VNS3 API

v20120521

Requirements

You have an Amazon AWS account that CohesiveFT can use for enabling your access to a VNS3 (vns-cubed) AMI.

You have agreed to the terms of service provided for the VNS3 AMI.

Access to a deployment launch tool for launching the VNS3 AMI

- ElasticFox
- AWS Console
- Rightscale

Ability to create/configure EC2 security groups using a tools like; ElasticFox, AWS Console, CohesiveFT's Elastic Server On-Demand, or EC2 Command Line tools

Basic knowledge of Linux software installation and use of command line tools.

Getting Help with VNS3 API Preview

This guide assumes you know how to configure and launch a VNS3 Manager. If you do not please learn those steps using the VNS3 DataCenter Connect - Trial Edition available at www.cohesiveft.com

For support requests use our community forums at: **http://getsatisfaction.com/cohesiveft**

or for other support inquiries including paid support services contact:

sales@cohesiveft.com

Your Configuration Begins Here!

Getting Started with the VNS3 API

- For you initial use of the API you will need an encrypted license file for your VNS3 edition, or we will provide one in these instructions which gives a topology identical to the Datacenter Connect Trial Edition.
- You will also need to have available a snapshot of your existing manager configuration or you should make one for use with the API.
- Additionally the API uses a Ruby script and Ruby language binding for the API. You will need Ruby libraries on your machine.

VNS3 API: Calling Structure

VNS3 API calls take several arguments in common across all calls: “-K” for access key, “-S” for access secret, “-H” for the ip address of the manager you are connecting to.

Some of the calls can only be made after a manager is licensed or configured. This will be noted in the documentation for the API call.

Here is an example call that can be made at any time in a VNS3 Manager’s lifecycle.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address desc_config
```

```
$vns3api/vnscubed.rb -K api -S i-6b344d01 -H 174.129.238.73 desc_config
```

Response:

```
public_ipaddress: 174.129.238.73
vnscubed_version: 0.7.9999.2-20100823171847
licensed: false
private_ipaddress: 10.220.23.235
topology_checksum: 3309dfc9832c45280b590341e39de1277c17ffbd
```

The manager is unlicensed at this point, meaning neither a license, nor a snapshot with a license has been provided to it. The private_ipaddress is the actual underlying LAN address provided by the cloud provider. The topology checksum was randomly generated, and changes with the license and/or snapshot info.

VNS3 API: Calling Structure Arguments

In this document API calls have a list of arguments in the form of:
“**Argument Switch:** --enabled (boolean)”

The example above shows an allowed argument switch of “--enabled” that can have a value of true or false.

Here are common argument types and their meanings:

CIDR - a string in subnet format with a subnet mask specified in the form of “172.10.20.0/24” where the /24 is the subnet mask.

boolean - has a value of true or false

integer - a whole number greater than or equal to zero

positive_or_negative_integer - a whole number which can have a negative sign in front, or be zero or greater than zero

address-string - an IP address without a subnet mask “172.10.20.10” for example

string - just plain text, for example “this is my random crypto string”

pathname - a file system path and file name used as an input file or output file

VNS3 API: Notes

Basic Workflow Choices:

A) Use the web UI to “design” one or more networks. Take snapshots of the managers and start API testing from launching managers and calling “import_snapshot”

B) Start with launching managers and use “upload_license” to begin designing your network interactively with the API.

Notes:

There are no specific errata for VNS3 API at this time.

API Command Line Calls

Command Line Call	Call Category	Valid Pre-License
reset_password	Admin	**Yes**
setup_remote_support	Admin	No
generate_remote_support_keypair (new)	Admin	No
setup_admin_ui	Admin	No
server_action	Admin	No
edit_config	Admin	No
desc_config	Status	**Yes**
desc_status	Status	No
desc_client_status	Status	No
desc_system_status	Status	No
desc_link_history (changed)	Status	No
desc_connected_subnets (removed)	Status	No
desc_license	License	No
upload_license	License	**Yes**
upgrade_license (new)	License	No
set_license_parameters (new)	License	No
desc_keyset	Clientpacks	No
setup_keyset	Clientpacks	No
desc_clientpacks	Clientpacks	No
fetch_clientpacks	Clientpacks	No
get_next_available_clientpack	Clientpacks	No
edit_clientpack	Clientpacks	No
reset_client (new)	Clientpacks	No
reset_all_clients (new)	Clientpacks	No
desc_peering	Peering	No
set_manager_id	Peering	No
add_peer	Peering	No
edit_peer	Peering	No
delete_peer	Peering	No
desc_snapshot	Snapshots	No
create_snapshot	Snapshots	No
delete_snapshot	Snapshots	No
import_snapshot	Snapshots	**Yes**
fetch_snapshot	Snapshots	No

Command Line Call	Call Category	Valid Pre-License
desc_ipsec_status	IPsec	No
setup_ipsec	IPsec	No
edit_ipsec_config	IPsec	No
get_ipsec_local_address (deprecated)	IPsec	No
set_ipsec_local_address (deprecated)	IPsec	No
create_ipsec_endpoint	IPsec	No
edit_ipsec_endpoint	IPsec	No
delete_ipsec_endpoint	IPsec	No
desc_ipsec_endpoint	IPsec	No
create_ipsec_tunnel (new)	IPsec	No
edit_ipsec_tunnel (new)	IPsec	No
delete_ipsec_tunnel (new)	IPsec	No
create_ebgp_peer (new)	IPsec	No
delete_ebgp_peer (new)	IPsec	No
create_remote_subnet (removed)	IPsec	No
delete_remote_subnet (removed)	IPsec	No
desc_firewall	FW	No
add_firewall_rule	FW	No
delete_firewall_rule	FW	No
desc_routes	Routes	No
add_route	Routes	No
delete_route	Routes	No

VNS3 API: Admin

Call: reset_password
Argument Switch: --password (string)
Argument: Password you want to use for the API interaction.
Allowed Pre-License: Yes
Purpose: Allows you to change the API password/secret key. To change the Web UI password (or username) use setup_admin_ui.

command prompt>vnscubed.rb -K "api" -S "myapisecret" -H manager-ip-address reset_password --password "mysuperpassword"

$vns3api/vnscubed.rb -K api -S i-6b344d01 -H 174.129.238.73 reset_password --password "apidemopassword"

Response:
password_reset: ok

Note: Now that we have changed the password, we use this as our new "-S" argument.

VNS3 API: Admin

Call: setup_remote_support
Argument Switch: --enabled (boolean)
Argument: The “--enabled” arg specifies whether remote support is enabled.
Allowed Pre-License: No
Purpose: Enables and disables remote support.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address setup_remote_support --enabled true
```

```
$vns3api/vnscubed.rb -K api -S "apidemopassword" -H 174.129.238.73 setup_remote_support --enabled true
```

Response:

```
---
enabled: true
```

VNS3 API: Admin

Call: generate_remote_support_keypair
Argument Switch: --encrypted-passphrase (input pathname), -o (output pathname)
Argument: The “--encrypted-passphrase” arg specifies an encrypted passphrase file which will be used to generate an X509 key for accessing the VNS3 Manager in support mode, the “-o” argument lets you put in a path and output file name for the remote access key which can be used to access the internals of the VNS3 manager.
Allowed Pre-License: No
Purpose: Generating a remote support key which can be shared with CohesiveFT to provide access to the internal of the VNS3 Manager remotely as a “one time key”. Once CFT has used the key it can be revoked and access terminated.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address
generate_remote_support_keypair --encrypted-passphrase passphrase-path -o output-path
```

```
$vns3api/vnscubed.rb -K api -S "apidemopassword" -H 174.129.238.73 generate_remote_support_keypair
--encrypted-passphrase /home/myhome/mypassphrase-file -o /home/myhome/remotecred-for-cft
```

Response: There is no printed response to the command line call, there should be a file as named with the “-o” argument.

VNS3 API: Admin

Call: setup_admin_ui
Argument Switch: --enabled (boolean), --username (string), --password (string)
Argument: The “--enabled” arg specifies whether the web UI is enabled. The “--username” arg specifies the username for the web UI. The “--password” arg specifies the password for the web UI.
Allowed Pre-License: No
Purpose: Enables and disables the web UI. Allows the username and password for the web UI to be set.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address setup_admin_ui --enabled true --username newusername --password newpassword
```

```
$vns3api/vnscubed.rb -K api -S "apidemopassword" -H 174.129.238.73 setup_admin_ui --enabled true --username vnscubed -password vnscubed
```

Response:

```
username: vnscubed
enabled: true
```

VNS3 API: Admin

Call: server_action
Argument Switch: --reboot (boolean)
Argument: The “--reboot” arg specifies whether to reboot the VNS3 manager.
Allowed Pre-License: No
Purpose: Re-boots the manager.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address server_action --reboot
true
```

```
$vns3api/vnscubed.rb -K api -S "apidemopassword" -H 174.129.238.73 server_action --reboot true
```

Response:

```
---
status: rebooting
```

VNS3 API: Admin

Call: edit_config
Argument Switch: --topology-name (string)
Argument: “--topology-name” specifies a text name to display at the top of the web ui and in the desc_config API response.
Allowed Pre-License: Yes
Purpose: Provides general information about the manager’s topology, license state and checksums

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address edit_config --topology-name “mytopologyname”
```

$vns3api/vnscubed.rb -K api -S apidemopassword -H 174.129.238.73 edit_config --topology-name “Your Excellent Topology”

Response:

```
---
public_ipaddress: 174.129.238.73
peered: false
licensed: true
vns3_version: 2.9999.14-20120523200216
private_ipaddress: 10.4.117.122
topology_name: Your Excellent Topology
topology_checksum: e5bfef539780d269c3f692132a2df8c51e7557d7
```

VNS3 API: Status

Call: desc_config
Argument Switch: None
Argument: None
Allowed Pre-License: Yes
Purpose: Provides general information about the manager’s topology, license state and checksums

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address desc_config
```

```
$vns3api/vnscubed.rb -K api -S apidemopassword  -H 174.129.238.73 desc_ipsec_status
```

Response:

```
---
public_ipaddress: 174.129.238.73
peered: false
licensed: true
vns3_version: 2.9999.14-20120523200216
private_ipaddress: 10.4.117.122
topology_name: Your Excellent Topology
topology_checksum: e5bfef539780d269c3f692132a2df8c51e7557d7
```

VNS3 API: Status

Call: desc_status
Argument Switch: None
Argument: None
Allowed Pre-License: No, also requires the Peering operations to be completed.
Purpose: Provides general information on all defined IPsec and client overlay connections

command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address desc_status

$vns3api/vnscubed.rb -K api -S apidemopassword -H 174.129.238.73 desc_status

Response: (next page)

VNS3 API: Status

Call: desc_status (continued)
Response: (list of connected clients, info about connected tunnels)

```
---
connected_clients:
  172.31.3.10:
    ipaddress: 63.250.226.147
    managerid: 1
    overlay_ipaddress: 172.31.3.10

ipsec:
  "6":
    tunnel_params:
      phase1_cipher: aes_256
      phase1: up
      nat_t: "on"
      phase2: up
      phase1_dh_group: 2
      dpd: "on"
      phase2_algo: AES_256-HMAC_SHA1
      phase1_prf: sha
    remote_subnet: 192.168.4.0/24
    connected: true
    endpointid: 5
    local_subnet: 172.31.3.0/24
    active: true
connected_subnets:
- - 192.168.4.0
  - 255.255.255.0
```

VNS3 API: Status

Call: desc_ipsec_status
Argument Switch: None
Argument: None
Allowed Pre-License: No
Purpose: Provides information on all defined IPsec connections

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address desc_ipsec_status
```

```
$vns3api/vnscubed.rb -K api -S apidemopassword  -H 174.129.238.73 desc_ipsec_status
```

Response: (next page)

VNS3 API: Status

Call: desc_ipsec_status (continued)
Response:

```
---
"6":
  tunnel_params:
    phase1_cipher: aes_256
    phase1: up
    nat_t: "on"
    phase2: up
    phase1_dh_group: 2
    dpd: "on"
    phase2_algo: AES_256-HMAC_SHA1
    phase1_prf: sha
  remote_subnet: 192.168.4.0/24
  connected: true
  endpointid: 5
  local_subnet: 172.31.3.0/24
  active: true
```

VNS3 API: Status

Call: desc_clients_status
Argument Switch: None
Argument: None
Allowed Pre-License: No
Purpose: Provides information on connected overlay network devices (sometimes referred to as OLNDs)

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address desc_client_status
```

```
$vns3api/vnscubed.rb -K api -S apidemopassword  -H 174.129.238.73 desc_client_status
```

Response:

```
---
172.31.3.10:
  ipaddress: 63.250.226.147
  managerid: 1
  overlay_ipaddress: 172.31.3.10
```

GET RESPONSE VNS3 API: Status

Call: desc_system_status
Argument Switch: None
Argument: None
Allowed Pre-License: No
Purpose: Provides information about the underlying appliance; memory, cpu, disk space, etc..

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address desc_system_status
```

```
$vns3api/vnscubed.rb -K api -S apidemopassword  -H 174.129.238.73 desc_system_status
```

Response:
TBD

CAN ALL ARGS BE CALLED AT ONCE, GET RESPONSE VNS3 API: Status

Call: desc_link_history (Changed)
Argument Switch: --remote (CIDR), --local (CIDR), --tunnel-id (integer)
Argument: "--remote" is an address string in CIDR format to display link history to a remote endpoint, "--local is an address string in CIDR format which will display status of the local route, "--tunnel-id" will display link history of just the tunnel specified, which may be only one tunnel to a remote endpoint.
Allowed Pre-License: No
Purpose: Provides information about the connection history of the subnet or tunnel

command prompt>vnscubed.rb -K "api" -S "myapisecret" -H manager-ip-address desc_link_history --remote cidr_as_string --local cidr_as_string --tunnel-id tunnelnumber

$vns3api/vnscubed.rb -K api -S apidemopassword -H 174.129.238.73 --remote "123.10.10.19/32"

Response:
TBD

VNS3 API: Status

Call: desc_connected_subnets (New)
Argument Switch: --extended-output (boolean)
Argument: “--extended-output” is true to receive verbose information about the connected subnets.
Allowed Pre-License: No
Purpose: Provides information about any connected subnets

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address desc_connected_subnets
--extended-output true
```

```
$vns3api/vnscubed.rb -K api -S apidemopassword  -H 174.129.238.73 --extended-output true
```

Response:

```
---
- netmask: 255.255.255.0
  cidr_mask: "24"
  network: 192.168.4.0
  origin: ipsec
  subnet: 192.168.4.0/24
  managerid: 1
```

VNS3 API: License

Call: upload_license
Argument Switch: --license (input pathname)
Argument: Valid path to a file containing the encrypted license
Allowed Pre-License: Yes
Purpose: License a VNS3 Manager to be a part of a specific topology

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address upload_license --license /pathtolicensefile/license.txt
```

```
$vns3api/vnscubed.rb -K api -S apidemopassword  -H 174.129.238.73 upload_license --license apidemo_ipsectrial_license.txt
```

Response: (next page) It is important to note that the response is not “finalized” meaning you will see default addressing that satisfies the criteria of the license. You can change the subnet, manager addresses, client pack addresses , etc. subsequently with the set_license_parameters call.

Note: *The sample response on the next page provides a response to a more full-featured license than the example license containing the equivalent of the VNS3 Datacenter Connect Trial Edition. The example license is available on the page following the response page. Place it in a plain text file (no rich text) with the name of your choice for use in testing the API call.*

VNS3 API: License

Response:

```
---
capabilities:
- IPsec
- eBGP
- LinearAddressing
- LinearAddressingConfigurable
- CloudWAN
default_topology:
  ipsec_max_endpoints: 50
  overlay_subnet: 172.31.1.0/24
  total_clients: 50
  managers:
  - manager_id: 1
    asn: 65001
    overlay_ipaddress: 172.31.1.1
  - manager_id: 2
    asn: 65002
    overlay_ipaddress: 172.31.1.5
```

VNS3 API: License

Response: (continued)

```
ipsec_max_subnets: 100
  clients:
  - 172.31.1.17
  - 172.31.1.21
  - 172.31.1.25
  - 172.31.1.29
(some clientpacks removed for display purposes)
license: accepted
finalized: false
license_present: true
```

VNS3 API: License

```
-----BEGIN PGP MESSAGE-----
Version: GnuPG v1.4.6 (GNU/Linux)

hQIOA302B0R4m9gZEAgAjMTiUeW81pL53CgKkyYQdQT/PpOo+VDd0+cmSk7gx6Wy
D8oFwqkec/Sm6YHV70oVtiRSP/sWqkHm1aNNvI1hKmZPIYSxZse6jtSZDrnoalG4
FssBe/zQJ0iWqpaWcrL94Gqo9CJlwl/BZ34uQG3Xb45oxPLN5tBdqrOyrQG4uL0y
cyxGI4Qg4K2OP3kk7loLQVmIc+7BCCysSIjYdnHsKDEKZXjURCFln+NuKf4/itxw
hjTB8NpWNgC8W+rEresTmqLG0GaclRUfc/Myw5wRVv07bntv46KmoSC3xcEnWmUs
jwvvE4yoOQI0kJAjPEmtXgFmQBWvVHjLxveGWgNELAf8CW0v6Ruk9I6Cs7LQAcis
oBTN7VrcMyqGzZ5Ql4/JQjTIHjXLRagdHuiPn2ltia9uNA0ljc4jwrpH3AbZeN+F
0/DD6S5HdygWTvck8zHgmgZlGDRZ3aR8eVOh1X2uQZDDrq5mSneyBGv2rKKvcF2o
SH9D6xDUYzxje1o9QVwbhnqIWtFT5rAfVKmVLc3CQcIBKN9vQoLinWsMHdninO1/
kOTUooxqp7msro6QihgKFc4OZzQqBaE/lcr2H5xbP3PzqgS+9BWgXD3NoQqz6aaj
ymEKSS4+9QTbBOfnsmIRwJWEm1ZJSnq7rxXoATfaux4oUyzKMSDyulQmHe8aDNVY
HNLAtQEMXGlWHiHJBvywWv20+YAX+JxMfWUGZ0BiVcx5/z5ftpuHFJ9iNdpzXHYn
G4r+q9DjsYCmnIvyrUEyOj65LaEFKAXY7bpIoXYmDRalCvW8mF6d5oXxXhOzRD+g
2KxwMvg+e0QTrXx5l8nNchzkV9NqnbkrpytKgdus9YZmNCgn716mwyYrKWQK1Wsh
iibxeyPuPjakqv4Jz5jHGStOtqVeb3YiNV+mfyBw6ho1cCr+LG3+5x4K+8RWEFzD
Dp+ECdNsUvGgMTRY34WgtnjizxB+IqaocVQBreng+/hM/+3HpO12ub49k657cEyH
7BGP3GdTPLVbx1KzbS+pm4R7eOHP2L+5v5tKFSxFgUSaXr4Cg+mF7BOL8K2iryez
kT7rV8PYifld+KYrz6wfOZtvYNSif5aQXVx4D+OovjKBSY4ObCz3OGWgGSZbhCwu
AenSCJf2efEA/d1JXMWEga7cWdJkDCR7XFdx3CjW+5MQRgpSjsACpDk=
=E11U
-----END PGP MESSAGE-----
```

VNS3 API: License

Call: upgrade_license
Argument Switch: --license (input pathname)
Argument: Valid path to a file containing the encrypted license
Allowed Pre-License: No
Purpose: Upgrades parts of a VNS3 Manager topology parameters (additional client packs, more tunnels, more endpoints, etc..)

command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address upgrade_license --license /pathtolicensefile/license.txt

$vns3api/vnscubed.rb -K api -S apidemopassword -H 174.129.238.73 upgrade_license --license apidemo_ipsectrial_license.txt

Response:
license: accepted

VNS3 API: License

Call: desc_license
Argument Switch: None
Argument: None
Allowed Pre-License: No
Purpose: Provides information about the manager's license

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address desc_license
```

```
$vns3api/vnscubed.rb -K api -S apidemopassword  -H 174.129.238.73 desc_license
```

Response:

```
---
capabilities:
- IPsec
- eBGP
- LinearAddressing
- LinearAddressingConfigurable
- CloudWAN
topology:
  ipsec_max_endpoints: 50
  overlay_subnet: 172.31.3.0/24
```

VNS3 API: License

Call: desc_license
Response: (continued)

```
total_clients: 47
  managers:
  - manager_id: 1
    asn: 65001
    overlay_ipaddress: 172.31.3.1
  - manager_id: 2
    asn: 65002
ipsec_max_subnets: 100
  clients:
  - 172.31.3.10
  - 172.31.3.11
(some clientpacks removed for display purposes)
uploaded_at: Fri May 25 19:34:25 +0000 2012
sha1_checksum: 9ab6482da85c1f2d5660f5a9e9948dff6835c64d
finalized: true
uploaded_at_i: 1337974465
license_present: true
custom_addressing: true
```

VNS3 API: License

Call: set_license_parameters
Argument Switch: --subnet (CIDR), --managers (multiple address string), --clients (multiple address string), --my-manager-vip (address-string), --default (boolean false)
Argument: The “--subnet” arg specifies the CIDR of the virtual network created for use with the VNS3 Manager. The “--managers” arg specifies a whitespace delimited address string in the subnet to use for the VNS3 Managers’ virtual interfaces. The “--clients” arg specifies a comma delimited, or hyphenated sequence of addresses for use as client addresses in the virtual network. The “--my-manager-vip” address must be allocated from the subnet, and must be the same for all managers (it is used in the internal implementation of VNS3). The “--default” arg should be called with “true” to accept default addressing.
Allowed Pre-License: No
Purpose: Allows customization of the overlay network elements. It also “finalizes” the license, once these settings are made they cannot be changed. To change requires launching a new manager and configuring with different parameters.

command prompt> vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address set_license_parameters --subnet “cidr of overlay network” --managers “manager1address manager2address” --clients “address1, address2, address3” --my-manager-vip “vipaddress”

$vns3api/vnscubed.rb -H 184.72.101.243 -K api -S i-1d15e67a set_license_parameters --subnet "172.31.3.0/24" --managers "172.31.3.1 172.31.3.2" --clients "172.31.3.10-172.31.3.110" --my-manager-vip "172.31.3.254"
Response: (next page)

VNS3 API: License

Call: set_license_parameters

Response:

```
---
parameters:
  my_manager_vip: 172.31.3.254
  subnet: 172.31.3.0/24
  managers:
  - 172.31.3.1
  - 172.31.3.2
  - 172.31.3.3
  - 172.31.3.4
  clients:
  - 172.31.3.10
  - 172.31.3.11
  - 172.31.3.12
  - 172.31.3.13
  (some clientpacks removed for display purposes)
license: accepted
finalized: true
```

VNS3 API: Clientpacks

Call: desc_keyset
Argument Switch: None
Argument: None
Allowed Pre-License: No
Purpose: Returns status of whether cryptographic credentials, which are used to provide overlay devices access to the topology, have been generated.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address desc_keyset
```

```
$vns3api/vnscubed.rb -K api -S apidemopassword  -H 174.129.238.73 desc_keyset
```

Response: (One of three states; there are no keys, they are being generated, they have been generated.)

```
keyset_present: false
in_progress: false
```

```
keyset_present: false
running: 0
in_progress: true
```

```
keyset_present: true
created_at: 2010/08/28 19:25:57 +0000
created_at_i: 1283023557
checksum: 0d61536900908f1b985eae65aa9473a2f312c94c
```

VNS3 API: Clientpacks

Call: setup_keyset
Argument Switch: --source (address-string), --token (string)
Argument: Source Manager with hostname or address as a string, Token is any arbitrary key used to help randomize the checksum, it must be identical for each manager in a topology.
Allowed Pre-License: No
Purpose: Generates cryptographic credentials which are used to provide overlay devices access to the topology.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H another-manager-ip-address setup_keyset --
source manager-ip-address --token "arandomstring”
```

Note: *--source is not required for the first invocation on a manager in the topology, without --source the keys are generated, with source the keys are pulled from one manager to another.*

```
$vns3api/vnscubed.rb -K api -S apidemopassword  -H 174.129.238.73 setup_keyset  --token
“arandomstring”
```

Response:

(see following page)

VNS3 API: Clientpacks

```
$vns3api/vnscubed.rb -K api -S apidemopassword  -H 174.129.238.73 setup_keyset  --token
"arandomstring"
```

```
---
keyset_present: false
running: 0
in_progress: true
started_at_i: 1337976741
started_at: 2012/05/25 20:12:21 +000
```

After several minutes; query again, and the keyset has been generated.

```
$vns3api/vnscubed.rb -K api -S apidemopassword -H 174.129.238.73 desc_keyset
```

```
---
keyset_present: true
created_at: 2012/05/25 20:12:43 +0000
created_at_i: 1337976763
uuid: fc8433e4-a6a5-11e1-9d32-123139268763
checksum: 79ac7639cb044978b2bfef38231fa365d10861ee
```

VNS3 API: Clientpacks

Now invoke the command on ANOTHER overlay manager; and get the keys.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H another-manager-ip-address setup_keyset --
source manager-ip-address --token "arandomstring”
```

NOTE: “-H” is a different machine in this example with our previous host as the --source.

$vns3api/vnscubed.rb -K api -S apidemopassword -H 174.129.238.68 setup_keyset --token “arandomstring” --source 174.129.238.73

Response:

```
---
keyset_present: false
running: 0
in_progress: true
started_at_i: 1337976836
started_at: 2012/05/25 20:13:56 +0000
```

Note: You will see that the hosts from our example both have the same checksum when a query of “desc_keyset” id performed. This means the 2 managers can be peered together in the topology.

VNS3 API: Clientpacks

Call: desc_clientpacks
Argument Switch: --sorted (boolean)
Argument: The “--sorted” arg allows you to specify a true to get the clientpack info back in numerically sorted order.
Allowed Pre-License: No
Purpose: Returns detailed information about all of the clientpacks in the topology. If manager’s are properly peered, this information can come from any of the managers.

command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager1-ip-address desc_clientpacks

$vns3api/vnscubed.rb -K api -S apidemopassword -H 174.129.238.68 desc_clientpacks

Response: (a list of this information for all clientpacks in the topology)

```
name: 172_31_3_56
tarball_file: 172_31_3_56.tar.gz
checked_out: false
tarball_sha1: 23e14ce164a57cf865c488302233043bb7212396
zip_file: 172_31_3_56.zip
zip_sha1: 828fa1390e111282fb574086c534bd25851d2faf
enabled: true
sequential_id: 47
overlay_ipaddress: 172.31.3.56
```

VNS3 API: Clientpacks

Call: fetch_clientpack
Argument Switch: --name (string), --format (zip or tarball), -o (output pathname)
Argument: The “--name” is the name of the clientpack as returned by the “desc_clientpacks” call. The “ --format” has two possible values “tarball” (default) or “zip” which as stated, returns the clientpack in that compression format. The “-o” is an output file name for the clientpack
Allowed Pre-License: No
Purpose: Clientpacks are compressed files with the necessary information and credentials for an overlay client to be connected to the VNS3 topology.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager1-ip-address fetch_clientpack --name 172_31_1_17 --format “zip”  -o “mycredentials.zip”
```

```
$vns3api/vnscubed.rb -K api -S apidemopassword  -H 174.129.238.68 fetch_clientpack  --name “172_31_1_17”  --format “zip” -o “mycredentials.zip”
```

```
$vns3api/vnscubed.rb -K api -S apidemopassword  -H 174.129.238.68 fetch_clientpack  --name “172_31_1_17”  --format “tarball” -o “mycredentials.tar”
```

Response: No success code is printed. Clientpack is output to the requested file name.

VNS3 API: Clientpacks

Call: get_next_available_clientpack
Argument Switch: None
Argument: None
Allowed Pre-License: No
Purpose: For situations where devices need overlay credentials but not for a specific address. This is usually in “autoscale” situations.

command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager1-ip-address get_next_available_clientpack

$vns3api/vnscubed.rb -K api -S apidemopassword -H 174.129.238.68 get_next_available_clientpack

Response: (the info of the next available clientpack for use in a subsequent “fetch_clientpack”)

```
name: 172_31_3_10
tarball_file: 172_31_3_10.tar.gz
checked_out: true
tarball_sha1: 5e2597372e5c96ac07f5a5471f05f5d1a663e53f
zip_file: 172_31_3_10.zip
zip_sha1: 8e14120ed58e3bf5c05d6755eed954adc3f13146
enabled: true
sequential_id: 1
overlay_ipaddress: 172.31.3.10
```

VNS3 API: Clientpacks

Warning: You must use only one manager of the topology with "get_next_available_clientpack"!

VNS3 Managers are not (at this time) able to synchronize information automatically. Using multiple managers could cause distribution of the same client pack to multiple overlay devices which is not allowed. (Multiple clients with the same credentials will continuously knock each other off of the overlay network.)

Note: The "get_next_available_clientpack" call does not actually fetch the credentials. It provides sufficient information for you to the call "fetch_clientpack".

Note: In order for get_next_available_clientpack to not exhaust your pool of clientpacks, devices need to release the clientpack when done using by calling "edit_clientpack" for their clientpack, setting the "--checked_out" property to "false".

VNS3 API: Clientpacks

Call: edit_clientpack
Argument Switch: --name (string), --enabled (boolean), --checked_out (boolean)
Argument: The “--name” switch is the name of a clientpack, Both “--enabled” and “--checked_out” take “true” or “false” as values.
Allowed Pre-License: No
Purpose: For changing properties of clientpacks.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager1-ip-address edit_clientpack --name clientpackname --enabled true --checked_out false
```

```
$vns3api/vnscubed.rb -K api -S apidemopassword  -H 174.129.238.68 edit_clientpack --name “172_31_3_10” --enabled true  --checked_out false
```

Response: (returns the clientpack info for the named clientpack)

```
---
name: 172_31_3_10
tarball_file: 172_31_3_10.tar.gz
checked_out: false
tarball_sha1: 5e2597372e5c96ac07f5a5471f05f5d1a663e53f
zip_file: 172_31_3_10.zip
zip_sha1: 8e14120ed58e3bf5c05d6755eed954adc3f13146
enabled: true
sequential_id: 1
overlay_ipaddress: 172.31.3.10
```

VNS3 API: Clientpacks

Call: reset_client (New)
Argument Switch: --name (string),
Argument: The “--name” switch is the name of a clientpack,
Allowed Pre-License: No
Purpose: For resetting the connection of a client to a VNS3 Manager.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager1-ip-address reset_client --name clientpack-name
```

```
$vns3api/vnscubed.rb -K api -S apidemopassword  -H 174.129.238.68 reset_client --name “172_31_0_10”
```

Response:

```
---
overlay_ipaddress: 172.31.3.10
disconnecting: 172_31_3_10
```

VNS3 API: Clientpacks

Call: reset_all_clients (New)
Argument Switch: None
Argument:
Allowed Pre-License: No
Purpose: For resetting all of the connections of clients connected to the VNS3 Manager.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager1-ip-address reset_all_clients
```

```
$vns3api/vnscubed.rb -K api -S apidemopassword  -H 174.129.238.68 reset_all_clients
```

Response:

```
---
resetting: []
```

VNS3 API: Snapshots

Call: create_snapshot
Argument Switch: None
Argument: None
Allowed Pre-License: No
Purpose: Create a snapshot file on that manager being queried. The snapshot is named based on date, time and IP address of the manager.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address create_snapshot
```

```
$vns3api/vnscubed.rb -K api -S apidemopassword  -H 174.129.238.68 create_snapshot
```

Response:

```
---
snapshot_20120525_1337978471_50.17.119.158:
  size: 840098
  created_at: 2012/05/25 20:41:12 +0000
  created_at_i: 1337978472
  sha1_checksum: b4fc625d068c2936ef4f480c0bc09e0594f6660f
```

Note: Snapshots have a 1-to-1 relationship with managers in a topology. To recover an “N” manager topology from snapshots requires “N” distinct snapshot files.

VNS3 API: Snapshots

Call: fetch_snapshot
Argument Switch: --name (string), -o (output pathname)
Argument: The "--name" switch is the name of the snapshot desired. "The "-o" switch is for the name of the output file you want for the snapshot.
Allowed Pre-License: No
Purpose: Download a snapshot file for later use.

command prompt>vnscubed.rb -K "api" -S "myapisecret" -H manager-ip-address fetch_snapshot --name snapshot_as_string -o mysnapshotfile

$vns3api/vnscubed.rb -K api -S apidemopassword -H 174.129.238.68 fetch_snapshot --name snapshot_20100830_1283176087_174.129.238.68 -o m1_snapshot -o MyManager.snap

Response: The requested snapshot file is downloaded to the file specified by the "-o" switch.

VNS3 API: Snapshots

Call: delete_snapshot
Argument Switch: --name (string)
Argument: The “--name” switch is the name of the snapshot desired.
Allowed Pre-License: No
Purpose: Delete the named snapshot from the manager.

command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address delete_snapshot --name snapshot_as_string

$vns3api/vnscubed.rb -K api -S apidemopassword -H 174.129.238.68 delete_snapshot --name snapshot_20100830_1283176087_174.129.238.68

Response: (the call returns the list of remaining snapshots, in this case an empty list)
{}

VNS3 API: Snapshots

Call: import_snapshot
Argument Switch: --snapshot (input pathname)
Argument: The “--snapshot” switch is the file containing the snapshot you wish to import.
Allowed Pre-License: Yes
Purpose: Imports the snapshot into the manager and triggers a reboot for the configuration to take effect.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address import_snapshot --snapshot filename_of_snapshot
```

```
$vns3api/vnscubed.rb -K api -S apidemopassword  -H 174.129.238.68 import_snapshot --snapshot m1_snapshot
```

Response:

```
snapshot: accepted
```

VNS3 API: Snapshots

Call: desc_snapshots
Argument Switch: None
Argument: None
Allowed Pre-License: No
Purpose: Shows snapshots that have been taken on that manager being queried.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address desc_snapshots
```

```
$vns3api/vnscubed.rb -K api -S apidemopassword  -H 174.129.238.68 desc_snapshots
```

Response:

```
snapshots:
  snapshot_20100830_1283176087_174.129.238.68:
    size: 912974
    created_at: 2010/08/30 13:48:08 +0000
    created_at_i: 1283176088
    sha1_checksum: 6b1ddca58d454022ca804c31fc016447613df218
latest_snapshot: snapshot_20100830_1283176087_174.129.238.68
```

Note: This response shows either an empty list token “{}” or returns a list of snapshots with the token “latest_snapshot” indicating the most recent snapshot created.

VNS3 API: Manager Peering

Call: desc_peering
Argument Switch: None
Argument: None
Allowed Pre-License: No
Purpose: Provides the status of whether a manager is peered to other managers.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address desc_peering
```

Note: In the order of operations performed, there are 2 managers; at 174.129.238.68 and 174.129.238.73. Neither of them has been peered so this operation will return false until add_peer has been called on them. Prior to add_peer the managers must be given “IDs” for their manager number in the topology via the set_manager_id call.

```
$vns3api/vnscubed.rb -K api -S apidemopassword  -H 174.129.238.73 desc_peering
```

Response:
TBD

VNS3 API: Manager Peering

Call: set_manager_id
Argument Switch: --id (integer)
Argument: The manager ID as an integer, cannot be the same as the id of another manager in the topology, and cannot be greater than the number of managers in the topology.
Allowed Pre-License: No
Purpose: Sets the Manager ID of a manager so that it can be peered within a topology.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address set_manager_id --id 1
```

Note: We will set the manager ID of our example host 174.129.238.68 to “1” and our example host 174.129.238.73 to “2”. They will then have the same license, keysets, checksum, and proper manager ids, meaning they can then subsequently be “peered”.

```
$vns3api/vnscubed.rb -K api -S apidemopassword  -H 174.129.238.68 set_manager_id --id 1
```

```
Response:
id: 1
managers:
  "1":
    self: true
    id: 1
```

VNS3 API: Manager Peering

```
$vns3api/vnscubed.rb -K api -S apidemopassword  -H 174.129.238.73 set_manager_id --id 2
```

Response:

```
peered: true
id: 2
managers:
  "1":
    not_set: true
    id: 1
  "2":
    self: true
    id: 2
```

VNS3 API: Manager Peering

Call: add_peer
Argument Switch: --id (integer), --name (string)
Argument: The “--id” is the manager ID as an integer of the the manager you are peering with, NOT the id of the manager you are calling “add_peer” on. The “--name” is the IP address or host name of the manager you are peering with.
Allowed Pre-License: No
Purpose: Creates a peering relationship from a manager to another manager. The peering call is unidirectional. Reciprocal calls must be made to peer two managers together.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager1-ip-address add_peer --id 2 --name manager2-ip-address
```

Note: The manager ID of our example host 174.129.238.68 is “1” and our example host 174.129.238.73 is “2”.

```
$vns3api/vnscubed.rb -K api -S apidemopassword  -H 174.129.238.68 add_peer --id 2 --name 174.129.238.73
```

Response: (next page)

VNS3 API: Manager Peering

Response:

```
peered: true
id: 1
managers:
  "1":
    self: true
    id: 1
  "2":
    address: xxx.xxx.xxx.xxx.compute-1.amazonaws.com
    reachable: false
    id: 2
```

Now we can do reciprocal manager peering calls to establish the mesh connection between the managers.

$vns3api/vnscubed.rb -K api -S apidemopassword -H 174.129.238.73 add_peer --id 1 --name 174.129.238.68

NOTE: Sometimes “reachable” will say :false for a short period of time before the managers finish their connection process. If it says :false for a long period of time, there is an error.

VNS3 API: Manager Peering

Call: delete_peer
Argument Switch: --id (integer)
Argument: The “--id” is the manager ID as an integer of the the manager you want to break your peering relationship with.
Allowed Pre-License: No
Purpose: Breaks a peering relationship from a manager to another manager. The peering call is unidirectional. Reciprocal calls must be made to fully break the peer relationship.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager1-ip-address delete_peer --id 2
```

Note: The manager ID of our example host 174.129.238.68 is “1” and our example host 174.129.238.73 is “2”. When the delete_peer call is made we are returned to the state prior to the add_peer call.

$vns3api/vnscubed.rb -K api -S apidemopassword -H 174.129.238.68 delete_peer --id 2

Response:

```
peered: true
id: 1
managers:
  "1":
    self: true
    id: 1
  "2":
    not_set: true
    id: 2....
```

VNS3 API: IPsec

Note: IPsec configuration from the command line is the most complicated of the calling sequences covered in this document. It requires knowledge of IPsec connectivity.

In order to show the commands a pre-existing network configuration will be connected to via IPsec. We have used an IPSec device running at 63.250.226.147 to connect to. It is running with NAT-Traversal enabled and uses the preshared key “testing”. It has Perfect Forward Secrecy Enabled and is looking for “aes256” encryption for both Phase 1 and Phase 2 negotiations. It is looking for “sha1” hash method for Phase 1 and Phase 2 negotiations. It uses Diffie Hellman Group 5 when needed.

Our approach with the API will be:
- Create an IPsec endpoint and then interrogate it and manipulate it with the other calls.

VNS3 API: IPsec

Call: create_ipsec_endpoint
Argument Switch: --name (string), --ipaddress (address-string), --secret (string), --pfs (boolean), --extra_config (string), --private_ipaddress (address-string)
Argument: The “--name” arg is a user-supplied name for the connection which will show up as a label in the web UI. The “--ipaddress” arg is the IP Address of the remote gateway. The “--secret” arg is the pre-shared key for the connection. The “--pfs” enables Perfect Forward Secrecy if true, disables if false. The “--extra_config” arg is a string with additional options for the connection. The “--private_ipaddress” arg is the internal NAT address of the remote gateway.
Allowed Pre-License: No
Purpose: Create IPsec connection to the defined remote gateway.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address create_ipsec_endpoint --name “IPsec Connection” --ipaddress remote-gateway-address --secret preshared-key --pfs true --extra_config  config-string
```

```
$vpn3api/vpncubed.rb -K $common_api_key -S $common_api_secret -H $manager1 create_ipsec_endpoint --name "API_Created_Endpoint"  --ipaddress 63.250.226.147 --secret "testing" --pfs true --extra-config "phase1=aes256-sha1-dh5 phase2=aes256-sha1"
```

VNS3 API: IPsec

Call: create_ipsec_endpoint (continued)

Response:

```
---
name: API_Created_Endpoint
ipaddress: 63.250.226.147
tunnels: {}

id: 4
extra_config:
- phase1=aes256-sha1-dh5
- phase2=aes256-sha1
pfs: true
bgp_peers: {}
```

Note: This defined the remote gateway endpoint. In order to create a live tunnel a tunnel definition needs to be added to this endpoint.

VNS3 API: IPsec

Call: desc_ipsec
Argument Switch: None
Argument: None
Allowed Pre-License: No
Purpose: Returns information about all IPsec endpoints and subnets defined.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address desc_ipsec
```

```
$vns3api/vnscubed.rb -K api -S "apidemopassword" -H 50.17.119.158 desc_ipsec
```

Response: (next page)

VNS3 API: IPsec

Call: desc_ipsec (continued)

Response:

```
this_endpoint:
overlay_subnet: 172.31.3.0/24
  ipaddress: 50.17.119.158
  nat_traversal: true
  asn: 65001
  private_ipaddress: 50.17.119.158
  ipsec_local_ipaddress: 50.17.119.158
remote_endpoints:
  "4":
    name: API_Created_Endpoint
    ipaddress: 63.250.226.147
    tunnels: {}
  id: 4
    extra_config:
    - phase1=aes256-sha1-dh5
    - phase2=aes256-sha1
    pfs: true
    bgp_peers: {}
```

VNS3 API: IPsec

Call: edit_ipsec_config
Argument Switch: --nat-traversal (boolean), --ipsec-local-ipaddress (address-string)
Argument: "--nat-traversal is true to enable nat-traversal, false to disable. This is for all connections on the VNS3 Manager, i.e. a device wide setting. "--ipsec_local_ipaddress effectively a "cloud NAT" address, since you don't know what your LAN address will be between invocations in a cloud, this address can be used by remote endpoints as your "behind a NAT" address if needed (Watchguard Firebox for example).
Allowed Pre-License: No
Purpose: Allows the setting of 2 device wide parameters; one for NAT Traversal and one for local LAN address. NOTE: These are device wide and must be set before any remote endpoint definitions are created. If they need to be changed, all remote endpoint information and tunnel information must be deleted, these settings changed, and then re-create remote endpoints and tunnels.

command prompt>vnscubed.rb -K "api" -S "myapisecret" -H manager-ip-address edit_ipsec_config --nat-traversal true --ipsec-local-ipaddress vpn3-manager-local-address

$vpn3api/vpncubed.rb -K $common_api_key -S $common_api_secret -H $manager1 edit_ipsec_config --nat-traversal true --ipsec-local-ipaddress 50.17.119.158

Response: (next page)

VNS3 API: IPsec

Call: edit_ipsec_config (continued)
Response:

```
---
this_endpoint:
  overlay_subnet: 172.31.3.0/24
  ipaddress: 50.17.119.158
  nat_traversal: true
  asn: 65001
  private_ipaddress: 50.17.119.158
  ipsec_local_ipaddress: 50.17.119.158
remote_endpoints: {}
```

Note: This call can only be done before there are any remote endpoint defined. It is device wide across all endpoints.

Note: In this example we used edit_ipsec_endpoint to add a property we didn't set the first time; the NAT address of the VPN3 Manager.

VNS3 API: IPsec

Call: edit_ipsec_endpoint
Argument Switch: --endpointid (integer), --name (string), --ipaddress (address-string), --secret (string), --pfs (boolean), --extra_config (string), --private_ipaddress (address-string)
Argument: The "--endpointid" arg is the id number of the specific endpoint which is being edited. The "--name" arg is a user-supplied name for the connection which will show up as a label in the web UI. The "--ipaddress" arg is the IP Address of the remote gateway. The "--secret" arg is the pre-shared key for the connection. The "--pfs" enables Perfect Forward Secrecy if true, disables if false. The "--extra_config" arg is a string with additional options for the connection. The "--private_ipaddress" arg is the internal NAT address of the remote gateway (to unset or clear --private-ipaddress use "0.0.0.0" as its value.
Allowed Pre-License: No
Purpose: Edit IPsec connection to the defined remote gateway.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address
edit_ipsec_endpoint --endpointid 1 --private-ipaddress gateway_nat_address
```

```
$vpn3api/vpncubed.rb -K $common_api_key -S $common_api_secret -H $manager1 edit_ipsec_endpoint --
endpoint 4  --ipaddress 63.250.226.147 --secret "testing" --pfs true --extra-config "phase1=aes256-sha1-
dh5 phase2=aes256-sha1 dpdaction=restart dpdtimeout=90s dpddelay=30s” --private-ipddress
192.168.1.30
```

Response: (next page)

VNS3 API: IPsec

Call: edit_ipsec_endpoint (continued)
Response:

```
name: API_Created_Endpoint
ipaddress: 63.250.226.147
tunnels: {}
id: 4
extra_config:
- phase1=aes256-sha1-dh5
- phase2=aes256-sha1
- dpdaction=restart
- dpdtimeout=90s
- dpddelay=30s
pfs: true
private_ipaddress: 192.168.1.30
bgp_peers: {}
```

Note: In this example we used edit_ipsec_endpoint to add a property we didn't set the first time; the NAT address of the remote gateway.

VNS3 API: IPsec

Call: setup_ipsec
Argument Switch: --restart (boolean)
Argument: The “--restart” arg restarts the VNS3 manager’s IPsec subsystem when the value is “true”. The value of “false” has no value.
Allowed Pre-License: No
Purpose: Does a complete restart of the IPsec subsystem, resetting all IPsec tunnels.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address setup_ipsec  --restart true
```

```
$vns3api/vnscubed.rb -K api -S "apidemopassword" -H 50.17.119.158 setup_ipsec --restart true
```

Response:

```
restart: true
```

VNS3 API: IPsec

Call: get_ipsec_local_ipaddress (DEPRECATED, please use desc_ipsec)
Argument Switch: None
Argument: None
Allowed Pre-License: No
Purpose: Because VNS3 managers run as virtual instances in virtual infrastructure - there is a possibility that an instance will be re-launched, ending up with a different internal NAT address than it originally had, which would require all connected devices to change their information. As a result VNS3 has a “fake” NAT which is used so that it can remain constant across launches, meaning no reconfiguration by peer gateways.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address
get_ipsec_local_ipaddress
```

```
$vns3api/vnscubed.rb -K api -S "apidemopassword" -H 50.17.119.158 get_ipsec_local_address
```

Response:

```
ipsec_local_ipaddress: 192.0.2.254
```

VNS3 API: IPsec

Call: set_ipsec_local_ipaddress (DEPRECATED, please use edit_ipsec_config)
Argument Switch: None
Argument: None
Allowed Pre-License: No
Purpose: Because VNS3 managers run as virtual instances in virtual infrastructure - there is a possibility that an instance will be re-launched, ending up with a different internal NAT address than it originally had, which would require all connected devices to change their information. As a result VNS3 has a “fake” NAT which is used so that it can remain constant across launches, meaning no reconfiguration by peer gateways.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address
set_ipsec_local_ipaddress --ipaddress my_new_persistent_NAT_address
```

```
$vns3api/vnscubed.rb -K api -S "apidemopassword" -H 50.17.119.158 set_ipsec_local_address --ipaddress
192.0.8.200
```

Response:
Error: OperationNotAllowedError: Detected active IPsec configuration.

Note: This call will not work if there are any defined IPsec subnets or Remote Endpoints. In this case since we defined a remote endpoint; an error it returned.

VNS3 API: IPsec

Call: desc_ipsec_endpoint
Argument Switch: --endpointid (integer)
Argument: The "--endpointid" arg specifies which of the defined remote endpoints for which detailed information is desired.
Allowed Pre-License: No
Purpose: Returns detailed information about the IPsec endpoint specified.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address desc_ipsec_endpoint --endpointid integer_of_one_of_the_endpoints
```

```
$vns3api/vnscubed.rb -K api -S "apidemopassword" -H 50.17.119.158 desc_ipsec_endpoint --endpointid 1
```

Response:
The same information is returned for this single IPsec endpoint specified - identical to the information returned via desc_ipsec.

VNS3 API: IPsec

Call: delete_ipsec_endpoint
Argument Switch: --endpointid (integer)
Argument: The “--endpointid” arg specifies which of the defined remote endpoints is to be deleted.
Allowed Pre-License: No
Purpose: Deletes the IPsec endpoint specified.

command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address delete_ipsec_endpoint --endpointid integer_of_one_of_the_endpoints

$vns3api/vnscubed.rb -K api -S "apidemopassword" -H 50.17.119.158 delete_ipsec_endpoint --endpointid 1

Response:
The same information is returned for this single IPsec endpoint specified - identical to the information returned via desc_ipsec.

VNS3 API: IPsec

Call: create_ipsec_tunnel
Argument Switch: --endpoint (integer), --remote-subnet (CIDR), --local-subnet (CIDR), --ping-ipaddress (address-string), --ping-interval (integer), --description (string)
Argument: The “--endpoint” arg specifies which of the defined remote endpoints the tunnel is to be created for. The “--remote-subnet” arg is the subnet desired from the remote endpoint. The “--local-subnet” arg specifies the local address space to offer up to the remote endpoint for communicating with the address space defined by “--remote-subnet”. The “--ping-ipaddress” arg is the IP address of a host in the “--remote-subnet” address space that will have a ping sent to it every “--ping-interval”. (Note: This ping goes through the tunnel, not outside the tunnel.) The “--description” arg is a description of the tunnel.
Allowed Pre-License: No
Purpose: Creates an ipsec tunnel description offering up a local subnet specification for talking to a remote subnet on the other side of a remote endpoint/gateway. Optionally allows the creation of a “Ping Host”; a host address on the remote subnet side which is pinged (ICMP ping) every N seconds. This keeps traffic moving through the tunnel, and in some interoperability cases prevents tunnel lapses.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address create_ipsec_tunnel --endpoint integer_of_one_of_the_endpoints --remote-cubnet cidr_string --local-subnet cidr_string --ping-ipaddress address_on_remote_subnet --ping-interval seconds_as_integer
```

```
$vpn3api/vpncubed.rb -K $common_api_key -S $common_api_secret -H $manager1 create_ipsec_tunnel --endpoint 5 --remote-subnet "192.168.4.0/24" --local-subnet "172.31.3.0/24" --ping-ipaddress 192.168.4.1 --ping-interval 60
```

VNS3 API: IPsec

Call: create_ipsec_tunnel (continued)
Response:

```
---
name: API_Created_Endpoint
ipaddress: 63.250.226.147
tunnels:
  "5":
    remote_subnet: 192.168.4.0/24
    ping_ipaddress: 192.168.4.1
    id: 5
    description:
    ping_interval: 60
    local_subnet: 172.31.3.0/24
id: 5
extra_config:
- phase1=aes256-sha1-dh5
- phase2=aes256-sha1
- dpdaction=restart
- dpdtimeout=90s
- dpddelay=30s
pfs: true
private_ipaddress: 192.168.1.30
bgp_peers: {}
```

VNS3 API: IPsec

Call: edit_ipsec_tunnel
Argument Switch: --endpoint (integer), --ping-address (address-string), --ping-interval (integer), --description (string), --bounce (boolean)
Argument: The “--endpoint” arg specifies which of the defined remote endpoints the tunnel is to be created for. The “--remote-subnet” arg is the subnet desired from the remote endpoint. The “--ping-address” arg is the IP address of a host in the “--remote-subnet” address space that will have a ping sent to it every “--ping-interval”.
Allowed Pre-License: No
Purpose: Allows edit of a “Ping Host”; a host address on the remote subnet side which is pinged (ICMP ping) every N seconds. This keeps traffic moving through the tunnel, and in some interoperability cases prevents tunnel lapses. Allows edit of description and adds a tunnel reset capability. The “--remote-subnet” and “--local-subnet” args from create_ipsec_tunnel cannot be edited, the tunnel must be deleted and re-created. The “--bounce” arg, when “true” resets the IPsec connection for this specfic tunnel.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address edit_ipsec_tunnel --endpoint integer_of_one_of_the_endpoints  --ping-address address_on_remote_subnet --ping-interval seconds_as_integer --bounce true
```

```
$vpn3api/vpncubed.rb -K $common_api_key -S $common_api_secret -H $manager1 edit_ipsec_tunnel --endpoint 5 --tunnelid 5  --ping-ipaddress 192.168.4.1 --ping-interval 60 --bounce true
```

Call: edit_ipsec_tunnel (continued)
Response:

```
---
remote_subnet: 192.168.4.0/24
ping_ipaddress: 192.168.4.1
id: 5
description:
ping_interval: 60
bounce: true
local_subnet: 172.31.3.0/24
```

VNS3 API: IPsec

Call: delete_ipsec_tunnel
Argument Switch: --endpoint (integer), --tunnelid (integer)
Argument: The “--endpoint” arg specifies which of the defined remote endpoints the tunnel is to be deleted for. The “--tunnelid” arg is the tunnel id of the specific tunnel for the specified endpoint to be deleted.
Allowed Pre-License: No
Purpose: Delete an IPsec tunnel from a specified endpoint.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address delete_ipsec_tunnel --endpoint integer_of_one_of_the_endpoints --tunnelid integer_of_one_of_the_tunnels
```

$vpn3api/vpncubed.rb -K $common_api_key -S $common_api_secret -H $manager1 delete_ipsec_tunnel --endpoint 5 --tunnelid 5

Response: (next page)

VNS3 API: IPsec

Call: delete_ipsec_tunnel

Response:
```
---
name: API_Created_Endpoint
ipaddress: 50.17.119.158
tunnels: {}
id: 5
extra_config:
- phase1=aes256-sha1-dh5
- phase2=aes256-sha1
- dpdaction=restart
- dpdtimeout=90s
- dpddelay=30s
pfs: true
private_ipaddress: 192.168.1.30
bgp_peers: {}
```

VNS3 API: IPsec

Call: create_ebgp_peer
Argument Switch: --endpoint (integer), --ipaddress (address-string), --asn (integer)
Argument: The “--endpoint” arg specifies which of the defined remote endpoints the eBGP peer is to be created for. The “--ipaddress” arg is the IP address of the desired BGP peer. The “--asn” argument is the “autonomous system number” assigned to the device at “--ipaddress”.
Allowed Pre-License: No
Purpose: Create a BGP peer connection.

command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address create_ebgp_peer --endpoint integer_of_one_of_the_endpoints --ipaddress ipaddress_of_bgppeer --asn asn_number

$vns3api/vnscubed.rb -K api -S "apidemopassword" -H 174.129.238.73 create_ebgp_peer --endpoint 1 --address 10.10.1.14 --asn 65321

Response:

VNS3 API: IPsec

Call: delete_ebgp_peer
Argument Switch: --endpoint (integer), --ebgppeerid (integer)
Argument: The “--endpoint” arg specifies which of the defined remote endpoints the eBGP peer is to be deleted for. The “--ebgppeerid” arg is the id of the BGP Peer gotten from the create command or via a desc_ipsec command.
Allowed Pre-License: No
Purpose: Deletes a BGP peer connection.

command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address delete_ebgp_peer --endpoint integer_of_one_of_the_endpoints --ebgppeerid integer_of_one_of_peers

$vns3api/vnscubed.rb -K api -S "apidemopassword" -H 174.129.238.73 delete_ebgp_peer --endpoint 1 --ebgppeerid 1

Response:

VNS3 API: IPsec

Call: create_remote_subnet (Removed from API, please use create_ipsec_tunnel)
Argument Switch: --endpointid (integer), --subnet (CIDR)
Argument: The “--endpointid” arg specifies the remote endpoints for which a subnet tunnel is to be created. The subnet is a CIDR specification for the subnet at the remote gateway.
Allowed Pre-License: No
Purpose: Creates an IPsec tunnel to the remote gateway for communicating with the subnet specified.

command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address create_remote_subnet --endpointid integer_of_one_of_the_endpoints --subnet cidr_spec

$vns3api/vnscubed.rb -K api -S "apidemopassword" -H 174.129.238.73 create_remote_subnet --endpointid 1 --subnet 192.168.1.0/24

Response:

```
remote_subnets:
  "2":
    id: 2
    subnet: 192.168.1.0/24
```

Note: The endpoint info is returned along with a “remote_subnets:” section. The endpoint IDs and remote subnets share the same numbering sequence. In this case with the endpoint having an ID of “1”, the next endpoint or subnet gets an id of “2”, as seen in this response.

VNS3 API: IPsec

Call: delete_remote_subnet (Removed from API, please use delete_ipsec_tunnel)
Argument Switch: --endpointid (integer), --subnetid (integer)
Argument: The “--endpointid” arg specifies the remote endpoint for which a subnet tunnel is to be deleted. The “--subnetid” arg specifies a subnet which must be associated with the endpoint chosen.
Allowed Pre-License: No
Purpose: Removes an IPsec tunnel to the remote gateway as specified by the subnet id.

command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address delete_remote_subnet --endpointid integer_of_one_of_the_endpoints --subnetid subnet_id

$vns3api/vnscubed.rb -K api -S "apidemopassword" -H 174.129.238.73 delete_remote_subnet --endpointid 1 --subnetid 2

Response:
The endpoint info is returned.

VNS3 API: Firewall

Note: The VNS3 Network firewall settings are distinct to the VNS3 manager the rules are set on. Due to the flexibility of software defined networking, it is not a “given” that all firewall rules put on one manager in a topology should be on all managers in the topology. If you have a multi-manager topology and want to have identical rules throughout the virtual network please use the API or the Web UI to copy the same rule set to all managers.

VNS3 API: Firewall

Call: desc_firewall
Argument Switch: None
Argument: None
Allowed Pre-License: No
Purpose: Describes any firewall setting on the VNS3 Manager.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address desc_firewall
```

```
$vns3api/vnscubed.rb -K api -S "apidemopassword" -H 174.129.238.73 desc_firewall
```

The response means “this rule at position 0” in the rule table.

Response:

```
---
- - -s 192.168.11.5/32 -j ACCEPT
  - 0
```

VNS3 API: Firewall

Call: add_firewall_rule
Argument Switch: --rule (string), --position (positive or negative integer)
Argument: The “--rule” arg is a string to add to the firewall at position --position in the list of rules. It is a string that needs to be compatible with a Linux “iptables” statement. The “--position” arg is the position which the rule will be inserted in the list of Firewall rules. The default is “-1”, meaning at the top of the rule set.
Allowed Pre-License: No
Purpose: Adds a firewall rule to the VNS3 Manager’s firewall.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address add_firewall-rule --
position -1 --rule iptables_string
```

```
$vns3api/vnscubed.rb -K api -S "apidemopassword" -H 174.129.238.73 add_firewall_rule --position -1 “-s
192.168.11.5/32” -j ACCEPT”
```

Note: This rule may not be relevant to the rest of the topology parameters and is just for example purposes. The response means “this rule at position 0” in the rule table.

Response:

```
---
- - -s 192.168.11.5/32 -j ACCEPT
  - 0
```

VNS3 API: Firewall

Call: delete_firewall_rule
Argument Switch: --position (positive integer)
Argument: The “--position” arg is the position which the rule will be deleted in the list of Firewall rules.
Allowed Pre-License: No
Purpose: Removes a firewall rule from the VNS3 Manager’s firewall.

command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address delete_firewall-rule --position position_in_rule_list

$vns3api/vnscubed.rb -K api -S "apidemopassword" -H 174.129.238.73 delete_firewall_rule --position 0

Response:

```
---
rules: []
deleted: -s 192.168.11.5/32 -j ACCEPT
```

VNS3 API: Routes

Call: desc_routes
Argument Switch: None
Argument: None
Allowed Pre-License: No
Purpose: Describes routes that this manager has access to via its network interfaces (virtual or otherwise) that it wants to publish to overlay network devices. Other VPN3 Managers will receive the route essentially instantly. Network clients will receive it when they get their next route push, which is normally on a re-connect or in neartime if they use the VNS3 routing agent on their cloud servers. Remote endpoints (other data centers) would not receive the route unless specified as part of their IPsec configuration AND the configuration of such a tunnel on the VNS3 manager.

```
command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address desc_routes
```

$vpn3api/vpncubed.rb -K $common_api_key -S $common_api_secret -H $manager1 desc_routes

Response:
```
--- {}
```

VNS3 API: Routes

Call: add_route
Argument Switch: --cidr (CIDR)
Argument: The “--cidr” argument is the CIDR of a route that the VNS3 Manager has access to that it wants to publish throughout the routing tables of the overlay network.
Allowed Pre-License: No
Purpose: Pushes routes that this manager has access to via its network interfaces (virtual or otherwise) that it wants to publish to overlay network devices.

command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address add_route --cidr cidr-string-of-interface-i-am-connected-to

$vpn3api/vpncubed.rb -K $common_api_key -S $common_api_secret -H $manager1 add_route --cidr "10.1.10.10/32"

Response:
```
10.1.10.10/32:
  netmask: 255.255.255.255
  id: 167840266-32
  cidr: 10.1.10.10/32
```

VNS3 API: Routes

Call: delete_route
Argument Switch: --id (id from desc_route or add_route response)
Argument: The “--id” argument is the id number of the route as returned in the add_route response, or from the desc_routes response.
Allowed Pre-License: No
Purpose: Deletes routes that this manager has access to via its network interfaces (virtual or otherwise) but that it wants to “un-publish” to overlay network devices.

command prompt>vnscubed.rb -K “api” -S “myapisecret” -H manager-ip-address delete_route --id route-id

$vpn3api/vpncubed.rb -K $common_api_key -S $common_api_secret -H $manager1 delete_route --id 167840266-32

Response:
```
--- {}
```

Troubleshooting and FAQ for the VNS3 API

Issue appears to be "hopping" on and off the network. This is usually the result of the same client keys being installed on two client machines in the network. Only one client machine can use a set of credentials at a given time.

Fetch Keyset appears to hang or not work. Check to see if the Amazon security group is correct for port 8000 between the manager you are getting the keyset from and the manager you are do the fetch from. If they are separated across Amazon USA and Amazon EU you will need to have thier security group reference the public IP addresses. When you do the "Fetch Keyset" command use the managers public IP address.

Manager IDs seem correct, EC2 security groups seem correct, but managers, especially ones launched via separate launch commands will not "peer". Review your worksheet and your launch commands. Ensure that the managers were all launched with the same alphanumeric string.

End