs and automation tools, and how to best design your solutions.

The vCloud Air API Platforms

vCloud Air is a set of modular services that are tied to a platform.The platform governs the various services and provides common features such as identity and access management, service discovery, and billing and metering. The service offerings available on the platform include compute, storage, database as a service, and more.

We provide two distinct platforms:

- vCloud Hybrid Services (vCHS), which focuses on compute services such as vCloud Air Dedicated Cloud
- vCloud Air (vCA) , which is service agnostic and supports current forthcoming vCloud Air service offerings, such as vCloud Air Object Storage and vCloud Air SQL, along with Virtual Private Cloud OnDemand

When consuming vCloud Air through API interfaces, it is important to understand the differences between the platform you are accessing, and your purchase model—subscription versus pay-as-you-go.

Logging in to the API Platforms

VMware vCloud Air is designed so that there is an API hand-off between the platform and the individual services. The nature of this hand-off depends on the platform that controls the service and the service itself.

For example, with the compute service controlled by the vCHS platform, the service discovery process will provide the compute end point as well as the credentials to consume that end-point.

With the compute service controlled by the vCA platform, the service discovery process will provide the compute end point against which you will need to login explicitly.

With the Object Storage services (both powered by EMC and by Google Cloud Platform) the process involves grabbing the keys from the vCloud Air UI and then using them against the proper end-points advertised by the platform service discovery process.

Code Reuse

The modularity of the vCloud Air architecture enables the ability to reuse the code fragments in related services delivered by our partners.

Since vCloud Air uses vCloud Director to deliver the compute service, there is a common foundation for vCloud APIs that is compatible with our vCloud Air Network service providers. There is a process to discover the vCloud tenant end-points that is specific to vCloud Air, but the operations inside a virtual data center for creating, editing, and deleting virtual machines and virtual networks is the same. This means that you can re-use your code (and existing integrations) that exist today for this community, including the incredible amount of public and private clouds that are built on vCloud Director.

Similarly, vCloud Air tenants that want to leverage vCloud Air Object Storage powered by Google Cloud Platform can re-use any piece of code that is compatible with this Google Cloud Platform service (including Google's own gsutil CLI). The vCloud Air platform only manages the access keys and programmatic access to create, edit, and delete buckets and objects are the same.

vCloud Air Compute Service Overview

Compute services are the foundation for cloud computing and typically the entry point into vCloud Air. This section describes the procedures for accessing Virtual Private Cloud OnDemand and Virtual Private Cloud (subscription) with the vCA platform.

You can query the service controller of the vCA platform:

GET https://vca.vmware.com/api/sc/instances

This will return a list of service instances that you have available in your tenant. Some of these instances will be compute instances if you have already deployed virtual machines in your tenant.

This is a sample response of a compute instance as represented in the service controller:


```
"name": "Virtual Private Cloud OnDemand",
"id": "bc129d20-770a-456f-b23a-4a4ac112aae7",
"description": "Create virtual machines, and easily scale up or down as your
needs change.",
"region": "us-california-1-3.vchs.vmware.com",
"instanceVersion": "1.0",
"planId": "region:us-california-1-3.vchs.vmware.com:planID:c65d5821-aa97-
4141-915a-7d7eab0a9d51",
"serviceGroupId": "49d03ec7-15c3-4f62-ac73-ea99d7ad0cc9",
"apiUrl": "https://us-california-1-
3.vchs.vmware.com/api/compute/api/org/bc129d20-770a-456f-b23a-4a4ac112aae7",
"dashboardUrl": "https://us-california-1-
3.vchs.vmware.com/api/compute/compute/ui/index.html?orgName=92402aa7-5176-
4a29-956a-5be4d0b401fb&serviceInstanceId=bc129d20-770a-456f-b23a-
4a4ac112aae7&servicePlan=c65d5821-aa97-4141-915a-7d7eab0a9d51",
"instanceAttributes": "{\"orgName\":\"92402aa7-5176-4a29-956a-
5be4d0b401fb\",\"sessionUri\":\"https://us-california-1-
3.vchs.vmware.com/api/compute/api/sessions\",\"apiVersionUri\":\"https://us-
california-1-3.vchs.vmware.com/api/compute/api/versions\"}"
```


Note: even on the new vCA platform some of the URLs have *vchs* in them. Do not get confused by that.

These are the relevant fields you should focus on:

- **id**: this represents the unique ID of the instance. It also represents the vCloud Director Org ID (as found in the api Url entry).
- **region**: this tells you the region where this instance is deployed
- **service GroupId**: this is the cost center tied to the instance (more on this later)
- **api Url**: this is the actual compute service URL for the hand-off
- **instance Attributes**: this includes the parameters to login into the instance

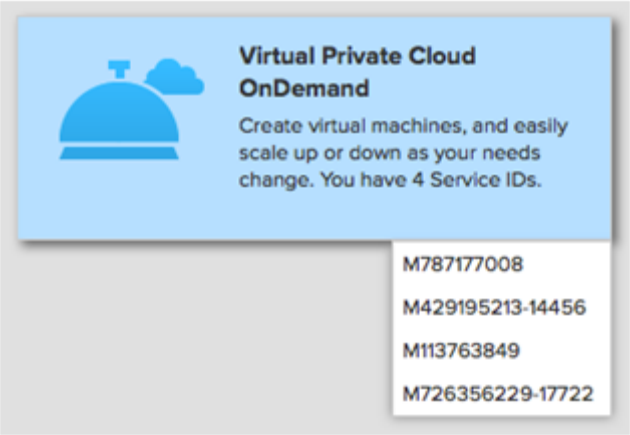
You cannot use the vCA platform token you have received at login and pass it to vCloud Director pointing to the api Url today. Instead, you will need to use the instance Attributes coordinate to login into the Org. There you have everything you need: sessionUri (the login end-point) and orgName (the Organization name).

From this point on you can treat your instance as a standard vCloud Director Organization.

Multiple Service IDs

Note that there is typically only one vCloud Director Org per region, per serviceGroupId. However, it is possible to have more than one service GroupId provisioned which means you may have more than one vCloud Director Org per region and multiple Service IDs.

The easiest way to find out if you have more than one service GroupId is to hover your mouse on Virtual Private Cloud OnDemand in the UI. If you see something similar to the picture below it means you have more than one service GroupId:



You can look up the relationship between the serviceGroupId and the Service IDs by querying the vCA platform using GET https://vca.vmware.com/api/billing/service-group/

Resources

Common Use Cases

Deploying an Internet-connected virtual machine from the public catalog

- **vCA-CLI:** https://github.com/vmware/vca-codesamples/blob/master/vca-cli/CreateInternetVM.sh
- **Python-SDK:** https://github.com/vmware/vca-codesamples/blob/master/pyvcloud/create_internet_vm.py
- **Ansible:** https://github.com/vmware/vca-codesamples/blob/master/ansible/CreateInternetVM.yml

Uploading a vSphere local template to a private catalog

- **PowerCLI:** https://github.com/vmware/vca-codesamples/blob/master/powercli/UploadOVF.ps1

Deploying a batch of 20 identical virtual machines

- **PowerCLI:** https://github.com/vmware/vca-codesamples/blob/master/powercli/Deploy20VMs.ps1
- **vCA-CLI:** https://github.com/vmware/vca-codesamples/blob/master/vca-cli/Deploy20VMs.sh
- **Python-SDK:** https://github.com/vmware/vca-codesamples/blob/master/pyvcloud/deploy_20_vms.py
- **Ansible:** https://github.com/vmware/vca-codesamples/blob/master/ansible/Deploy20VMs.yml

Listing all virtual machines running in a virtual data center

- **vCA-CLI:** https://github.com/vmware/vca-codesamples/blob/master/vca-cli/ListVMs.sh
- **PowerCLI:** https://github.com/vmware/vca-codesamples/blob/master/powercli/ListVMsinVDC.ps1
- **Python-SDK:** https://github.com/vmware/vca-codesamples/blob/master/pyvcloud/list_vms_in_vdc.py

End-to-End Scenarios

Simple application deployment from a code source (GitHub)

Melanie is a developer working on a new software application. The target customers of this application typically host this application in a VMware-virtualized environment. As she is in the early stages of development, she is looking for a way to automatically deploy her builds into a public cloud environment for testing. As she does this many times a day, she'd like this to be automated.

Her code is stored in a GitHub repository and she wants to automate the process of:

1. Creating a virtual machine in vCloud Air
2. Creating proper firewall and NAT rules to make the virtual machine Internet-accessible
3. Installing the middleware required to support the application (nginx)
4. Installing the application from GitHub

The solution below uses an Ansible playbook that leverages the vCloud Air integration module and others that are available:
https://github.com/vmware/vca-codesamples/blob/master/simple-app-automation/simple-app-automation.yml

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com