Service Operations & Maintenance Guide

Copyright

Trademarks

Software License Agreement

Disclaimer

Architecture

VeloCloud Orchestrator

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. Bare metal installation of the component is currently not recommended.

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)
    - Minimum 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 used for geolocating IP addresses of last mile links and data center tunnel connections

An account needs to be created and funded at http://www.maxmind.com. The VCO uses the ISP/City/Org Service and VeloCloud recommends to set the service up to auto-renew and auto-charge in order to avoid service interruptions that could extend to the VCO.

The account will be provisioned through a system property as detailed later in the guide.

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.

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 log in either with a password or with an SSH key. The use of both methods is possible but not required. 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. Please refer to <<<Velocloud Orchestrator Administration Guide>> for details on 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.

Log in to the vSphere Client.

Select **File > Deploy OVF Template**.

Respond to the prompts with information specific to your deployment.

Source: Type a URL or navigate to the OVA package location.

OVF template details: Verify that you pointed to the correct OVA template for this installation.

Name and location: Name of the virtual machine

Storage: Select the location to store the virtual machine files.

Provisioning: Select the provisioning type. "thin" is recommended for database and binary log volumes

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.

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

Right-click, the newly-added Velocloud Orchestrator VM and select **Edit Settings**.

From the Virtual Machine Properties window, select CD/DVD Drive.

Select the Use an ISO image option.

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.

Select “Connect on Power On.”

Click OK to exit the Properties screen.

Run the Velocloud Orchestrator virtual machine

To start up the Velocloud Orchestrator virtual machine, click to highlight it, then select the **Power On** button.

Select the **Console** tab to watch as the virtual machine boots up.

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>
```

In bold

In Bold the sections you will need to edit from the XML

Create VM

To create the VM in use the standard virsh commands.

virsh define vco.xml

virsh start vco.xml

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

Login into VCO console

Generate VCO private key (do not encrypt the key - it must remain unencrypted on VCO system):

```
openssl genrsa -out server.key 2048
```

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 - locallity (city)
- O - company
- OU - department (optional)
- CN - VCO fully qualified domain name

Send server.csr to Certificate Authority for signing. You should get back SSL certificate (server.crt). Ensure that it is in PEM format.

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
```

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 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. 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.

Login into https://console.developers.google.com

Create new project if not already created.

Locate button "Enable API", click and under "Google Maps APIs" enable "Google Maps JavaScript API" and "Google Maps Geolocation API"

On the left side of screen click on the link "Credentials".

Under Credentials page, click “Create Credentials” then select “API key”. Create API key.

Set service.client.googleMapsApi.key VCO system property to API key

Set service.client.googleMapsApi.enable to “true”

Twilio

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

The account can be provisioned in the VCO through the Operator Portal's System Properties page. The properties are called:

- service.maxmind.enable allow the service to be disabled in the event no internet access is available to the VCO
- service.maxmind.userid hold 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 Edge activation messages as well as for alarms and notifications. 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

Then, upload 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

```
sudo /opt/vc/bin/vco_software_update
```

Data Management

Expanding Disk Size (VMWare)

Database volume is LVM device and can be resized online provided underlying virtualization technology support online disk expansion.

Login into VCO system console

Identify physical disks backing the database volume

```
vgs -o +devices db_data
```

Example:

```
root@vco:~# vgs -o +devices db_data
\  VG      #PV #LV #SN Attr   VSize   VFree   Devices
  db_data   1   1   0 wz--n- 500.00g 125.00g /dev/sdb(0)
```

Identify physical disk attachment

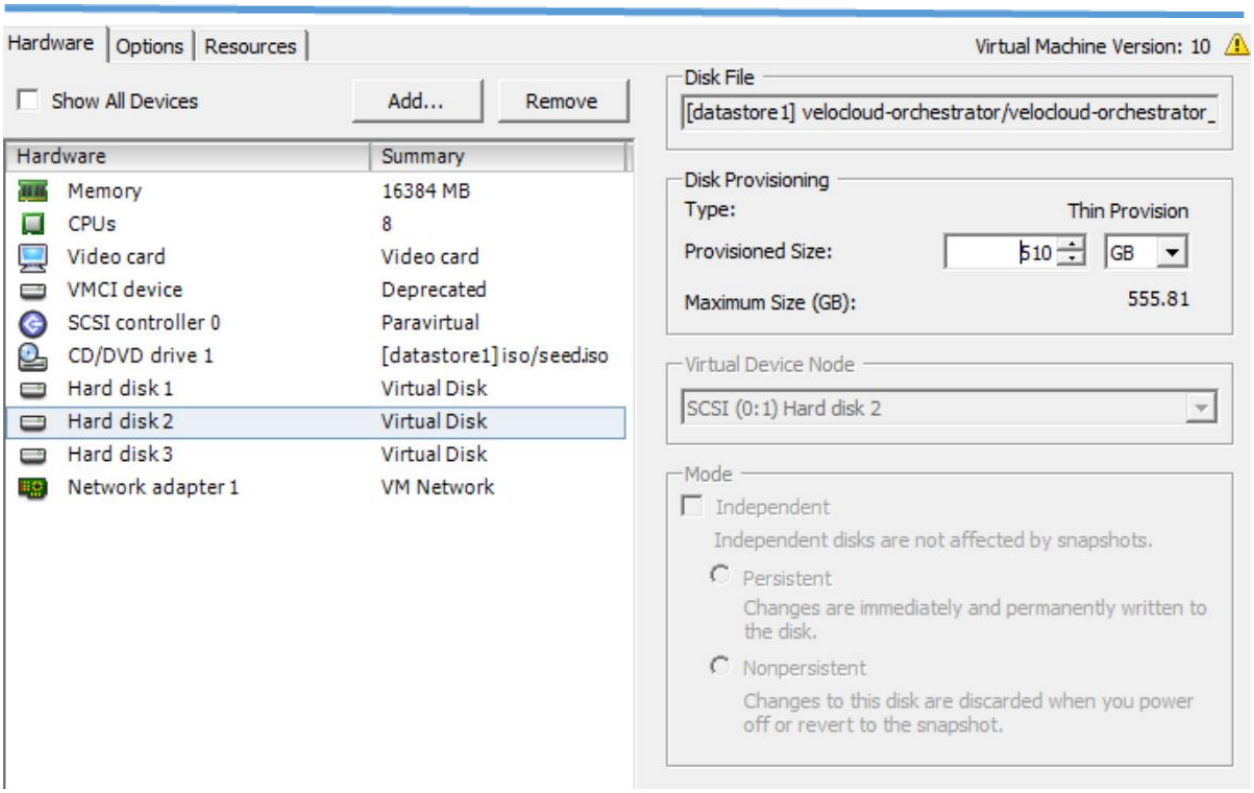
```
lshw -class volume
```

Example: /dev/sdb is attached to scsi@2:0.1.0 (Host: scsi2 Channel: 00 Id: 01 Lun: 00)

```
root@vco:~# lshw -class volume
  *-volume
       description: EXT4 volume
       vendor: Linux
       physical id: 1
       bus info: scsi@2:0.0.0,1
       logical name: /dev/sda1
       logical name: /
       version: 1.0
       serial: 9d212247-77c4-4f98-a5c2-7f8470fa2da8
       size: 10239MiB
       capacity: 10239MiB
       capabilities: primary bootable journaled extended_attributes
large_files huge_files dir_nlink recover extents ext4 ext2
initialized
       configuration: created=2016-02-22 20:49:38 filesystem=ext4
label=cloudimg-rootfs lastmountpoint=/ modified=2016-02-22 21:18:58
mount.fstype=ext4 mount.options=rw,relatime,data=ordered
mounted=2016-10-06 23:22:04 state=mounted
  *-disk:1
       description: SCSI Disk
       physical id: 0.1.0
       bus info: scsi@2:0.1.0
```

```
      logical name: /dev/sdb
      serial: v5V2zm-Lvbh-Mfx3-W8ki-COI9-DAtP-RXndhu
      size: 500GiB
      capacity: 500GiB
      capabilities: lvm2
      configuration: sectorsize=512
 *-disk:2
      description: SCSI Disk
      physical id: 0.2.0
      bus info: scsi@2:0.2.0
      logical name: /dev/sdc
      serial: fTQFJ2-giAV-WsXL-1Wha-V305-oQkV-qqS3SA
      size: 100GiB
      capacity: 100GiB
      capabilities: lvm2
      configuration: sectorsize=512
```

On hypervisor host locate disk attached to VM using bus info. Example: SCSI(0:1)



Extend virtual disk. See VMWare KB 1004047: http://kb.vmware.com/kb/1004047

Login back into VCO system console

Re-scan block device for resized physical volume. Example

```
echo 1 > /sys/block/$DEVICE/device/rescan
```

Example

```
echo 1 > /sys/block/sdb/device/rescan
```

Resize LVM physical disk

```
pvresize /dev/sdb
```

Determine amount of free space in database volume group

```
vgdisplay db_data |grep Free
```

Example:

```
root@vco:~# vgdisplay db_data |grep Free
  Free  PE / Size       34560 / 135.00 GiB
```

Extend database logical volume

```
lvextend -L+#G /dev/db_data/vco
```

Example:

```
root@vco:~# lvextend -L+10G /dev/db_data/vco
  Extending logical volume vco to 385.00 GiB
  Logical volume vco successfully resized
```

Resize database volume filesystem:

```
resize2fs /dev/db_data/vco
```

Example:

```
root@vco:~# resize2fs /dev/db_data/vco
```

```
resize2fs 1.42.9 (4-Feb-2014)
Filesystem at /dev/db_data/vco is mounted on /store; on-line resizing
required
old_desc_blocks = 24, new_desc_blocks = 25
The filesystem on /dev/db_data/vco is now 100924416 blocks long.
```

See new size of the volume

```
df -h /dev/db_data/vco
```

Example:

```
root@vco:~# df -h /dev/db_data/vco
Filesystem               Size  Used Avail Use% Mounted on
/dev/mapper/db_data-vco  379G  1.2G  359G   1% /store
```

Truncating Partitions

Contact Velocloud Support before running this commands.

Usage: /opt/vc/bin/mysql_vco_truncate_partition table partition

Table:

VELOCLOUD_EDGE_FIREWALL_LOGS

VELOCLOUD_ENTERPRISE_EVENT

VELOCLOUD_FILE_PROCESSING_QUEUE

VELOCLOUD_FLOW_STATS

VELOCLOUD_FLOW_STATS_RES_24

VELOCLOUD_FLOW_STATS_RES_576

VELOCLOUD_LINK_QUALITY_EVENT

VELOCLOUD_LINK_STATS

VELOCLOUD_OPERATOR_EVENT

VELOCLOUD_PROXY_EVENT

Partition: partition number from 0 to 11 corresponding to the month (0 - Jan, 1- Feb, …)

Example: Truncate flow stats for the month of September

```
/opt/vc/bin/mysql_vco_truncate_partition VELOCLOUD_FLOW_STATS 8
```

SNMP Integration

Edit /etc/snmp/snmpd.conf. Add the following lines (bold) to the config with source IP of the systems that will be connecting to SNMP service:

```
agentAddress udp:161

com2sec local localhost vc-vco
com2sec myenterprise 10.0.0.0/8 vc-vco

group rogroup v2c local
group rogroup v2c myenterprise

view all included .1 80

access rogroup "" any noauth exact all none none

#sysLocation    Sitting on the Dock of the Bay
#sysContact     Me <me@example.org>
```

```
sysServices    72

master agentx

#
#  Disk Monitoring
#
                                # 100MBs required on root disk, 5%
free on /var, 10% free on all other disks
disk       /     100000
disk       /var  5%
includeAllDisks  10%

#
#  System Load
#
                                # Unacceptable 1-, 5-, and 15-minute
load averages
load   12 10 5
```

Modify local firewall rules: etc/iptables/rules.v4. Add the following line (bold) to enable source IP of the systems that will be connecting to SNMP service:

```
*filter

:INPUT ACCEPT [0:0]

-A INPUT -m conntrack --ctstate RELATED,ESTABLISHED -m comment --
comment "allow established" -j ACCEPT

-A INPUT -p udp -m udp --source 127.0.0.1 --dport 161 -m comment --
comment "allow SNMP port" -j ACCEPT

-A INPUT -p udp -m udp --source 10.0.0.0/8 --dport 161 -m comment --
comment "allow SNMP port" -j ACCEPT

-A INPUT -p tcp -m tcp --dport 80 -m comment --comment "nginx HTTP" -
j ACCEPT

-A INPUT -p tcp -m tcp --dport 443 -m comment --comment "nginx HTTPS"
```

```
-j ACCEPT

-A INPUT -p tcp -m tcp --dport 22 -m comment --comment "allow ssh
port" -j ACCEPT

-A INPUT -i lo -m comment --comment "allow local connections" -j
ACCEPT

-A INPUT -m comment --comment "block everybody else" -j DROP

:FORWARD ACCEPT [0:0]

:OUTPUT ACCEPT [0:0]

COMMIT
```

Restart snmp and iptables services:

```
service snmpd restart
service iptables-persistent restart
```

Configure local mail forwarding

System mail is delivered locally to root account. VCO has postfix server pre-installed for local mail delivery. If mail alerts from the local system need to be forwarded to external email address, then postfix will need to be configured to forward email.

Login to system console and run:

```
sudo dpkg-reconfigure postfix
```

1) Choose 'Internet with smarthost' (recommended) to use another mail server for relaying mail

2) Enter system mail name (vco.customer.net)

3) Enter SMTP relay host (mail.customer.net)

4) Root and postmaster mail recipient (admin@customer.net)

5) Other destinations to accept mail for (e.g.: vco, localhost.localdomain, localhost)

6) Force synchronous updates on mail queue (No)

7) Local networks (127.0.0.0/8 [::ffff:127.0.0.0]/104 [::1]/128)

8) Use procmail for local delivery (Yes)

9) Mailbox size limit (bytes) (0 - unlimited or number. For example 1G: 1000000000)

10) Local address extension character (+)

11) Internet protocols to use (ipv4)

Backing Up the Orchestrator

Setup database backup location. Normally, it is NFS or local volume mounted at /archive endpoint. Contact Velocloud Support about additional information on setting this up.

Create /archive/vco/db folder

Edit etc/cron.d/vco_jobs and uncomment backup job.

```
# database backup
5 1 * * 0 root      /opt/vc/scripts/db_backup.sh /archive/vco/db

# database verification
5 1 * * 6 root      /opt/vc/scripts/db_verify.sh
```

Restoring the Orchestrator

For restoring VCO from the backup please contact Velocloud Support

Archiving Data

Setup database backup location. Normally, it is NFS or local volume mounted at /archive endpoint. Contact Velocloud Support about additional information on setting this up.

Create /archive/vco/stats folder

Edit etc/cron.d/vco_jobs and uncomment backup job.

```
# archive VCO logs/stats on 2nd of each month
5 0 2 * * root      /opt/vc/scripts/archive_data_monthly.sh
/archive/vco/stats

# purge VCO logs/stats on 10th of each month
5 0 10 * * root     /opt/vc/scripts/purge_data_monthly.sh
```

Disaster Recovery

Overview

The VeloCloud Orchestrator (VCO) Disaster Recovery (DR) feature prevents the loss of stored data and resumes VCO services in the event of system or network failure. The approach taken is to stand up an active/standby VCO pair with data replication and a manually-triggered failover mechanism. The recovery time objective (RTO) therefore is dependent on explicit action by the operator to trigger promotion of the standby. The recovery point objective (RPO) however is essentially zero, regardless of the recovery time because all configuration is instantaneously replicated, and monitoring data that would have been collected during the outage is cached on the edges and gateways pending promotion of the standby.

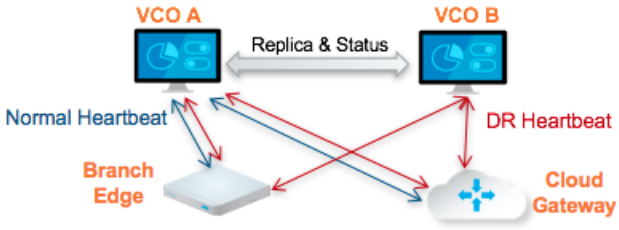
Feature Description

In a VCO DR deployment, two identical VCO systems are configured as an active / standby pair. The operator can view the state of DR readiness through the web UI on either of the servers. Edges and gateways are aware of both VCOs, and while they receive configuration changes only from the active VCO, they periodically send DR heartbeats to both systems to report their view of both servers and to query the DR system status. When the operator triggers a failover, the edges and gateways are informed of the change in their next DR heartbeat.

From the view of an operator, and of the edges and gateways, a VCO has one of four DR states:

- Standalone (no DR configured)
- Active (DR configured, acting as the primary VCO server)
- Standby (DR configured, acting as an inactive replica VCO server)
- Zombie (DR formerly configured and active, but no longer acting as the active or standby)

When DR is configured, the standby server runs in a limited mode, blocking all API calls except those related to the DR status and the DR heartbeats. When the operator invokes a failover, the standby is promoted to become fully operational as a Standalone server. The server that was formerly active is automatically transitioned to a Zombie state if it is responsive and visible from the promoted standby. In the Zombie state, management configuration services are blocked and any contact from edges and gateways that have not transitioned to the new active VCO are redirected to the promoted server.



Phases	VCO A Role	VCO B Role
Initial	Standalone	Standalone
Pairing	Active	Standby
Failover	Zombie	Standalone

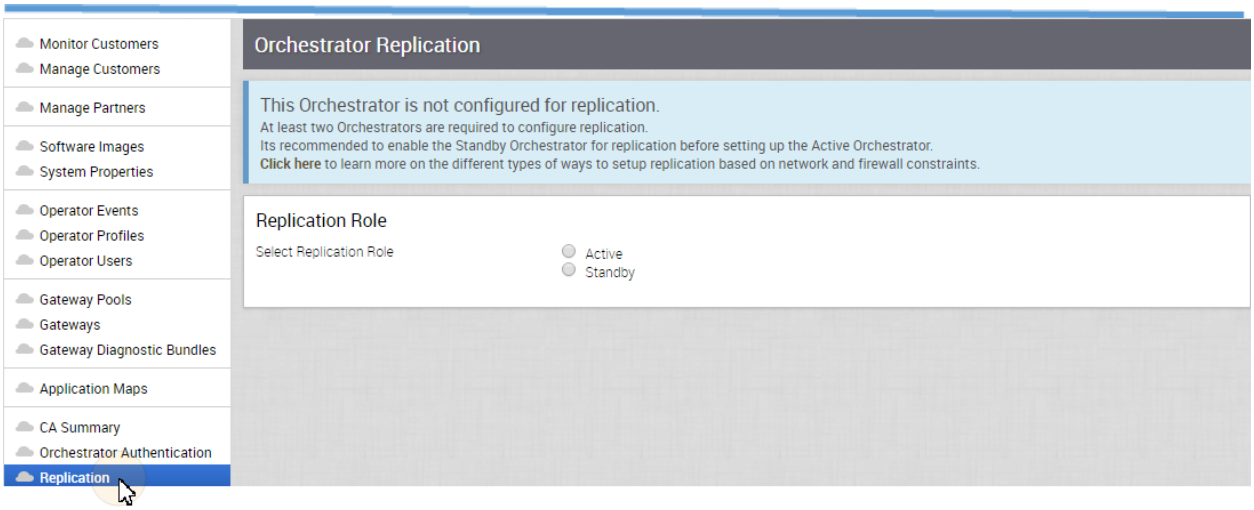
Setting Up VCO Replication

Two installed VCO instances are required to initiate replication. The selected standby is put into a STANDBY_CANDIDATE state, enabling it to be configured by the active server. The active server is then given the address and credentials of the standby and it enters the ACTIVE_CONFIGURING state. When a STANDBY_CONFIG_RQST is made from active to standby, the two servers synchronize through the state transitions shown below.

Note: You must have two installed VCOs to set up replication. It is highly recommended that you enable the Standby Orchestrator first.

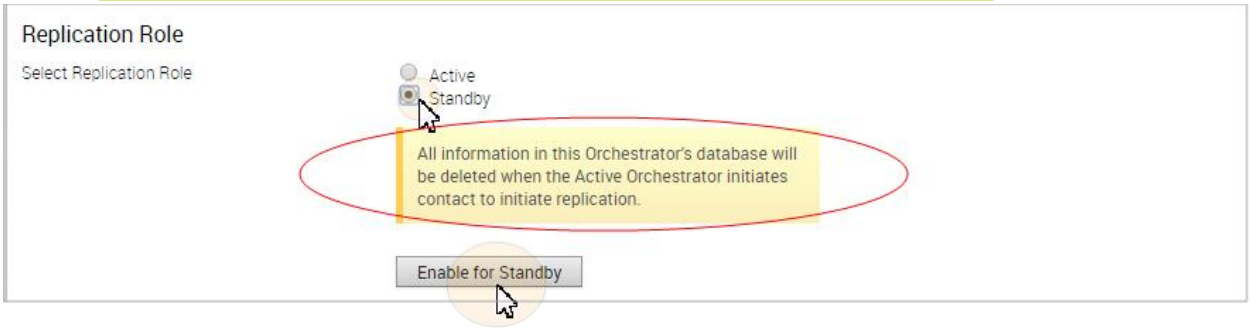
Standby Orchestrator Setup

1. To set up VCO replication (using the first of two VCOs), click Replication from the Navigation panel to display the Orchestrator Replication screen. See figure below.



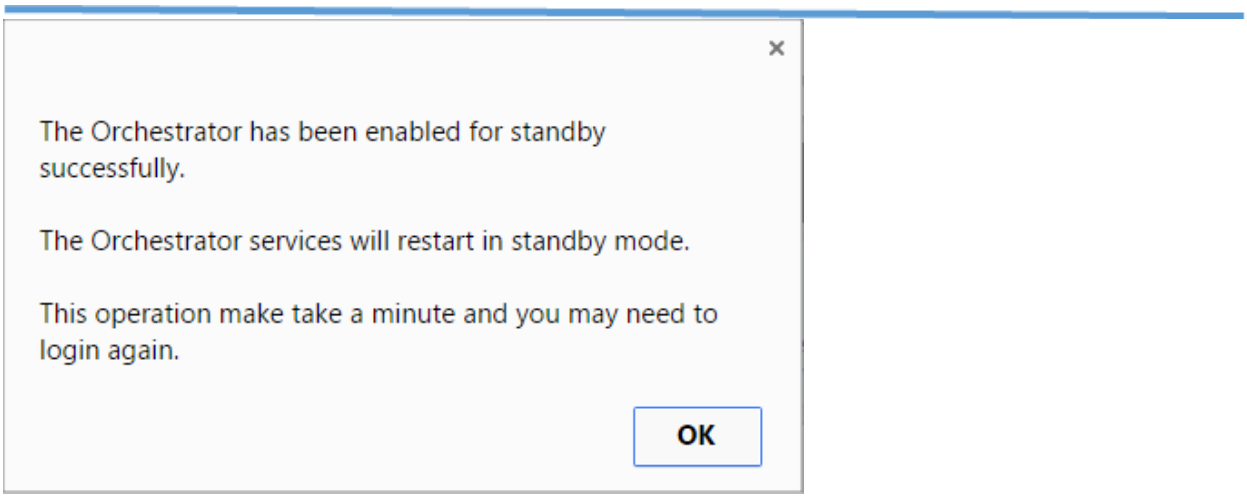
2. Enable the Standby Orchestrator by selecting the Standby (Replication Role) radio button.

   Note the following warning message on the screen: When you enable the VCO for the Standby Replication Role, all data in that database will be permanently deleted when the Active Orchestrator initiates data replication. The standby database should be backed up using SQL tools or volume snapshot if that data should be retained. See image below.



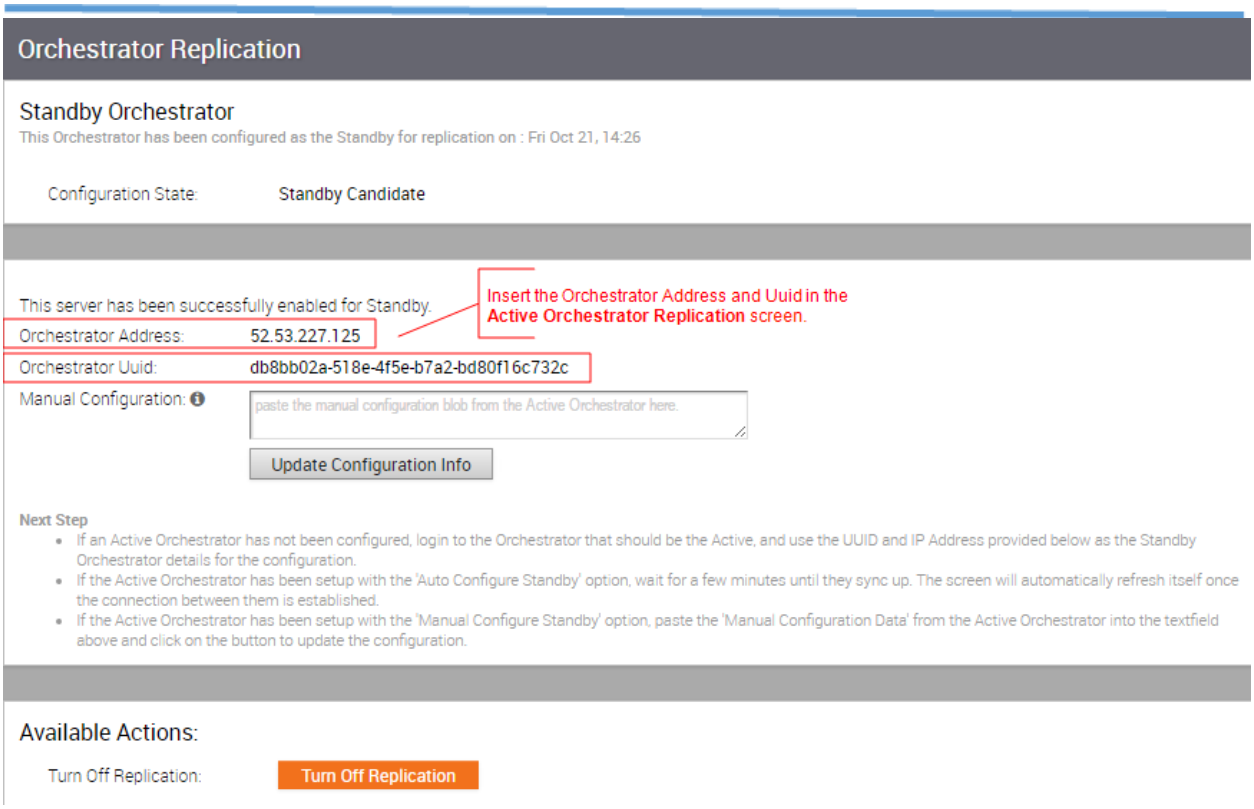
3. Click the **Enable for Standby** button.

   The **Orchestrator Success** dialog box displays indicating that the Orchestrator has been enabled for Standby, and that the Orchestrator will restart in Standby mode, as shown in the image below.



4. Click **OK**.

Note: As shown in the image below, the Standby Orchestrator screen displays the Orchestrator Uuid and the Orchestrator Address. You will need this information when you set up the Active Orchestrator.



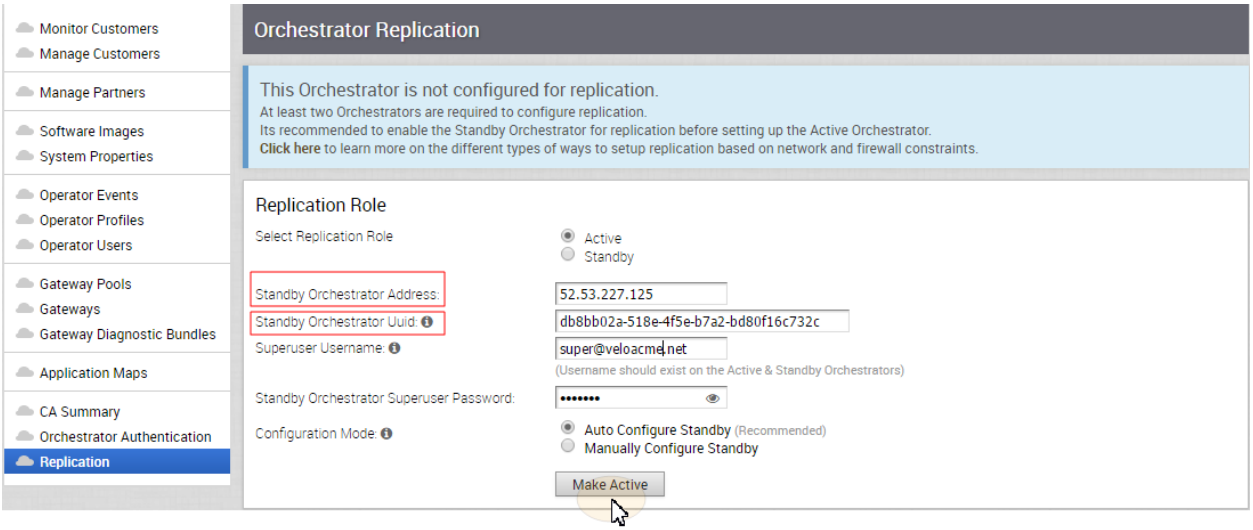
After the Standby Orchestrator has been configured for replication, configure the Active Orchestrator by following the instructions below.

Active Orchestrator Setup

1. Configure the second VCO, which will be the Active Orchestrator, by clicking Replication from the Navigation panel. (The Orchestrator Replication screen displays).

2. Choose the Active Replication Role as shown in the figure below.

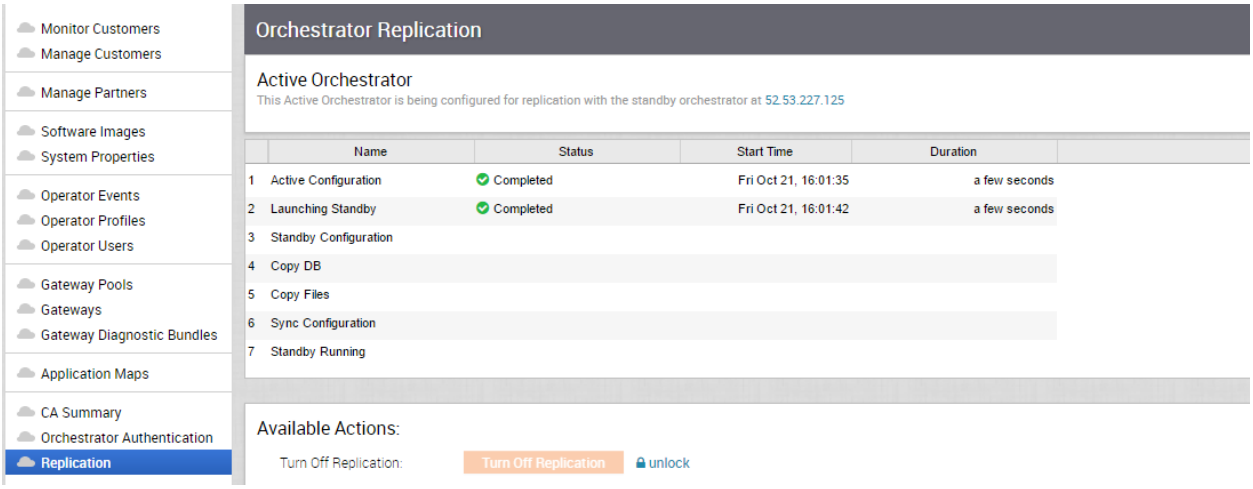
3. Type in the Standby Orchestrator Address and the Standby Orchestrator Uuid. The Orchestrator Address and Uuid are displayed in the Standby Orchestrator screen. (See image above).



4. Type in the username and password for the Orchestrator Superuser to be used for replication. (This Superuser should already exist on both systems).

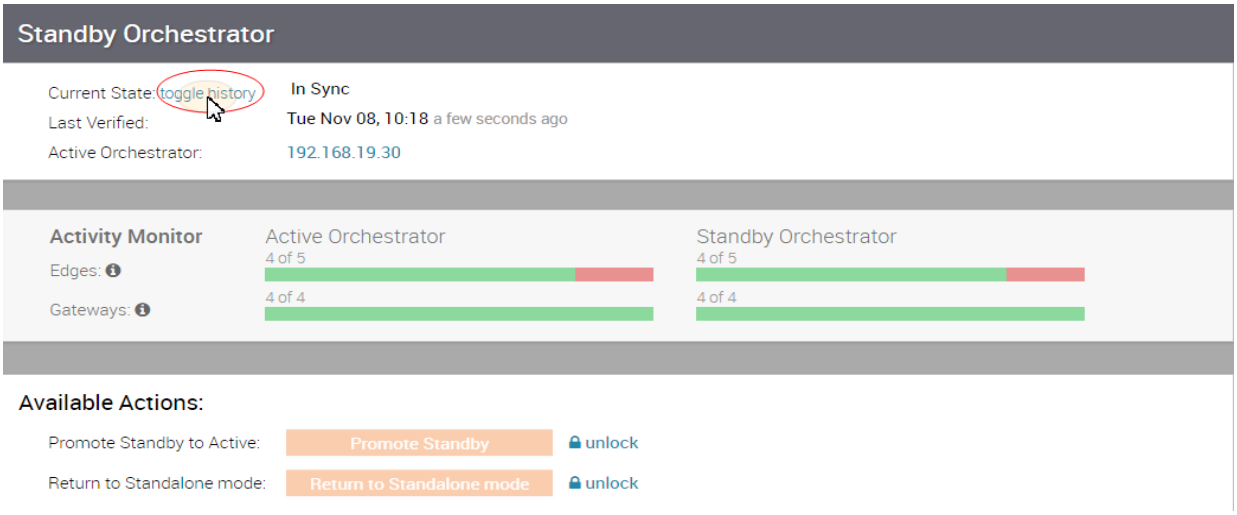
5. Click the **Make Active** button.

The **Active Orchestrator** screen displays showing a status of the current state.

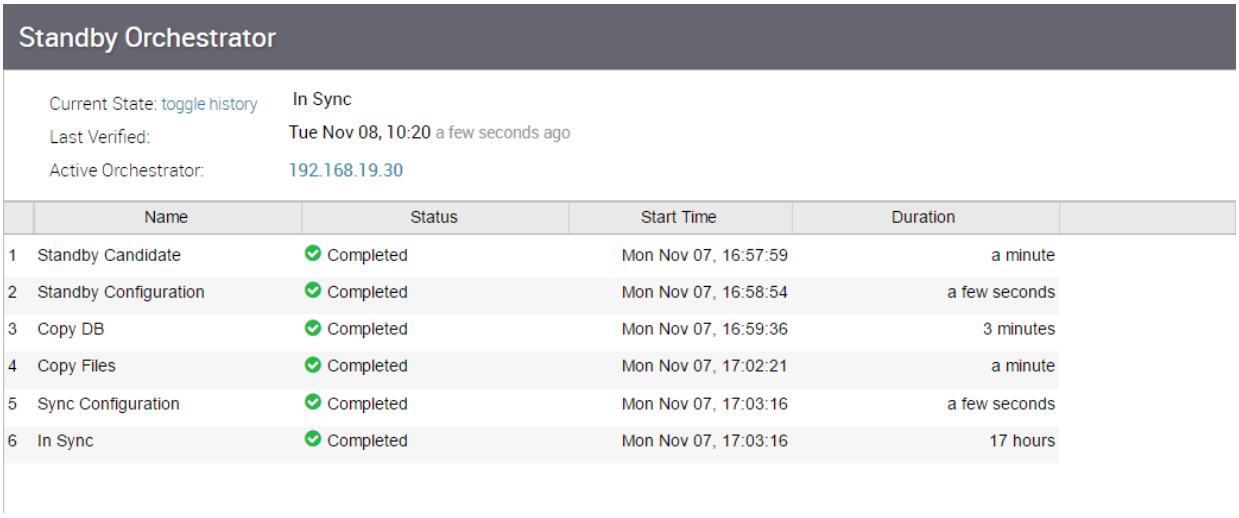


When configuration is complete, both Orchestrators (Standby and Active) will be in sync. See images below.

Standby Orchestrator in Sync

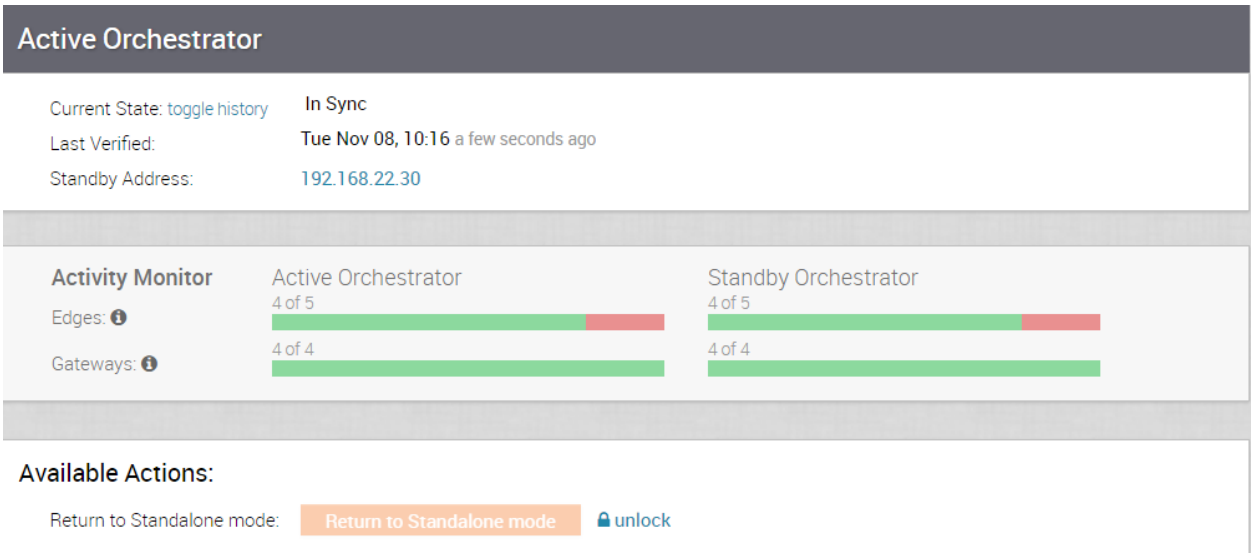


You can click the **toggle history** link to view the status of each state as shown in the image below.



	Name	Status	Start Time	Duration
1	Standby Candidate	Completed	Mon Nov 07, 16:57:59	a minute
2	Standby Configuration	Completed	Mon Nov 07, 16:58:54	a few seconds
3	Copy DB	Completed	Mon Nov 07, 16:59:36	3 minutes
4	Copy Files	Completed	Mon Nov 07, 17:02:21	a minute
5	Sync Configuration	Completed	Mon Nov 07, 17:03:16	a few seconds
6	In Sync	Completed	Mon Nov 07, 17:03:16	17 hours

Active Orchestrator in Sync

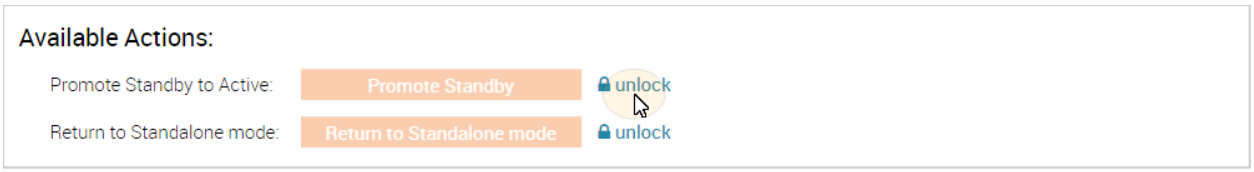


Testing Failover

The following testing failover scenarios are forced failovers for example purposes. You can perform these actions in the **Available Actions** area of the **Active** and **Standby** screens.

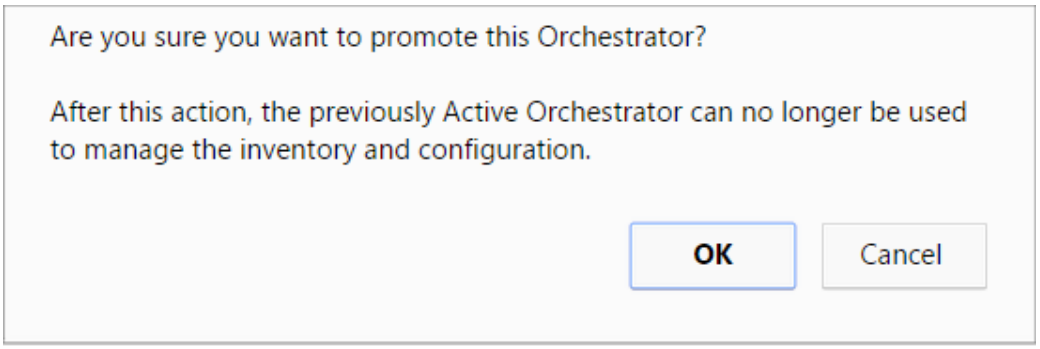
Promote Standby

You can promote a Standby Orchestrator by clicking the **unlock** link, and then clicking the **Promote Standby** button in the **Available Actions** area on the **Standby Orchestrator** screen. See image below.



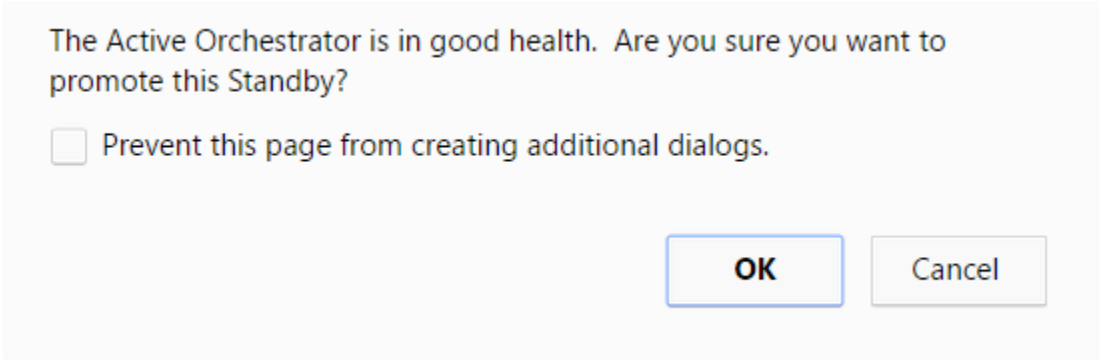
The following dialog box will display indicating that when you promote your Standby Orchestrator, administrators will no longer be able to manage the VCO using the previously Active Orchestrator.

Click the **OK** button to promote the Standby Orchestrator.



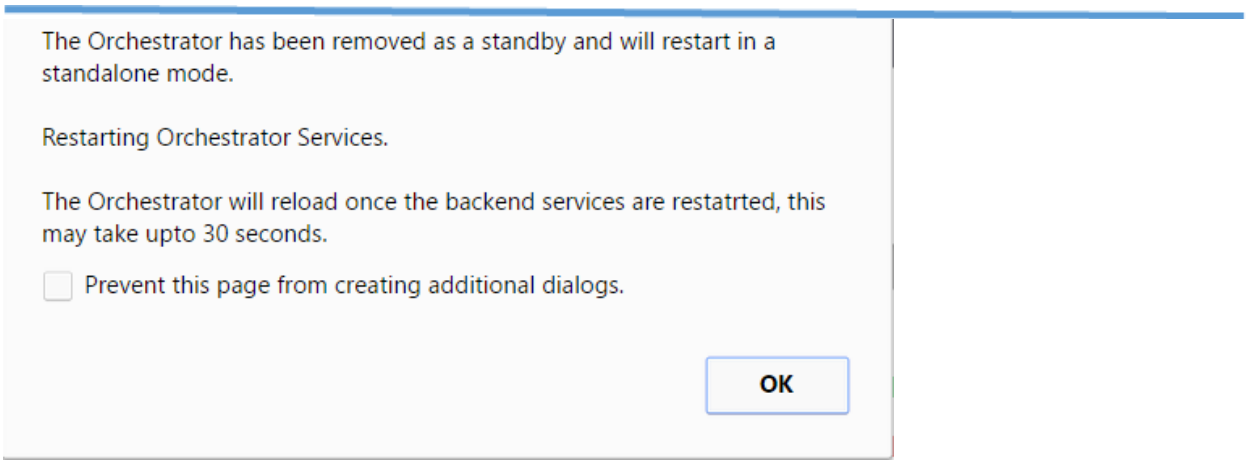
Another message dialog box displays to verify your request to promote the Standby Orchestrator. This message will only display if the Standby Orchestrator perceives the Active Orchestrator to be in good health, meaning the Standby is communicating with the Active and duplicating data.

Click **OK** to promote the Orchestrator.



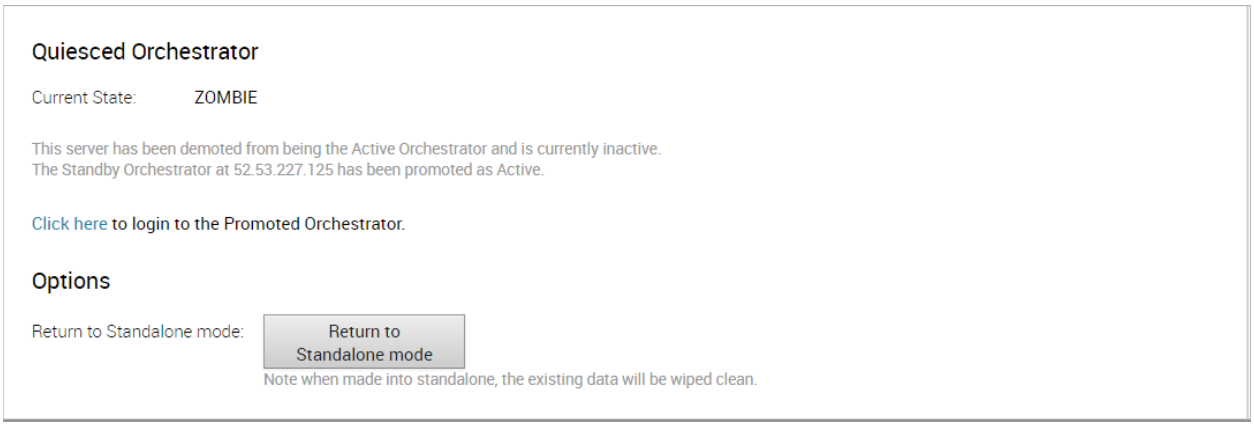
Note: If you would like to cancel your request, click the Cancel button. An "Action Canceled Successfully" dialog displays.

After you click **OK**, a final dialog box displays indicating that the Orchestrator is no longer a Standby and will restart in Standby mode, as shown in the image below.



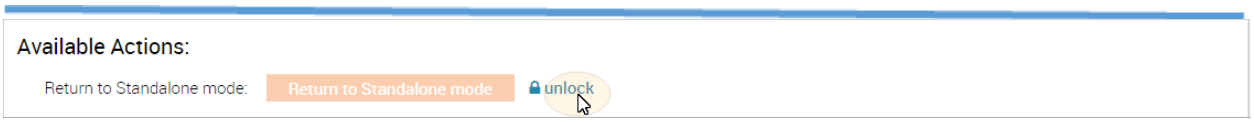
When you promote a Standby Orchestrator, it restarts in Standalone mode.

If the Standby can communicate with the formerly Active Orchestrator, it will instruct that Orchestrator to enter a Zombie state. In Zombie state, the Orchestrator communicates with its clients (edges, gateways, UI/API) that it is no longer active, and that they must communicate with the newly promoted Orchestrator. If the promoted Standby cannot communicate with the formerly Active Orchestrator, the operator should if possible manually demote the formerly Active.



Return to Standalone Mode

To return the Zombie to standalone mode, click the Return to Standalone Mode button in the Available Actions area on the Active Orchestrator or Standby Orchestrator screens. See image below.



Troubleshooting

The following are the failure states of the system, which are also listed in the UI along with a more detailed description of the failure. Additional information is available in the VeloCloud log.

Recoverable Failures

The failures listed below can occur after VCO DR reaches an in sync state. If the problem causing these failures is corrected, VCO DR will automatically return to normal operation.

- FAILURE_GET_STANDBY_STATUS
- FAILURE_MYSQL_ACTIVE_STATUS
- FAILURE_MYSQL_STANDBY_STATUS

Unrecoverable Failures

The failures listed below can occur during configuration of the VCO DR. VCO DR will not automatically recover from these failures.

- FAILURE_ACTIVE_CONFIGURING
- FAILURE_LAUNCHING_STANDBY
- FAILURE_STANDBY_CONFIGURING
- FAILURE_COPYING_DB
- FAILURE_COPYING_FILES
- FAILURE_SYNC_CONFIGURING
- FAILURE_GET_STANDBY_CONFIG
- FAILURE_STANDBY_CANDIDATE
- FAILURE_STANDBY_UNCONFIG
- FAILURE_STANDBY_PROMOTION
- FAILURE_ACTIVE_DEMOTION

VeloCloud Gateway

This section of the OAM guide will explain how Partner Gateways are installed. Partner Gateways are gateways tailored to on-premise operation where the Gateway is installed with two interfaces.

One interface is facing the private and/or public WAN network and is dedicated to receiving VCMP encapsulated traffic from the remote edges as well as standard IPsec traffic from non-VeloCloud sites.

Another interface faces the datacenter and provides access to resources or networks attached to a PE router to which the Partner Gateway is connected to. The PE router typically affords access to shared managed services that are extended to the branches or access to a private (MPLS / IP-VPN) core network in which individual customers are segregated.

Prerequisites

Instance Requirements

VeloCloud recommends to install the Gateway as a virtual instance where the instances needs to meet the following minimum system specifications:

- 4 Intel vCPU's at 2.0 Ghz or higher. CPU must support AES-NI, SSSE3 and RDTSC instruction sets. 8 vCPU for gateways with DPDK support
- 8 GB of memory, 16GB for gateways with DPDK support
- 24 GB magnetic (minimum 100 IOPS) or SSD based, persistent disk volume
- 1 Gbps (or higher) network interface facing Edges
    - eth0 must be the interface connected to the public network and responsible for serving customer traffic
    - Single public IP address must be provisioned on eth0 (Can be either applied directly or be NAT'd on an upstream device)
- 1 Gbps (or higher) network interface facing datacenter hosted services or networks
    - Attached to eth1 and responsible for handling of customer traffic to the partner PE router.

    - PE router must be able to support 802.1q VLAN tagging as well as QinQ support
- DPDK support requires DPDK-compatible network card
- OS type: Ubuntu 14.04 LTS x64 server

VeloCloud recommends using the following components for the underlying hardware:

- Intel Xeon E5 Family processors
- Intel dual 10 Gbps NIC with support for DPDK (http://dpdk.org/doc/nics) to take advantage of future performance improvements
- Support for link bonding
- Consider disabling all power management in both the BIOS and host OS
- Recommended Server:
    - HP DL380G9 with NIC that has 82599/82599ES chipset.
    - Here are the detailed specs of this server: http://www.hp.com/hpinfo/newsroom/press_kits/2014/ComputeEra/HP_ProLiantDL380_DataSheet.pdf
    - Here are the detailed specs for the NIC with the 82599/82599ES chipset: https://www.hpe.com/h20195/v2/GetPDF.aspx/c04111506.pdf
-

Upstream Firewall Configuration

The VeloCloud gateway will receive VCMP (VeloCloud MultiPath Protocol) traffic from branches that originates from the deployed edges as well as traffic from non-VeloCloud sites that that connect to the gateways using standards based IPsec tunnels.

To allow this traffic to reach the VeloCloud gateway instances, upstream firewalls need to be configured to allow the following inbound flows:

-
- UDP/2426 (VCMP): VeloCloud MultiPath Protocol, providing transport abstractions from the underlying physical circuits
-

Cloud-init Preparation

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 Gateway 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: vcg01
local-hostname: vcg-01
```

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

```
network-interfaces: |
### MAIN INTERFACE###
auto eth0
iface eth0 inet static
metric '1'
 address 210.193.164.98
 netmask 255.255.255.240
 gateway 210.193.164.97
 dns-nameservers 8.8.8.8 8.8.4.4
### HANDOFF INTERFACE (optional) ###
auto eth1  IF USING TWO ARM DEPLOYMENT
iface eth1 inet static
 metric '13'
 address 100.125.1.34
 netmask 255.255.255.224
 gateway 100.125.1.33
```

```
 dns-nameservers 8.8.8.8 8.8.4.4
### MANAGEMENT INTERFACE (optional) ###
auto eth2
iface eth2 inet static
 address 192.168.225.102
 netmask 255.255.255.0
 up route add -net 10.0.0.0 netmask 255.0.0.0 gw 192.168.225.1
 up route add -net 192.168.0.0 netmask 255.255.0.0 gw 192.168.225.1
 dns-nameservers 8.8.8.8 8.8.4.4
```

Create cloud-init user-data file

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 Gateway. The following is an example of what the user-data file will look like:

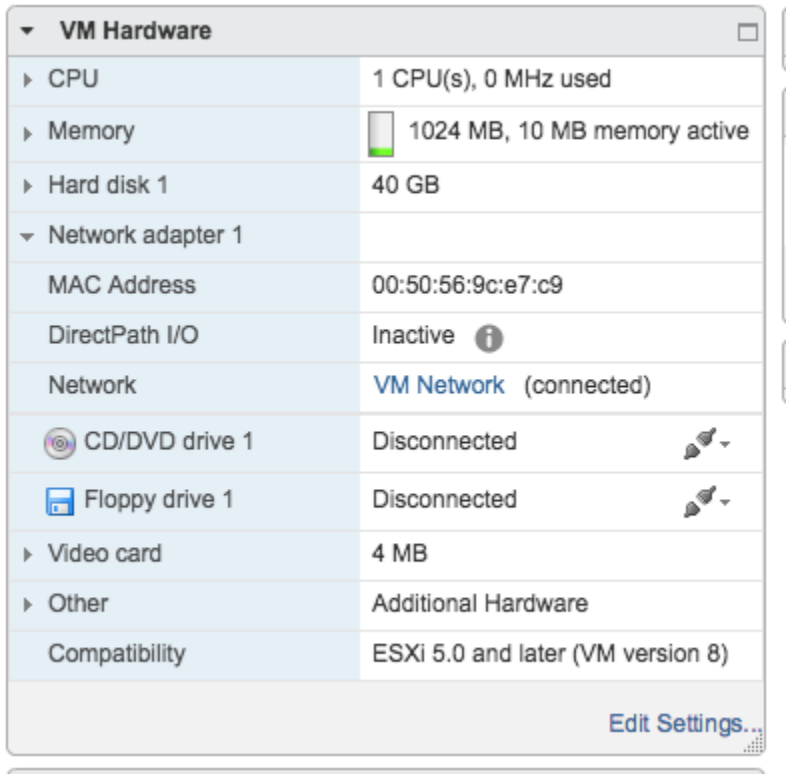
```
#cloud-config
password: Velocloud123
chpasswd: {expire: False}
ssh_pwauth: True
ssh_authorized_keys:
  - ssh-rsa AAA...SDvz user1@yourdomain.com
  - ssh-rsa AAB...QTuo user2@yourdomain.com
```

This user-data file enables the default user, vcadmin, to log in either with a password or with an SSH key. The use of both methods is possible but not required. 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.

The following is an example of user-data file:

```
#cloud-config
hostname: vcg
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
###  Used to register to VCO (optional) ###
velocloud:
   vcg:
      vco: 10.0.0.100
      activation code: XXXX-XXXX-XXXX
final_message: "==== Welcome to Velocloud ===="
write_files:
### NEEDED FOR VMWARE, THE MAC ADDRESS IS THE VM INTERFACE ADDRESS, AS SHOWN IN FIGURE 2 ###
 - path: "/etc/udev/rules.d/70-persistent-net.rules"
   permissions: '0644'
   content: |
      SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}=="00:50:56:93:26:e5",
ATTR{type}=="1", KERNEL=="eth*", NAME="eth0"
      SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}=="00:50:56:93:71:87",
ATTR{type}=="1", KERNEL=="eth*", NAME="eth1"
      SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}=="00:50:56:93:d8:b0",
ATTR{type}=="1", KERNEL=="eth*", NAME="eth2"
```

Figure 2. Find the MAC Address of VM Interface in VMWare



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 vcg01-cidata.iso, is created with the following command on Linux system:

```
genisoimage -output vcg01-cidata.iso -volid cidata -joliet -rock user-data
meta-data
```

NOTE: Always validate user-data and metadata, using http://www.yamllint.com/ for example.

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

VMWare

VMware vSphere provides a means of deploying and managing virtual machine resources. This section explains how to run Velocloud Gateway 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.

Log in to the vSphere Client.

Select **File > Deploy OVF Template**.

Respond to the prompts with information specific to your deployment.

Source: Type a URL or navigate to the OVA package location.

OVF template details: Verify that you pointed to the correct OVA template for this installation.

Name and location: Name of the virtual machine

Storage: Select the location to store the virtual machine files.

Provisioning: Select the provisioning type. "thin" is recommended for database and binary log volumes

Network mapping: Select the network for each virtual machine to use. There are 2 network interfaces “Outside Network” - for public interface. “Inside network” - for private interface (e.g. Partner Handoff). “Inside Network” can be left unconnected. Please refer to Deployment scenarios document to determine which option is right for you.

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.

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

Right-click on the newly-added Velocloud Gateway VM and select Edit Settings.

From the Virtual Machine Properties window, select CD/DVD Drive.

Select the Use an ISO image option.

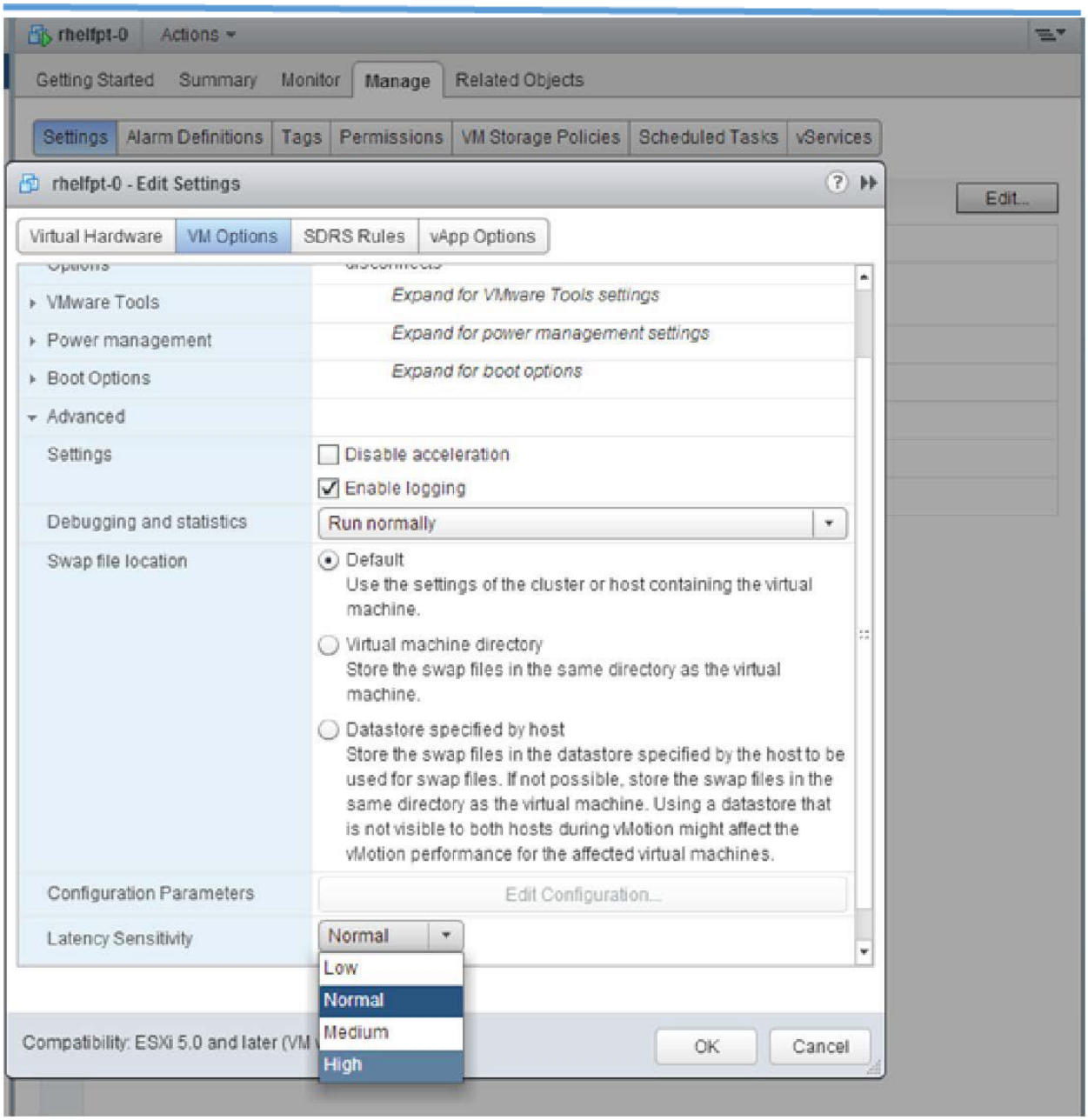
Browse to find the ISO image you created earlier (we called ours vcg1-cidata.iso), select it. The ISO can be found in the datastore that you uploaded it to, in the folder that you created.

Select “Connect on Power On”

Click OK to exit the Properties screen.

Set “Latency Sensitivity” option for VM

Under VM Setting -> VM Options, location “Latency Sensitivity” and set it to “High”



Run the Velocloud Gateway virtual machine

To start up the Velocloud Gateway virtual machine, click to highlight it, then select the Power On button.

Select the Console tab to watch as the virtual machine boots up.

If you configured Velocloud Gateway 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

Special Considerations

- Disable GRO (Generic Receive Offload) on physical interfaces (to avoid unnecessary re-fragmentation in VCG) using 'ethtool –K <interface> gro off tx off'
- Disable CPU C-states (power states affect real-time performance). Typically this can be done as part of kernel boot options by appending processor.max_cstate=1 or just disable in BIOS. See https://docs.fedoraproject.org/en-US/Fedora/13/html/Virtualization_Guide/chap-Virtualization-KVM_guest_timing_management.html
- For production deployment, vCPU's must be pinned to the instance. No oversubscription on the cores should be allowed to take place. See https://docs.fedoraproject.org/en-US/Fedora/13/html/Virtualization_Guide/ch25s06.html

KVM domain XML SAMPLE

```
<domain type='kvm'>
 <name>vcg-automation</name>
 <memory unit='KiB'>16388608</memory>
 <currentMemory unit='KiB'>16388608</currentMemory>
 <vcpu placement='static' cpuset='0-8'>8</vcpu>
  <cputune>
    <vcpupin vcpu='0' cpuset='4'/>
…… (Each cpu needs to be pinned to a hardware CPU)
  </cputune>

 <resource>
   <partition>/machine</partition>
 </resource>
 <os>
 <type>hvm</type>
 </os>
 <features>
```

```
    <acpi/>
    <apic/>
    <pae/>
  </features>
  <cpu mode='host-passthrough'>
  </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='#PWD#/VM/vcg/vcg-root.img'/>
      <target dev='hda' bus='ide'/>
      <alias name='ide0-0-0'/>
      <address type='drive' controller='0' bus='0' target='0' unit='0'/>
    </disk>
    <disk type='file' device='cdrom'>
      <driver name='qemu' type='raw'/>
      <source file='#PWD#/VM/vcg/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='network'>
      <source network='passthrough'/>
      <vlan>
      <tag id='100'/>
      </vlan>
      <alias name='hostdev1'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x11' function='0x0'/>
    </interface>
    <interface type='network'>
      <source network='passthrough'/>
      <vlan>
      <tag id='102'/>
      </vlan>
      <alias name='hostdev2'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x12' function='0x0'/>
    </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='none'/>
 </devices>
 <seclabel type='none' />
</domain>
```

Gateway Configuration

Network Planning for VeloCloud Gateway with Partner Handoff

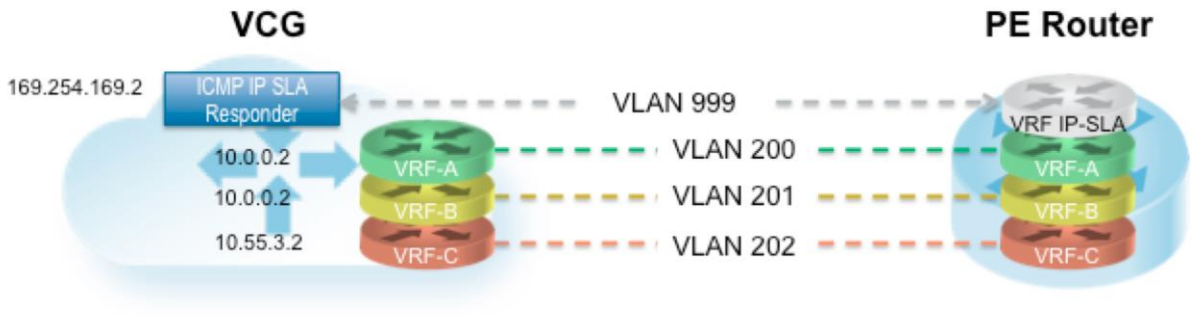


Figure 1 - VRF/VLAN Hand Off to PE

In this example, we assume eth0 is the interface facing the public network (Internet) and eth1 is the interface facing the internal network (customer VRF through the PE). BGP peering configuration is managed on the VCO on a per customer/VRF basis under “Configure > Customer”. Note that the IP

address of each VRF is configurable per customer. The IP address of the management VRF inherits the IP address configured on the VCG interface in Linux.

A **Management VRF** is created on the VCG and is used to send periodic ARP refresh to the default gateway IP to determine the next-hop MAC. It is recommended that a dedicated VRF is set up on the PE router for this purpose. The same management VRF can also be used by the PE router to send IP SLA probe to the VCG to check for VCG status (VCG has stateful ICMP responder that will respond to ping only when its service is up). BGP Peering is not required on the Management VRF.

If a Management VRF is not set up, then you can use one of the customer VRFs as Management VRF, although this is not recommended.

IMPORTANT: Note the metric for the public interface (eth0) must be lower than the Hand Off interface so eth0 is properly used as the default route for Internet traffic. Also, a default gateway **must** be configured on the Inside interface for the VCG to properly monitor the status of the interface.

Partner Gateway Configuration steps

After gateways are installed, the natively operate as a NAT gateway. Minimal configuration changes need to be done to convert the installation to a Partner Gateway to allow use of the Partner Handoff functionality.

STEP 1 -- Define the network interface in **/etc/network/interfaces** file.

NOTE: If it is done in the cloud-init meta-data, this step is not required.

Note that for the handoff interface (eth1 in this example), the IP address and default gateway specified in the file below is used as template to populate IP address and default gateway in every VRF that the VCG creates. The expectation is the peer PE router's IP address in every VRF is the same as what is specified in the VCG eth1 gateway IP below.

```
# OUTSIDE OR PUBLIC INTERFACE

auto eth0
```

```
iface eth0 inet static
    metric "1"
    address <IP ADDRESS>
    netmask <NETMASK
    network <NETWORK ADDRESS>
    broadcast <BROADCAST>
    gateway <NEXT-HOP PUBLIC>
    # dns-* options are implemented by the resolvconf package, if
installed
    dns-nameservers 8.8.8.8 8.8.4.4
    dns-search localdomain

#INSIDE OR HANDOFF INTERFACE
auto eth1
iface eth1 inet static
    metric "2"
    address 192.168.104.14
    netmask 255.255.255.0
    network 192.168.104.255
    broadcast 192.168.104.0
    gateway 192.168.104.1
```

STEP 2 -- Edit the **/etc/config/gatewayd** and specify the correct VCMP and WAN interface. VCMP interface is the interface that terminates the overlay tunnels. The WAN interface in this context is the handoff interface.

```
    "vcmp.interfaces": [
      "eth0"
       ],
(..snip..)
```

```
"wan": [
    "eth1"
    ],,
```

STEP 3 -- Configure the **Management VRF**. This VRF is used by the VCG to ARP for next-hop MAC (PE router). The same next-hop MAC will be used by all the VRFs created by the VCG. You need to configure the **Management VRF** parameter in **/etc/config/gatewayd**.

The Management VRF is the same VRF used by the PE router to send IP SLA probe to. The VCG only responds to the ICMP probe if the service is up and if there are edges connected to it. Below table explains each parameter that needs to be defined. This example has Management VRF on the 802.1q VLAN ID of 1000.

mode	QinQ (0x8100), QinQ (0x9100), none, 802.1Q, 802.1ad
c_tag	C-Tag value for QinQ encapsulation or 802.1Q VLAN ID for 802.1Q encapsulation
s_tag	S-Tag value for QinQ encapsulation
interface	Handoff interface, typically eth1

```
"vrf_vlan": {
    "tag_info": [
        {
         "resp_mode": 0,
         "proxy_arp": 0,
         "c_tag": 1000,
         "mode": "802.1Q",
```

```
 "interface": "eth1",
 "s_tag": 0
}
]
},
```

STEP 4 -- Edit the /etc/config/gatewayd-tunnel to include both interfaces in the wan parameter.

wan="eth0 eth1"

STEP 5 -- Remove blocked subnets

By default, the VCG blocks traffic to 10.0.0.0/8 and 172.16.0.0/14. We will need to remove them before using this VCG because we expect VCG to be sending traffic to private subnets as well. If you do not edit this file, when you try to send traffic to blocked subnets, you will find the following messages in /var/log/gwd.log

2015-12-18T12:49:55.639 ERR [NET] proto_ip_recv_handler:494 Dropping packet destined for 10.10.150.254, which is a blocked subnet.

2015-12-18T12:52:27.764 ERR [NET] proto_ip_recv_handler:494 Dropping packet destined for 10.10.150.254, which is a blocked subnet. [message repeated 48 times]

2015-12-18T12:52:27.764 ERR [NET] proto_ip_recv_handler:494 Dropping packet destined for 10.10.150.10, which is a blocked subnet.

On VCG, edit **/opt/vc/etc/vc_blocked_subnets.json** file. You will find that this file first has the following.

```
[
{
"network_addr": "10.0.0.0",
"subnet_mask": "255.0.0.0"
},
{
```

"network_addr": "172.16.0.0",

"subnet_mask": "255.255.0.0"

}

]

Remove the two networks. The file should look like below after editing. Save the change.

[

]

Restart the VCG process by **sudo /opt/vc/bin/vc_procmon restart**.

Gateway Provisioning

After a VeloCloud gateway instance has been prepared, the new gateway needs to be provisioned in the Orchestrator

[INCLUDE SCREEN SHOTS]

Gateway Activation

[TO BE COMPLETED]

Partner Gateway Configuration

[TO BE COMPLETED]

Custom Firewall Rules

Modify local firewall rules: /etc/iptables/rules.v4

IMPORTANT: only add targeted rules for addresses and ports. Do NOT add blanket drop or accept rules. VCG will append its own rules to the table and, since the rules are evaluated in order, that may prevent gateway software from functioning properly

```
# WARNING: only add targeted rules for addresses and ports
#          do not add blanket drop or accept rules since VCG will append its own
```

```
rules

#          and that may prevent it from functioning properly

*filter

:INPUT ACCEPT [0:0]

-A INPUT -p udp -m udp --source 127.0.0.1 --dport 161 -m comment --comment "allow
SNMP port" -j ACCEPT

:FORWARD ACCEPT [0:0]

:OUTPUT ACCEPT [0:0]

COMMIT
```

Restart iptables service:

```
service iptables-persistent restart
```

SNMP Integration

Edit /etc/snmp/snmpd.conf. Add the following lines (bold) to the config with source IP of the systems that will be connecting to SNMP service:

```
agentAddress udp:161

com2sec local localhost vc-vcg
com2sec myenterprise 10.0.0.0/8 vc-vcg

group rogroup v2c local
group rogroup v2c myenterprise
```

```
view all included .1 80

access rogroup "" any noauth exact all none none

#sysLocation    Sitting on the Dock of the Bay
#sysContact     Me <me@example.org>

sysServices    72

master agentx

#
#  Process Monitoring
#
                                # At least one  'gwd' process
proc  gwd
                                # At least one  'mgd' process
proc  mgd

#
#  Disk Monitoring
#
                                # 100MBs required on root disk, 5%
free on /var, 10% free on all other disks
disk       /     100000
disk       /var  5%
includeAllDisks  10%

#
#  System Load
#
                                # Unacceptable 1-, 5-, and 15-minute
load averages
load   12 10 5
```

Modify local firewall rules: /etc/iptables/rules.v4. Add the following line (bold) to enable source IP of the systems that will be connecting to SNMP service:

```
# WARNING: only add targeted rules for addresses and ports

#          do not add blanket drop or accept rules since VCG will append its own
rules

#          and that may prevent it from functioning properly

*filter

:INPUT ACCEPT [0:0]

-A INPUT -p udp -m udp --source 127.0.0.1 --dport 161 -m comment --comment "allow
SNMP port" -j ACCEPT

-A INPUT -p udp -m udp --source 10.0.0.0/8 --dport 161 -m comment --comment "allow
SNMP port" -j ACCEPT

:FORWARD ACCEPT [0:0]

:OUTPUT ACCEPT [0:0]

COMMIT
```

Restart snmp and iptables services:

```
service snmpd restart
service iptables-persistent restart
```

Upgrade

First download the Velocloud Gateway Update packageThen, upload image to Velocloud Gateway system, using, for example, scp command. Copy the image to the following location on the system: /var/lib/velocloud/software_update/vcg_update.tar

Connect to Velocloud Gateway console and run

```
sudo /opt/vc/bin/vcg_software_update
```

-