VeloCloud Orchestrator Installation

Pre Requisites

Instance Requirements

VeloCloud recommends installation of the Orchestrator and Gateway applications as a virtual machine i.e. guest instance on an existing hypervisor.

NOTE: VCO will not start with less than 10GB of memory.

The VeloCloud Orchestrator needs the following minimal guest instance specifications:

- 8 Intel vCPU's at 2.5 Ghz or higher
- 16 GB of memory
- 2x 1TB SSD based persistent volumes (expandable through LVM if needed)
    - Required Minimum IOPS: 5,000 IOPS
- 1 Gbps NIC
- Ubuntu x64 server VM compatibility
- Single public IP address (Can be made available through NAT)

Upstream Firewall Configuration

Upstream firewall needs to be configured to allow inbound HTTP (TCP/80) as well as HTTPS (TCP/443). If a stateful firewall is in place, established connections that are outbound originated should also be allowed to facilitate upgrades and security updates.

External Services

The VeloCloud Orchestrator relies on several external services. Ensure licenses are available for each of the services before proceeding with an installation:

Google Maps

Google Maps is used for displaying edges and data centers on a map. No account has to be created with Google to utilize the functionality but internet access must be available to the VCO instance in order for the service to be available.

The service is limited to 25,000 map loads each day, for more than 90 consecutive days. VeloCloud does not anticipate exceeding these limits for nominal use of the VCO.

Twilio

Twilio is used for SMS based alerting to enterprise customers to notify them of Edge or link outage events. An account needs to be created and funded at http://www.twilio.com

The account can be provisioned in the VCO through the Operator Portal's System Properties page. The account will be provisioned through a system property as detailed later in the guide

MaxMind

MaxMind is geolocations service. It is used to automatically detect Edge and Gateway locations and ISP names based on IP address. If this service is disabled then geolocation information will need to be updated manually. The account can be provisioned in the VCO through the Operator Portal's System Properties page. The properties are called:

Installation

Cloud-init Preparation

Cloud-init is a linux package responsible for handling early initialization of instances. If available in the distributions, it allows for configuration of many common parameters of the instance directly after installation. This creates a fully functional instances that is configured based on a series of inputs.

Cloud-init's behavior can be configured via user-data. User-data can be given by the user at instance launch time. This is typically done by attaching a secondary disk in ISO format that cloud-init will look for at first boot time. This disk contains all early configuration data that will be applied at that time.

The VeloCloud Orchestrator supports cloud-init and all essential configurations can be packaged in through an ISO image.

NOTE: The cloud-init file is a yaml file. So both meta-data and user-data need to be valid yaml files. To validate a yaml file use: http://www.yamllint.com/

Create Cloud-init Meta-data File

The final installation configuration options are set with a pair of cloud-init configuration files. The first installation configuration file contains the metadata. Create this file with a text editor and call it meta-data. This file provides information that identifies the instance of VeloCloud Orchestrator being installed. The instance-id can be any identifying name and the local-hostname should be a host name that follows your site standards, for example:

instance-id: vco01

local-hostname:
vco-01

Additionally, you can specify network interface information (if the network is not configured via DHCP, for example):

```
instance-id: vco01
local-hostname: vco-01
network-interfaces: |
  auto eth0
  iface eth0 inet static
  address 10.0.1.2
  network 10.0.1.0
  netmask 255.255.255.0
  broadcast 10.0.1.255
  gateway 10.0.1.1
```

The second installation configuration option file is the user data file. This file provides information about users on the system. Create it with a text editor and call it user-data. This file will be used to enable access to the installation of VeloCloud Orchestrator. The following is an example of what the user-data file will look like:

```
#cloud-config
password: Velocloud123
chpasswd: {expire: False}
ssh_pwauth: True
ssh_authorized_keys:
  - ssh-rsa AAA...SDvz user1@yourdomain.com
  - ssh-rsa AAB...QTuo user2@yourdomain.com
vco:
  super_users:
    list: |
      user1@yourdomain.com:password1
    remove_default_users: True
  system_properties:
```

```
    list: |
      mail.smtp.port:34
      mail.smtp.host:smtp.yourdomain.com
      service.maxmind.enable:True
      service.maxmind.license:todo_license
      service.maxmind.userid:todo_user
      service.twilio.phoneNumber:222123123
      network.public.address:222123123
write_files:
  - path: /etc/nginx/velocloud/ssl/server.crt
    permissions: '0644'
    content: "-----BEGIN CERTIFICATE-----\nMI....ow==\n-----END CERTIFICATE-----\n"
  - path: /etc/nginx/velocloud/ssl/server.key
    permissions: '0600'
    content: "-----BEGIN RSA PRIVATE KEY-----\nMII...D/JQ==\n-----END RSA PRIVATE
KEY-----\n"
  - path: /etc/nginx/velocloud/ssl/velocloudCA.crt
```

This user-data file enables the default user, vcadmin, to login either with a password or with an SSH key. The use of both methods is possible, but not required. The password login is enabled by the password and chpasswd lines. The password contains the plain-text password for the vcadmin user. The chpasswd line turns off password expiration to prevent the first login from immediately prompting for a change of password. This is optional. If you set a password, it is recommended that you change it when you first log in because the password has been stored in a plain text file.

The ssh_pwauth line enables SSH login. The ssh_authorized_keys line begins a block of one or more authorized keys. Each public SSH key listed on the ssh-rsa lines will be added to the vcadmin ~/.ssh/authorized_keys file.

In this example, two keys are listed. For this example, the key has been truncated, in a real file the entire public key must be listed. Note that the ssh-rsa lines must be preceded by two spaces, followed by a hyphen, followed by another space.

vco section configured VeloCloud Orchestrator services.

super_users contains list of VeloCloud Super Operator accounts and corresponding passwords.

system_properties section allows to customize VeloCloud Orchestrator System Properties. See section titled System Properties for details regarding system properties configuration.

write_files section allows to replace files on the system. By default, VeloCloud Orchestrator web services are configured with self-signed SSL certificate. If you would like to provide different SSL certificate, the above example replaces server.crt and server.key files in /etc/nginx/velocloud/ssl/ folder with user-supplied files. Please know, that server.key must be unencrypted, otherwise the service will fail to start without key password.

Create ISO file

Once you have completed your files, they need to be packaged into an ISO image. This ISO image is used as a virtual configuration CD with the virtual machine. This ISO image, called vco01-cidata.iso, is created with the following command on Linux system:

```
genisoimage -output vco01-cidata.iso -volid cidata -joliet -rock user-data
meta-data
```

Transfer the newly created ISO image to the datastore on the host running VMware.

VMWare Installation

VMware vSphere provides a means of deploying and managing virtual machine resources. This section explains how to run VeloCloud Orchestrator using the VMware vSphere Client.

Deploy OVA Template

NOTE: This procedure assumes familiarity with VMware vSphere and is not written with reference to any specific version of VMware vSphere.

1. Log in to the vSphere Client.
2. Select **File > Deploy OVF Template**.
3. Respond to the prompts with information specific to your deployment.
4. **Source:** Type a URL or navigate to the OVA package location.
5. **OVF template details:** Verify that you pointed to the correct OVA template for this installation.
6. **Name and location:** Name of the virtual machine.
7. **Storage:** Select the location to store the virtual machine files.
8. **Provisioning:** Select the provisioning type. "thin" is recommended for database and binary log volumes.
9. **Network mapping:** Select the network for each virtual machine to use.

IMPORTANT: Uncheck **Power On After Deployment**. Selecting it will start the virtual machine and it should be started later after the cloud-init ISO has been attached.

10. Click **Finish**.

NOTE: Depending on your network speed, this deployment can take several minutes or more.

Attach ISO Image as a CD/DVD to Virtual Machine

1. Right-click, the newly-added VeloCloud Orchestrator VM and select **Edit Settings**.
2. From the **Virtual Machine Properties** window, select CD/DVD Drive.
3. Select the **Use an ISO image** option.
4. Browse to find the ISO image you created earlier (we called ours vco01-cidata.iso), select it. The ISO can be found in the datastore that you uploaded it to, in the folder that you created.
5. Select **Connect on Power On**.
6. Click **OK** to exit the **Properties** screen.

Run the VeloCloud Orchestrator Virtual Machine

1. To start up the VeloCloud Orchestrator virtual machine, click to highlight it, then select the **Power On** button.
2. Select the **Console** tab to watch as the virtual machine boots up.
3. If you configured VeloCloud Orchestrator as described here, you should be able to log into the virtual machine with the user name vcadmin and password that you defined when you created the cloud-init ISO.

KVM Installation

This section explains how to run VeloCloud Orchestrator using the libvirt. This deployment was tested in Ubuntu 14.04LTS.

Images

For KVM deployment, VeloCloud will provide the VCO in three qcow images.

- OS
- DATABASE
- LOGS

The images are thin provisioned on deployment.

Start by copying the images to the KVM server. In addition, you will need to copy the cloud-init iso build as described in the previous section.

XML Sample

```
<domain type='kvm' id='49'>
 <name>vco</name>
 <uuid>b0ff25bc-72b8-6ccb-e777-fdc0f4733e05</uuid>
 <memory unit='KiB'>12388608</memory>
 <currentMemory unit='KiB'>12388608</currentMemory>
 <vcpu>2</vcpu>
 <resource>
  <partition>/machine</partition>
 </resource>
 <os>
 <type>hvm</type>
 </os>
 <features>
  <acpi/>
  <apic/>
  <pae/>
 </features>
  <cpu mode='custom' match='exact'>
  <model fallback='allow'>SandyBridge</model>
  <vendor>Intel</vendor>
  <feature policy='require' name='vme'/>
  <feature policy='require' name='dtes64'/>
  <feature policy='require' name='invpcid'/>
  <feature policy='require' name='vmx'/>
```

<feature policy='require' name='erms'/>

<feature policy='require' name='xtpr'/>

<feature policy='require' name='smep'/>

<feature policy='require' name='pbe'/>

<feature policy='require' name='est'/>

<feature policy='require' name='monitor'/>

<feature policy='require' name='smx'/>

<feature policy='require' name='abm'/>

<feature policy='require' name='tm'/>

<feature policy='require' name='acpi'/>

<feature policy='require' name='fma'/>

<feature policy='require' name='osxsave'/>

<feature policy='require' name='ht'/>

<feature policy='require' name='dca'/>

<feature policy='require' name='pdcm'/>

<feature policy='require' name='pdpe1gb'/>

<feature policy='require' name='fsgsbase'/>

<feature policy='require' name='f16c'/>

<feature policy='require' name='ds'/>

<feature policy='require' name='tm2'/>

<feature policy='require' name='avx2'/>

<feature policy='require' name='ss'/>

<feature policy='require' name='bmi1'/>

<feature policy='require' name='bmi2'/>

<feature policy='require' name='pcid'/>

<feature policy='require' name='ds_cpl'/>

<feature policy='require' name='movbe'/>

```
    <feature policy='require' name='rdrand'/>
  </cpu>
<clock offset='utc'/>
  <on_poweroff>destroy</on_poweroff>
  <on_reboot>restart</on_reboot>
  <on_crash>restart</on_crash>
  <devices>
    <emulator>/usr/bin/kvm-spice</emulator>
    <disk type='file' device='disk'>
      <driver name='qemu' type='qcow2'/>
      <source file='/images/vco/vco-root.img'/>
      <target dev='hda' bus='ide'/>
      <alias name='ide0-0-0'/>
      <address type='drive' controller='0' bus='0' target='0' unit='0'/>
    </disk>i
    <disk type='file' device='disk'>
      <driver name='qemu' type='qcow2'/>
      <source file='/images/vco/vco-db.img'/>
      <target dev='hdb' bus='ide'/>
      <alias name='ide0-0-1'/>
      <address type='drive' controller='0' bus='0' target='0' unit='1'/>
    </disk>
    <disk type='file' device='disk'>
      <driver name='qemu' type='qcow2'/>
      <source file='/images/vco/vco-binlog.img'/>
      <target dev='hdc' bus='ide'/>
      <alias name='ide0-0-2'/>
```

```
    <address type='drive' controller='0' bus='1' target='0' unit='0'/>
   </disk>
   <disk type='file' device='cdrom'>
    <driver name='qemu' type='raw'/>
    <source file='/images/vco/seed.iso'/>
    <target dev='sdb' bus='sata'/>
    <readonly/>
    <alias name='sata1-0-0'/>
    <address type='drive' controller='1' bus='0' target='0' unit='0'/>
   </disk>
   <controller type='usb' index='0'>
    <alias name='usb0'/>
    <address type='pci' domain='0x0000' bus='0x00' slot='0x01'
function='0x2'/>
   </controller>
   <controller type='pci' index='0' model='pci-root'>
    <alias name='pci.0'/>
   </controller>
   <controller type='ide' index='0'>
    <alias name='ide0'/>
    <address type='pci' domain='0x0000' bus='0x00' slot='0x01'
function='0x1'/>
   </controller>
   <interface type='direct'>
    <source dev='eth0' mode='vepa'/>
   </interface>
   <serial type='pty'>
```

```
    <source path='/dev/pts/3'/>
    <target port='0'/>
    <alias name='serial0'/>
   </serial>
   <console type='pty' tty='/dev/pts/3'>
    <source path='/dev/pts/3'/>
    <target type='serial' port='0'/>
    <alias name='serial0'/>
   </console>
   <memballoon model='virtio'>
    <alias name='balloon0'/>
    <address type='pci' domain='0x0000' bus='0x00' slot='0x03'
function='0x0'/>
   </memballoon>
  </devices>
  <seclabel type='none' />
<!--  <seclabel type='dynamic' model='apparmor' relabel='yes'/> -->
</domain>
```

The number of CPU's for production should be 8, VCO can work well even with 2. It's important to set a CPUset that is outside of what other VM's are using. To see allocation of a VM you can do

Libvirt virsh vcpuinfo "VM"

```
<vcpu cpuset='2-4'>3</vcpu>
```

The "IT Operator Guide for VeloCloud Orchestrator" speciafies a long list of CPU requirements. We found it easier to just do host-model. The only disadvantage of host-model is when VM's required to be live motion.

```
<cpu mode='host-model'>
<model fallback='allow'/>
```

</cpu>

The “IT Operator Guide for VeloCloud Orchestrator” speciafies a long list of CPU requirements. We found it easier to just do host-model. The only disadvantage of host-model is when VM’s required to be live motion.

<interface type='network'>

<source network='passthrough'/>

<vlan><tag id='#interface1_vlan#'/></vlan>

<alias name='hostdev1'/>

<address type='pci' domain='0x0000' bus='0x00' slot='0x11' function='0x0'/>

</interface>

No special requirements for VCO on the Network side, Bridge/SR-IOV/ Direct can be used.

In Bold

The "**In Bold**" the sections you will need to edit from the XML.

Create VM

To create the VM in using the standard virsh commands.

virsh define vco.xml

virsh start vco.xml

AWS installation

Minimum Instance Requirements

See the first section of the VeloCloud Orchestrator Installation, titled“ Instance Requirements" and select an AWS instance type matching these requirements. Both CPU and Memory requirements must be satisfied. Example: use c4.2xlarge or larger; r4.2xlarge or larger

AMI Image

Request an AMI ID from VeloCloud. It will be shared with the customer account. Please have an Amazon AWS account ID ready when requesting AMI access.

Installation

1. Launch EC2 instance in AWS cloud. Example: http://docs.aws.amazon.com/efs/latest/ug/gs-step-one-create-ec2-resources.html
2. Configure security group to allow inbound HTTP (TCP/80) as well as HTTPS (TCP/443)
3. After the instance is launched, point the web browser to Operator login URL: https://<name>/operator

Initial Configuration Tasks

- Configure system properties
- Set up initial operator profile
- Set up operator accounts
- Create gateways
- Setup gateway pools
- Create customer account / partner account

Install SSL Certificate

1. Login into VCO console. If you configured the VeloCloud Orchestrator as described here, you should be able to log into the virtual machine with the user name vcadmin and password that you defined when you created the cloud-init ISO).
2. Generate VCO private key (do not encrypt the key - it must remain unencrypted on VCO system):

openssl genrsa -out server.key 2048

3. Generate certificate request. Customize “subject” according to your organization information.

openssl req -new -key server.key -out server.csr -subj "/C=US/ST=California/L=Mountain View/O=Velocloud Networks Inc./OU=Development/CN=vco.velocloud.net"

Description of Subject Fields:

- C - country
- ST - state
- L - locality (city)
- O - company
- OU - department (optional)
- CN - VCO fully qualified domain name

4. Send server.csr to Certificate Authority for signing. You should get back SSL certificate (server.crt). Ensure that it is in PEM format.
5. Install certificate (requires root access). VCO SSL certificates are located in /etc/nginx/velocloud/ssl/

cp server.key server.crt /etc/nginx/velocloud/ssl/

chmod 600 /etc/nginx/velocloud/ssl/server.key

6. Restart nginx.

Service nginx restart

Configure System Properties

System Properties provide a mechanism to control system wide behavior of the VeloCloud Orchestrator. System Properties can be initially set via cloud-init config file under the VCO section (see “Create cloud-init meta-data file” section above). The following properties need to be configured to ensure proper operation of the service:

System Name

Enter fully qualified VCO domain name in network.public.address system property.

Google Maps

Google Maps is used for displaying edges and data centers on a map. Maps may fail to display without a license key. The Orchestrator will continue to function properly, but browser maps will not be available in this case.

1. Login into https://console.developers.google.com
2. Create a new project if not already created.
3. Locate the button **Enable API**. Click under the **Google Maps APIs** and enable both **Google Maps JavaScript API** and **Google Maps Geolocation API**.
4. On the left side of screen click the link **Credentials**.
5. Under the **Credentials** page, click **Create Credentials** then select **API key**. Create API key.
6. Set service.client.googleMapsApi.key VCO system property to API key.
7. Set service.client.googleMapsApi.enable to “true.”

Twilio

Twilio is a messaging service that allows you to receive VCO alerts via SMS. It is optional. The account can be provisioned in the VCO through the Operator Portal's System Properties page. The properties are called:

- service.twilio.enable allow the service to be disabled in the event no internet access is available to the VCO
- service.twilio.accountSid
- service.twilio.authToken
- service.twilio.phoneNumber in (nnn)nnn-nnnn format

Obtain service at https://www.twilio.com

MaxMind

MaxMind is a geolocations service. It is used to automatically detect Edge and Gateway locations and ISP names based on an IP address. If this service is disabled, then geolocation information will need to be updated manually. The account can be provisioned in the VCO through the Operator Portal's System Properties page. The properties are called:

- service.maxmind.enable allows the service to be disabled in the event no internet access is available to the VCO
- service.maxmind.userid holds the user identification supplied by MaxMind during the account creation
- service.maxmind.license holds the license key supplied by maxmind

Obtain license at: https://www.maxmind.com/en/geoip2-precision-city-service

Email

Email services can be used for both sending the Edge activation messages as well as for alarms and notifications. It is not required, but it is strongly recommended to configure this as part of VCO operations. The following system properties are available to configure the external email service used by the Orchestrator:

- mail.smtp.auth.pass - SMTP user password.
- mail.smtp.auth.user - SMTP user for authentication.
- mail.smtp.host - relay server for email originated from the VCO.
- mail.smtp.port - SMTP port.
- mail.smtp.secureConnection - use SSL for SMTP traffic.

Upgrade

Upload the image to VeloCloud Orchestrator system, using, for example, scp command. Copy the image to the following location on the system: /var/lib/velocloud/software_update/vco_update.tar

Connect to VeloCloud Orchestrator console and run.

NOTE: If you configured VeloCloud Orchestrator as described here, you should be able to log into the virtual machine with the user name vcadmin and password that you defined when you created the cloud-init ISO.

sudo
/opt/vc/bin/vco_software_update