

Netris docs Documentation

Release Netris v2.9

© 2021, Netris

August 23, 2021

Table of Contents

1 Concepts	3
1.1 Introduction to Netris	3
1.2 What is Netris Controller	3
1.3 Netris Switch Agent	4
1.4 Netris SoftGate	4
2 On-prem Netris Controller installation	7
2.1 KVM virtual machine	7
2.2 Installation steps for KVM hypervisor	7
2.3 Netris Controller Installation steps	7
2.4 Accessing the Netris Controller	8
2.5 Security hardening	9
3 Netris Controller Helm Chart	11
3.1 Prerequisites	11
3.2 Get Repo Info	11
3.3 Installing the Chart	11
3.4 Uninstalling the Chart	12
3.5 Configuration	12
3.6 Usage	23
4 Controller initial configuration	25
4.1 Definitions	25
4.2 Subnets	25
4.3 Inventory Profiles	27
4.4 Adding Switches to Topology	28
4.5 Topology Manager	29
4.6 Hairpin (Cumulus only)	30
4.7 Adding SoftGate nodes to Topology	32
5 Netris switch agent installation	35
5.1 For Cumulus Linux	35
5.2 For Ubuntu SwitchDev	37
6 Netris SoftGate agent installation	41
6.1 Minimal hardware requirements	41
6.2 BIOS configuration	41
6.3 Software installation	41
7 Basic BGP	45

8 Advanced BGP	47
8.1 BGP objects	47
8.2 BGP route-maps	49
9 Routes (static routing)	51
10 NAT	53
10.1 Enabling NAT	53
10.2 Defining NAT rules	54
11 Looking Glass	59
12 V-NET	63
13 Kubenet	67
14 ROH (Routing on the Host)	69
15 L3 Load Balancer (Anycast LB)	71
16 L4 Load Balancer (L4LB)	73
16.1 Enabling L4LB service	73
16.2 Consuming L4LB service	74
17 Access Control Lists (ACL)	77
17.1 ACL Default Policy	77
17.2 ACL rules	78
17.3 ACL approval workflow	80
17.4 The sequence order of ACL rules	81
18 Visibility (Telescope)	83
18.1 Graph Boards	83
18.2 API Logs	85
18.3 Dashboard	85
19 Accounts	87
19.1 Users	87
19.2 Tenants	88
19.3 Permission Groups	89
19.4 User Roles	89
20 Release notes	91

Netris is the automatic NetOps platform that runs the physical network and provides cloud-like user experience for NetOps and DevOps engineers.

Contents:

Concepts

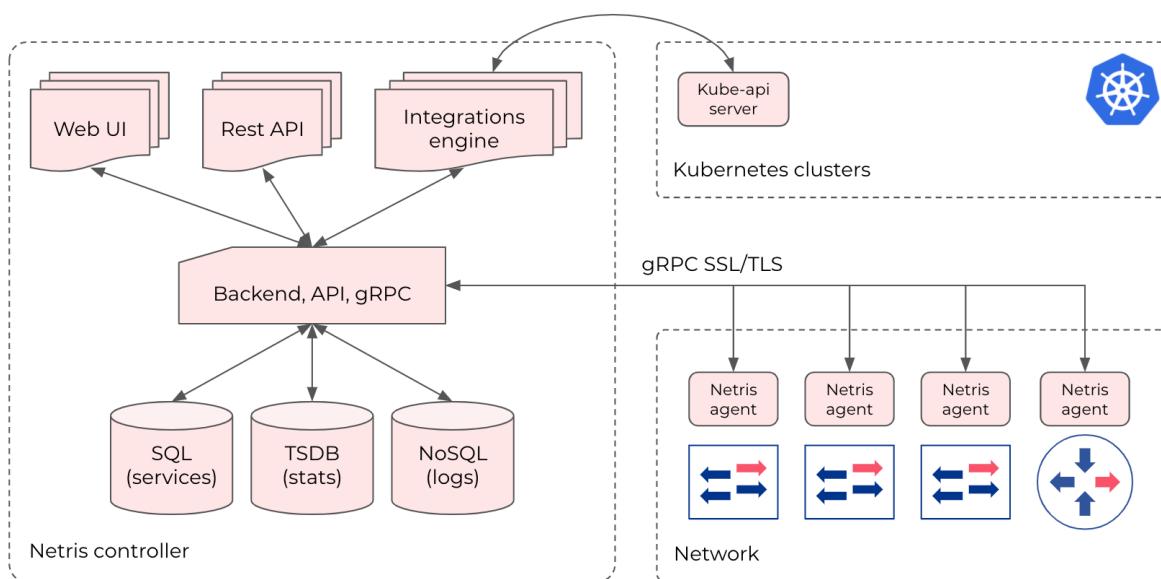
1.1 Introduction to Netris

Netris is an automatic netops software for operating physical networks like it is a cloud. Netris automatically configures switching, routing, load-balancing, and network security based on user-defined services and policies. Netris continuously monitors the network's health and either apply software remediation or informs you of necessary actions if human intervention is required. Netris abstracts away the complexities of detailed network configuration, letting you perform efficiently by operating your physical network in a top down approach like a cloud – instead of the legacy box by box operation.

1.2 What is Netris Controller

Netris Controller is the main operations control center for engineers using GUI/RestAPI/Kubernetes, systems, and network devices. Netris Controller stores the data representing the user-defined network services and policies, health, statistics, analytics received from the network devices, and information from integration modules with external systems (Kubernetes). Netris Controller can run as a VM or container, on/off-prem, or in Netris cloud.

Diagram: High level Netris architecture.



- **Controller HA.** We highly recommend running more than one copy of the controller for database replication.
- **Multiple sites.** Netris is designed to operate multiple sites with just a single controller with HA.
- **What if the controller is unreachable.** Netris operated switches/routers can tolerate the unreachability of the Netris Controller. Changes and stats collection will be unavailable during the controller unavailability window; however, switches/routers' core operation will not be affected.

1.3 Netris Switch Agent

Netris Switch Agent is software running in the user space of the network operating system (NOS) of the switch and is responsible for automatically generating the particular switch configuration according to service requirements and policies defined in the Netris Controller. Netris Switch Agent uses an encrypted GRPC protocol for secure communication with the Netris Controller accessible through a local management network or over the Internet.

1.4 Netris SoftGate

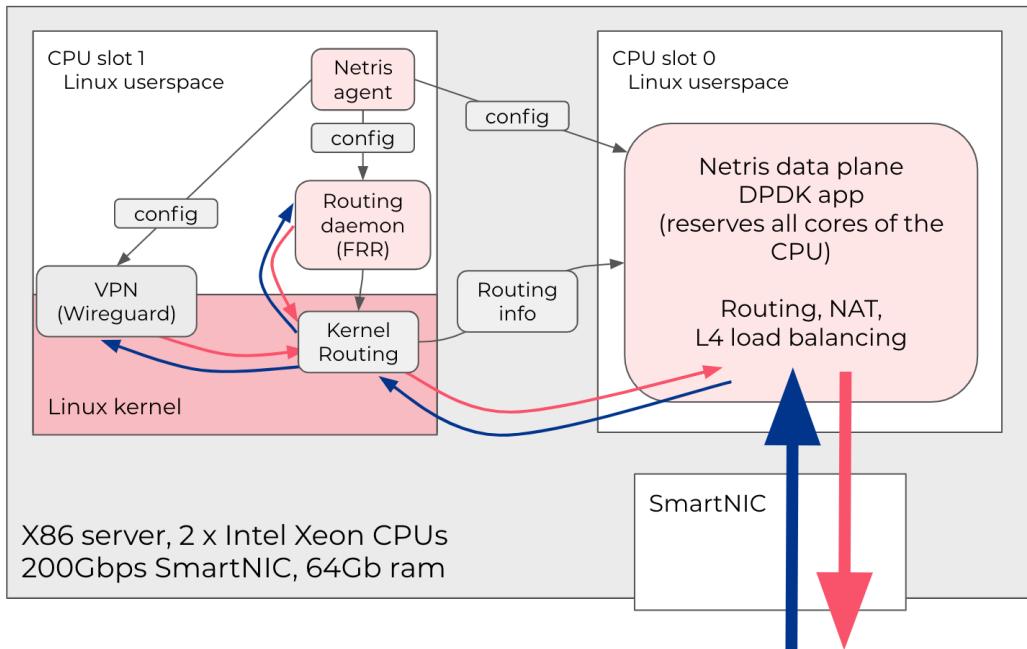
Netris SoftGate is automatic configuration software and reference architecture for enabling border routing, Layer-4 Load Balancing, Network Address Translation (NAT), and site-to-site VPN function on a regular x86 server with a SmartNIC card.

Netris SoftGate supports a high-performance DPDK data plane running in the user-space. It configures the system so that packets entering the NIC (network interface card) bypass Linux Kernel and go directly to the user space application. So traffic from the NIC travels through the PCIe bus to the closest CPU's last level cache and then into one of 8 cores, all reserved for the data-plane application. DPDK data-plane software processes the traffic for routing, load-balancing, NAT and makes necessary changes in the packet header (rewrites mac/VLAN-id) then returns the packet to the NIC, which sends it further into the switch for traveling further in Layer-2.

The server has to have 2 x Intel CPUs (8+ cores each). One CPU (closest to the SmartNIC card) is reserved for the data-plane process only (OS will report 100% CPU usage). Another CPU is used for running Linux OS, routing control plane (FRR), Netris agent, and other standard Linux utilities.

Netris agents can also configure Wireguard to form full mesh VPN tunnels between customer sites and then run necessary dynamic routing. So, servers and applications in multiple data centers can communicate over the Internet using encrypted tunnels.

Diagram: Netris SoftGate high level architecture



On-prem Netris Controller installation

Netris Controller can be hosted in Netris cloud, installed locally as a VM, or deployed as a Kubernetes application. All three options provide the same functionality. Cloud-hosted Controller can be moved into on-prem anytime.

2.1 KVM virtual machine

Minimal system requirements for the VM:

CPU - 8 Core

RAM - 16 Gb

Disk - 100Gb

Network - 1 virtual NIC

2.2 Installation steps for KVM hypervisor

If KVM is not already installed, install Qemu/KVM on the host machine (example provided for Ubuntu Linux 18.04)

```
sudo apt-get install virt-manager
```

2.3 Netris Controller Installation steps

1. Download the Netris Controller image. (contact Netris support for repository access permissions).

```
cd /var/lib/libvirt/images  
sudo wget http://img.netris.ai/netris-controller.qcow2
```

2. Download vm definition file.

```
cd /etc/libvirt/qemu  
sudo wget http://img.netris.ai/netris-controller.xml
```

3. Define the KVM virtual machine

```
sudo virsh define netris-controller.xml
```

Note Netris controller virtual NIC will bind to the “br-mgmt” interface on the KVM host machine. See below network interface configuration exam

Example: Network configuration on host (hypervisor) machine.

Note replace <Physical NIC>, <host server management IP/prefix length> and <host server default gateway> with the correct NIC and IP for your host machine.

```
sudo vim /etc/network/interfaces
```

```
#Physical NIC connected to the management network
auto <Physical NIC>
iface <Physical NIC> inet static
    address 0.0.0.0/0

#bridge interface
auto br-mgmt
iface br-mgmt inet static
    address <host server management IP/prefix length>
    gateway <host server default gateway>
    bridge-ports <Physical NIC>

source /etc/network/interfaces.d/*
```

```
sudo ifreload -a
```

4. Set the virtual machine to autostart and start it.

```
sudo virsh autostart netris-controller
```

```
sudo virsh start netris-controller
```

2.4 Accessing the Netris Controller

By default, Netris Controller will obtain an IP address from a **DHCP** server.

Below steps describe how to configure a **static IP** address for the Netris Controller.

1. Connecting to the VM console.

default credentials. **login:** netris **password:** newNet0ps

```
sudo virsh console netris-controller
```

Note Do not forget to change the default password (using passwd command).

2. Setting a static IP address.

Edit network configuration file.

```
sudo vim /etc/network/interfaces
```

Example: IP configuration file.

```
# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
iface eth0 inet static
    address <Netris Controller IP/prefix length>
    gateway <Netris Controller default gateway>
    dns-nameserver <a DNS server address>

source /etc/network/interfaces.d/*
```

Reload the network config.

```
sudo ifreload -a
```

Note Make sure Netris Controller has Internet access.

3. Reboot the controller

```
sudo reboot
```

After reboot, the Netris Controller GUI should be accessible using a browser. Use netris/newNet0ps credentials.



Don't forget to change the default password by clicking your login name in the top right corner and then clicking "Change Password".

2.5 Security hardening

Recommended for production use.

2.5.1 Changing the default GRPC authentication key.

Connect to the Netris Controller CLI (SSH or Console)

Tip: You can generate a random and secure key using sha256sum.

```
echo "<some random text here>" | sha256sum
```

example:

```
netris@iris:~$ echo "<some random text here>" | sha256sum  
6a284d55148f81728f932b28e9d020736c8f78e1950b3d576f6e679d90516df1 -
```

Set your newly generated secure key into Netris Controller.

```
sudo /opt/telescope/netris-set-auth.sh --key <your key>
```

Please store the auth key in a safe place as it will be required every time when installing Netris Agent for the switches and SoftGates.

2.5.2 Replacing the SSL certificate

1. Replace below file with your SSL certificate file.

```
/etc/nginx/ssl/controller.cert.pem;
```

2. Replace below file with your SSL private key.

```
/etc/nginx/ssl/controller.key.pem;
```

3. Restart Nginx service.

```
systemctl restart nginx.service
```

Netris Controller Helm Chart

- Installs the automatic NetOps platform Netris Controller

3.1 Prerequisites

- Kubernetes 1.12+
- Helm 3.1+
- PV provisioner support in the underlying infrastructure

3.2 Get Repo Info

Add the Netris Helm repository:

```
helm repo add netrisai https://netrisai.github.io/charts  
helm repo update
```

3.3 Installing the Chart

In order to install the Helm chart, you must follow these steps:

Create the namespace for netris-controller:

```
kubectl create namespace netris-controller
```

Generate strong auth key

```
export mystrongauthkey=$(date |base64 | md5sum | base64 | head -c 32)
```

Install helm chart with netris-controller

```
helm install netris-controller netrisai/netris-controller \  
--namespace netris-controller \  
--set app.ingress.hosts={my.domain.com} \  
--set netris.authKey=$mystrongauthkey
```

3.4 Uninstalling the Chart

To uninstall/delete the netris-controller helm release:

```
helm uninstall netris-controller
```

3.5 Configuration

The following table lists the configurable parameters of the netris-controller chart and their default values.

3.5.1 Common parameters

Parameter	Description	Default
nameOverride	String to partially override common.names.full-name template with a string (will prepend the release name)	nil
fullnameOverride	String to fully override common.names.fullname template with a string	nil
serviceAccount.create	Create a serviceAccount for the deployment	true
serviceAccount.name	Use the serviceAccount with the specified name	" "
serviceAccount.annotations	Annotations to add to the service account	{ }
podAnnotations	Pod annotations	{ }
podSecurityContext	Pod Security Context	{ }
securityContext	Containers security context	{ }
resources	CPU/memory resource requests/limits	{ }
nodeSelector	Node labels for pod assignment	{ }
tolerations	Node tolerations for pod assignment	[]
affinity	Node affinity for pod assignment	{ }

3.5.2 Netris-Controller common parameters

Parameter	Description	Default
netris.webLogin	Netris Controller GUI default login	netris
netris.webPassword	Netris Controller GUI default password	newNet0ps
netris.authKey	Netris Controller agents authentication key	mystrongkey

3.5.3 Netris-Controller app parameters

Parameter	Description	Default
app.replicaCount	Number of replicas in app deployment	1
app.image.repository	Image repository	netrisai/xcaas-xcaas-web

app.image.tag	Image tag. Overrides the image tag whose default is the chart appVersion	" "
app.image.pullPolicy	Image pull policy	IfNotPresent
app.imagePullSecrets	Reference to one or more secrets to be used when pulling images	[]
app.service.type	Kubernetes service type	ClusterIP
app.service.port	Kubernetes port where service is expose	80
app.service.portName	Name of the port on the service	http
app.ingress.enabled	Enables Ingress	true
app.ingress.annotations	Ingress annotations (values are templated)	kubernetes.io/ingress.class: nginx }
app.ingress.labels	Custom labels	{ }
app.ingress.path	Ingress accepted path	/
app.ingress.pathType	Ingress type of path	Prefix

app.ingress.hosts	Ingress accepted hostnames	["chart-example.local"]
app.ingress.tls	Ingress TLS configuration	[]
app.autoscaling.enabled	Option to turn autoscaling on for app and specify params for HPA. Autoscaling needs metrics-server to access cpu metrics	false
app.autoscaling.minReplicas	Default min replicas for autoscaling	1
app.autoscaling.maxReplicas	Default max replicas for autoscaling	100
app.autoscaling.targetCPUUtilizationPercent	The desired target CPU utilization for autoscaling	80

3.5.4 Netris-Controller grpc parameters

Parameter	Description	Default
grpc.replicaCount	Number of replicas in grpc deployment	1
grpc.image.repository	Image repository	netrisai/xcaas-agent-api-server

grpc.image.tag	Image tag. Overrides the image tag whose default is the chart appVersion	" "
grpc.image.pullPolicy	Image pull policy	IfNotPresent
grpc.imagePullSecrets	Reference to one or more secrets to be used when pulling images	[]
grpc.service.type	Kubernetes service type	ClusterIP
grpc.service.port	Kubernetes port where service is expose	443
grpc.service.portName	Name of the port on the service	grpc

grpc.autoscaling.enabled	Option to turn autoscaling on for app and specify params for HPA. Autoscaling needs metrics-server to access cpu metrics	false
grpc.autoscaling.minReplicas	Default min replicas 1 for autoscaling	1
grpc.autoscaling.maxReplicas	Default max replicas 100 for autoscaling	100
grpc.autoscaling.targetCPUUtilizationPercentage	The desired target CPU utilization for autoscaling	80

3.5.5 Netris-Controller telescope parameters

Parameter	Description	Default
telescope.replicaCount	Number of replicas in telescope deployment	1
telescope.image.repository	Image repository	netrisai/xcaas-telescope-go

<code>telescope.image.tag</code>	Image tag. Overrides the image tag whose default is the chart appVersion	" "
<code>telescope.image.pullPolicy</code>	Image pull policy	IfNotPresent
<code>telescope.imagePullSecrets</code>	Reference to one or more secrets to be used when pulling images	[]
<code>telescope.service.type</code>	Kubernetes service type	ClusterIP
<code>telescope.service.port</code>	Kubernetes port where service is exposed	80
<code>telescope.service.portName</code>	Name of the port on the service	ws
<code>telescope.service.securePort</code>	Kubernetes secure port where service is exposed	443
<code>telescope.service.securePortName</code>	Name of the secure port on the service	wss

<code>telescope.autoscaling.enabled</code>	Option to turn autoscaling on for app and specify params for HPA. Autoscaling needs metrics-server to access cpu metrics.
<code>telescope.autoscaling.minReplicas</code>	Default min replicas1 for autoscaling
<code>telescope.autoscaling.maxReplicas</code>	Default max replicas100 for autoscaling
<code>telescope.autoscaling.targetCPUUtilizationPercentage</code>	The desired target CPU utilization for autoscaling

3.5.6 Netris-Controller k8s-watcher parameters

Parameter	Description	Default
<code>k8s-watcher.replicaCount</code>	Number of replicas in k8s-watcher deployment	1
<code>k8s-watcher.image.repository</code>	Image repository	<code>netrisai/xcaas-kuberis-k8-api-agent</code>

k8s-watcher.image.tag	Image tag. Overrides the image tag "" whose default is the chart appVersion
k8s-watcher.image.pullPolicy	Image pullIfNotPresent policy
k8s-watcher.imagePullSecrets	Reference to one or more secrets [] to be used when pulling images
k8s-watcher.autoscaling.enabled	Option to turn autoscaling on for app and specify params for HPA. Autoscaling needs metrics-server to access cpu metrics
k8s-watcher.autoscaling.minReplicas	Default min replicas for autoscaling
k8s-watcher.autoscaling.maxReplicas	Default max replicas for autoscaling

<code>k8s-watcher.autoscaling.targetCPUUtilizationPercentage</code>	The desired target CPU utilization for autoscaling
---	--

3.5.7 Netris-Controller telescope-notifier parameters

Parameter	Description	Default
<code>telescope-notifier.replicaCount</code>	Number of replicas in telescope-notifier deployment	1
<code>telescope-notifier.image.repository</code>	Image repository	<code>netrisai/xcaas-xcaas-notifier</code>
<code>telescope-notifier.image.tag</code>	Image tag. Overrides the image tag whose default is the chart appVersion	" "
<code>telescope-notifier.image.pullPolicy</code>	Image pull policy	<code>IfNotPresent</code>
<code>telescope-notifier.imagePullSecrets</code>	Reference to one or more secrets to be used when pulling images	[]

<code>telescope-notifier.autoscaling.enabled</code>	Option to turn autoscaling on for app and specify params for HPA. Autoscaling needs metrics-server to access cpu metrics
<code>telescope-notifier.autoscaling.minReplicas</code>	Default min replicas for autoscaling
<code>telescope-notifier.autoscaling.maxReplicas</code>	Default max replicas for autoscaling
<code>telescope-notifier.autoscaling.targetCPUUtilizationPercentage</code>	The desired target CPU percentage utilization for autoscaling

3.5.8 Mariadb parameters

Using default values from

Parameter	Description	Default
<code>mariadb.image.repository</code>	MariaDB image name. We extended bitnami's mariadb image with own plugin	netrisai/netris-mariadb
<code>mariadb.image.tag</code>	MariaDB image tag. (only 10.1 is supported)	10.1
<code>mariadb.initdbScriptsConfigMap</code>	ConfigMap with the initdb scripts.	netris-controller-initdb
<code>mariadb.auth.database</code>	Name for a database to create	netris
<code>mariadb.auth.username</code>	Name for a user to create	netris

mariadb.auth.password	Password for the new user	changeme
mariadb.auth.rootPassword	Password for the root user	changeme

Auth from existing secret not supported at the moment

3.5.9 Mongodb parameters

Using default values from

Parameter	Description	Default
mongodb.useStatefulSet	Use StatefulSet instead of Deployment when deploying standalone	true
mongodb.initdbScriptsConfigMap	ConfigMap with the initdb scripts.	netris-controller-initdb-mongodb
mongodb.auth.database	Name for a database to create	netris
mongodb.auth.username	Name for a user to create	netris
mongodb.auth.password	Password for the new user	changeme
mongodb.auth.rootPassword	Password for the root user	changeme

Auth from existing secret not supported at the moment

3.5.10 Redis parameters

Using default values from

Parameter	Description	Default
redis.cluster.enabled	Use master-slave topology	false
redis.usePassword	Use password	false

Auth not supported at the moment

3.5.11 Smtip parameters

Using default values from

Parameter	Description	Default
smtp.config.DISABLE_IPV6	Disable IPv6	1

smtp.config.RELAY_NETWORKS	Relay networks. Change if your CNI use other subnets	:172.16.0.0/12:10.0.0.0/8:192.168.0.0/16
----------------------------	---	--

3.5.12 HAproxy parameters

Using default values from

Parameter	Description	Default
haproxy.enabled	Enable HAProxy. Used for exposing netris agents ports from single loadbalancer ip. Disable if you can't have type:LoadBalancer service in cluster	true
haproxy.service.type	Kubernetes service type	LoadBalancer

3.5.13 Graphite parameters

Using default values from

Parameter	Description	Default
graphite.configMaps	Netris-Controller supported graphite's config files	see in values.yaml
graphite.service.type	Kubernetes service type	ClusterIP

3.6 Usage

Specify each parameter using the `--set key=value[,key=value]` argument to helm install. For example,

```
helm install netris-controller netrisai/netris-controller \
--namespace netris-controller \
--set app.ingress.hosts={my.domain.com} \
--set netris.authKey=$mystrongauthkey \
--set mariadb.auth.rootPassword=my-root-password \
--set mariadb.auth.password=my-password \
--set mongodb.auth.rootPassword=my-root-password \
--set mongodb.auth.password=my-password
```

The above command sets netris-controller application ingress host to `my.domain.com` and sets generated `netris.authKey`. Additionally, it sets MariaDB and MongoDB root account password to `my-root-password` and user account password to `my-password`.

Alternatively, a YAML file that specifies the values for the parameters can be provided while installing the chart. For example,

```
helm install netris-controller netrisai/netris-controller --namespace
netris-controller -f values.yaml
```

After installation use EXTERNAL-IP of haproxy service as `--controller` parameter in `netris-setup`

```
kubectl get svc -nnetris-controller |grep haproxy
```

and `$mystrongauthkey` as `--auth` parameter in `netris-setup`

```
echo $mystrongauthkey
```

Also you can see overrides values from helm get values

```
helm get values netris-controller
```

Controller initial configuration

4.1 Definitions

- **User** - A user account for accessing Netris Controller through GUI, RestAPI, and Kubernetes. The default username is `netris`, with password `newNet0ps`.
- **Tenant** - IP addresses and Switch Ports are network resources assigned to different Tenants to have under their management. Admin is the default tenant, and by default, it owns all the resources. You can use different Tenants for sharing and delegation of control over the network resources. Network teams typically use Tenants to grant access to other groups to request & manage network services using the Netris Controller as a self-service portal or programmatically (with Kubernetes CRDs) DevOps/NetOps pipeline.
- **Permission Group** - List of permissions on a per section basis can be attached individually to a User or a User Role.
- **User Role** - Group of user permissions and tenants for role-based access control.
- **Site** - Each separate deployment (each data center) should be defined as a Site. All network units and resources are attached to a site. Netris Controller comes with a “default” site preconfigured. Site entry defines global attributes such as; AS numbers, default ACL policy, Site Mesh (site to site VPN) type.
- **Subnet** - IPv4/IPv6 address resources linked to Sites and Tenants.
- **Switch Port** - Physical ports of all switches attached to the system. Switch port objects represent statuses, take basic parameters, and are assigned to Tenants.
- **Inventory** - This is an inventory of all network units that are operated using Netris Agent.
- **E-BGP** - Is for defining all External BGP peers (iBGP and eBGP).

4.2 Subnets

It is required to define at least two subnets to get started. One subnet is for the management interfaces, another for the loopback addresses. Every network unit managed with Netris should have at least one management IP and at least one loopback IP. Loopback IP addresses are used for network unit identification by network protocols and by Netris Agent/Controller. There’s no need for defining any IP addresses for the switch-to-switch links. Netris is using IPv6 link-local addresses for all switch-to-switch communication.

4.2.1 Example: (IP addresses used are just examples, please replace them following your IP planning.)

In NET->Subnets section of the Netris Controller GUI, you can add new subnet entries. Subnets are of 2 types of allocation and assignment. Allocations are the large blocks of IP resources assigned to the organization. Assignments are IP blocks that are smaller blocks inside the allocation and can be used by services or policies that yet to be defined.

1. Adding a new allocation. In this example, 10.0.0.0/8 is used as a large block of allocation. You can add as many allocations as required.

Name*	Private
Prefix*	10.0.0.0/8
Type	Allocation
Site *	Default

Cancel
Add

2. Adding two new assignments.

- 10.254.96.0/24 (netManagement) assigned to the tenant “Admin” and available for the site “Default”.
- 10.254.97.0/24 (netLoopbacks) assigned to the tenant “Admin” and available for the site “Default”.

Name*	netManagement
Prefix*	10.254.96.0/24
Type	Assignment
Assign *	Admin
Purpose *	standard

+Sites
?
Default

Cancel
Add

❖ Add Assignment

Name*	netLoopbacks
Prefix*	10.254.97.0/24
Type	Assignment
Assign *	Admin
Purpose *	standard

+Sites
?
Default
?

Cancel
Add

Screenshot: Listing of the Subnets section after adding the new objects.

Subnets				
Description	Subnets	Tenant	Site	Purpose
controller-lo	169.254.254.254/32		Default	
Private	10.0.0.0/8		Default	
netManagement	10.254.96.0/24	Admin	Default	standard
netLoopbacks	10.254.97.0/24	Admin	Default	standard

4.3 Inventory Profiles

Inventory profiles define access security, timezone, DNS, NTP settings profiles for network switches and SoftGate nodes. To create a new Inventory profile, click +Add under the Net→Inventory Profiles section.

4.3.1 Fields descriptions:

- **Name** - Profile name.
- **Description** - Free text description.
- **Allow SSH from IPv4** - List of IPv4 subnets allowed to ssh (one address per line)
- **Allow SSH from IPv6** - List of IPv6 subnets allowed to ssh (one address per line)
- **Timezone** - Devices using this inventory profile will adjust their system time to the selected timezone.
- **NTP servers** - List of domain names or IP addresses of NTP servers (one address per line). You can use your Netris Controller address as an NTP server for your switches and SoftGate.
- **DNS servers** - List of IP addresses of DNS servers (one address per line). You can use your Netris Controller address as a DNS server for your switches and SoftGate.

Example: In this example Netris Controller is used to provide NTP and DNS services to the switches (common setup).

The screenshot shows a configuration form for an inventory profile. The fields are as follows:

- Name*: my_inv_profile
- Description: Description
- Allow SSH from IPV4*: 10.254.0.0/16
- Allow SSH from IPV6: (empty)
- Timezone: (GMT-08:00) Pacific Time
- NTP servers: 10.254.96.10
- DNS servers: 10.254.96.10

Below the form, there is a section for "Custom Rules" with a "+ Add" button. At the bottom right are "Cancel" and "Add" buttons.

4.4 Adding Switches to Topology

You need to define every switch in the Net→Topology section. To add a switch, please go to Net→Topology and click +Add.

- **Name** - Descriptive name.
- **Owner Tenant** - Tenant(typically Admin) who administers this node.
- **Description** - Free text description.
- **Hardware Type** - For switches: Spine Switch or Leaf Switch.
- **NOS*** - Network operating system. Cumulus Linux, Ubuntu SwitchDev (Nvidia Mellanox only), SONiC (not for production use yet)
- **Site*** - The site where the switch belongs.
- **Inventory Profile** - Reference to Timezone, DNS, NTP, and Security features profile.
- **IP Address*** - IPv4 address for the loopback interface.
- **Management IP address** - IPv4 address for the out of band management interface.
- **Zero-touch provisioning** - Automatically install the NOS. (Experimental in this version)
- **MAC address** - Out of band management interface MAC address used for zero-touch provisioning. (Experimental in this version)
- **The number of ports** - It is required for the topology manager. Will be synced to the real number of Switch Ports when Netris Switch Agent establishes the very first connection with the Netris Controller.

Example: Adding a spine switch w/ Cumulus Linux.

 Add New Hardware x

Name*	spine1				
Owner Tenant*	Admin				
Description	description				
Hardware Type	 Spine Switch				
NOS*	Cumulus Linux				
Site*	Default				
Inventory Profile	my_inv_profile				
IP Address*	10.254.97.0/24(netLoop...)		10.254.97.11		
Management IP Address*	10.254.96.0/24(netMana...		10.254.96.11		
Zero touch provisioning	<input type="checkbox"/>				
Mac address	Mac address				
Number of Ports	<input type="radio"/> 16	<input type="radio"/> 32	<input type="radio"/> 48	<input checked="" type="radio"/> 54	<input type="radio"/> 56

Cancel
Add

Tip: You can drag/move the units to your desired positions and click “Save positions”.

Note: Repeat this process to define all your switches.

4.5 Topology Manager

The topology manager is for describing and monitoring the desired network topology. Netris Switch Agents will configure the underlying network devices according to this topology dynamically and start watching against potential failures.

To define the links, right-click on the spine switch, then click create a link. Select the “from port,” then “to device” and “port.” See the example below.

❖ Create Link

From

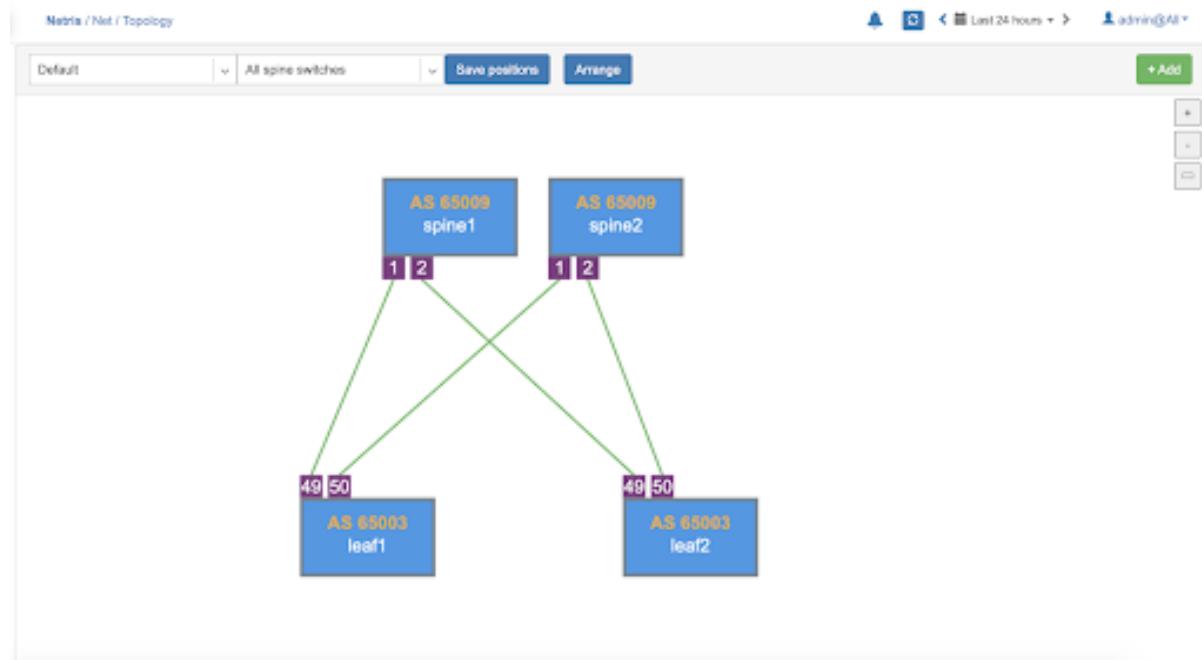
Device	spine1
Port	port2(port2)@spine1

To

Device	leaf2
Port	port49(port49)@leaf2

Cancel
Create

All links require definition in the topology manager. Topology links can also be described through a .yaml file when using Kubernetes CRD. (a GUI wizard is planned to be available in v2.10).



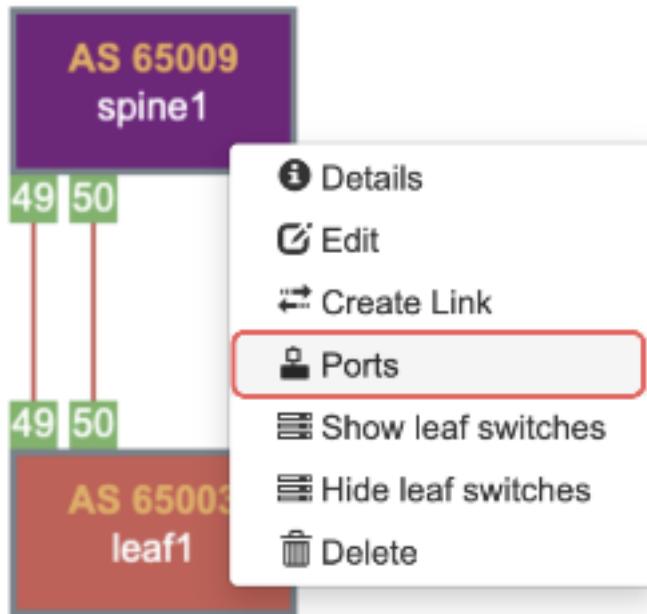
Now when network units and links are defined, your network is automatically configured as long as physical connectivity is in place and Netris Agents can communicate with Netris Controller.

4.6 Hairpin (Cumulus only)

With Cumulus Linux only, we need to loop two ports on spine switches (hairpin cable) in the current release, usually two upstream (higher capacity) ports. We are planning to lift this requirement in the next Netris release (v2.10).

To define what ports will be used as a hairpin, navigate to Net→Switch Ports, or right-click on the spine switch, click Ports in Net→Topology.

Example: Accessing Switch Ports from Net→Topology



For each spine switch, find the two ports that you are going to connect (loop/hairpin) and configure one port as a “hairpin l2” and another port as “hairpin l3”. The order doesn’t matter. The system needs to know which ports you have dedicated for the hairpin/loop on each spine switch. (do not do this for non-Cumulus switches) | | Example: Editing Switch Port from Net→Switch Ports.

▶ □ port52	port52	Admin	spine1	standard	Unknown	⋮
▶ □ port53	port53	Admin	spine1	standard	Unknown	⋮
▶ □ port54	port54	Admin	spine1	standard	Unknown	⋮

Example: Setting port types to “hairpin l2” and “hairpin l3”.

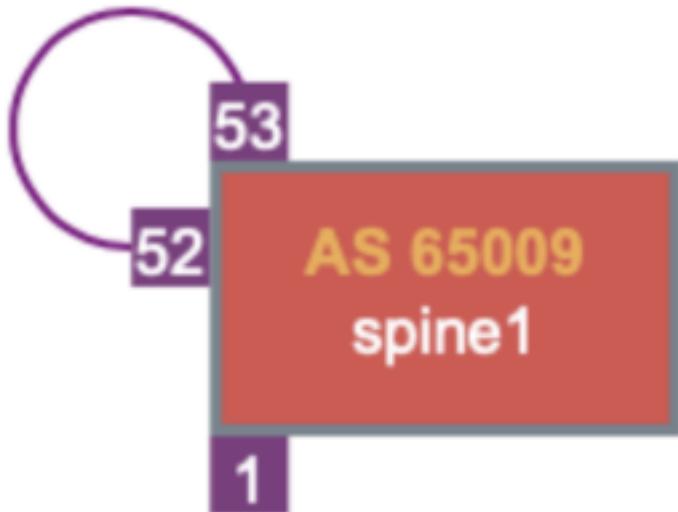
❖ Edit port52

Name	port52
Port type	hairpin l2
MTU	9000
Port Speed	Auto
Extension	None

❖ Edit port53

Name	port53
Port type	hairpin l3
MTU	9000
Port Speed	Auto
Extension	None

Screenshot: Hairpin visualized in Net→Topology



4.7 Adding SoftGate nodes to Topology

Every SoftGate node first needs to be defined in Netris Controller. To add a SoftGate node, please go to Net→Topology and click +Add.

- **Name** - Descriptive name.
- **Owner Tenant** - Tenant(typically Admin), who administers this node.
- **Description** - Free text description.
- **Hardware Type** - NFV node.
- **Site** - The data center where the current SoftGate node belongs.
- **Inventory Profile** - Profile describing the timezone, DNS, NTP, and Security features.
- **IP Address** - IPv4 address for the loopback interface.
- **Management IP address** - IPv4 address for the out of band management interface.
- **NFV Node Port** - A physical port on a spine switch where the SoftGate node's first SmartNIC port is connected. Typically each spine switch has one SoftGate node connected to it.
- **+NAT address** - Public IP addresses to be used as global IP for SNAT/DNAT. (check Enabling NAT section of Network Policies chapter)
- **+NAT address pool** - Public IP address subnets to be used as rolling global IP addresses for SNAT. (check Enabling NAT section of Network Policies chapter)

Example: Adding a SoftGate Node to Topology.

❖ Add New Hardware

Name*	softgate1			
Owner Tenant*	Admin			
Description	description			
Hardware Type	NFV node			
Site*	Default-site			
Inventory Profile	mt_inv_profile			
IP Address*	10.254.97.0/24(netLoop...)	10.254.97.33	Delete	Loopback IP
Management IP Address*	10.254.96.0/24(netMana...)	10.254.96.33	Delete	OOB management IP
NFV Node Port *	swp54(port54)@spine1	Switch Port on a spine switch to be wired with the first SmartNIC port of the SoftGate server.		
+ NAT Address				
+ NAT Address Pool				

Cancel Add

Netris switch agent installation

5.1 For Cumulus Linux

Requirements: * Fresh install of Cumulus Linux v. 3.7.(x) - Cumulus 4.X is in the process of validation and will be supported in the next Netris release.

5.1.1 Configure the OOB Management IP address

Configure out of band management IP address, and in case Netris Controller is not in the same OOB network then configure a route to Netris Controller. No default route or other IP addresses should be configured.

```
sudo vim /etc/network/interfaces
```

```
# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
iface eth0 inet static
    address <Management IP address/prefix length>
    up ip ro add <Controller address> via <Management network gateway> #delete
    this line if Netris Controller is located in the same network with the switch.

source /etc/network/interfaces.d/*
```

```
sudo ifreload -a
```

5.1.2 Configure Cumulus Linux license

```
sudo cl-license -i
```

Copy/paste the Cumulus Linux license string then press ctrl-d.

5.1.3 Install the Netris Agent

1. Add netris repository using Netris Controller as an http proxy. Replace <Your Netris Controller address> with your actual Netris Controller address.

Note Netris Controller built-in proxy, by default, permits RFC1918 IP addresses (10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16). If your management network is using IP addresses outside these ranges

you will need to configure iptables on the Netris Controller accordingly.

```
export http_proxy=http://<Your Netris Controller address>:3128
wget -qO - http://repo.netris.ai/repo/public.key | sudo apt-key add -
echo "deb http://repo.netris.ai/repo/ jessie main" | sudo tee
/etc/apt/sources.list.d/netris.list
```

2. Update the apt

```
echo -e 'Acquire::http::Proxy "http://<Your Netris Controller
address>:3128";\nAcquire::https::Proxy "http://<Your Netris Controller
address>:3128";' | sudo tee -a /etc/apt/apt.conf.d/netris-proxy
sudo apt update
```

3. Install Netris Agent and dependencies

```
sudo apt install netris-sw
```

4. Initialize the switch using netris-setup

Description of netris-setup parameters

```
--auth - Authentication key, "6878C6DD88224981967F67EE2A73F092" is the default
key.
--controller - IP address or domain name of Netris Controller.
--hostname - The hostname for the current switch, this hostname should match the
name defined in the Controller.
--lo - IP address for the loopback interface, as it is defined in the controller.
--type - Role of the switch in your topology: spine/leaf
```

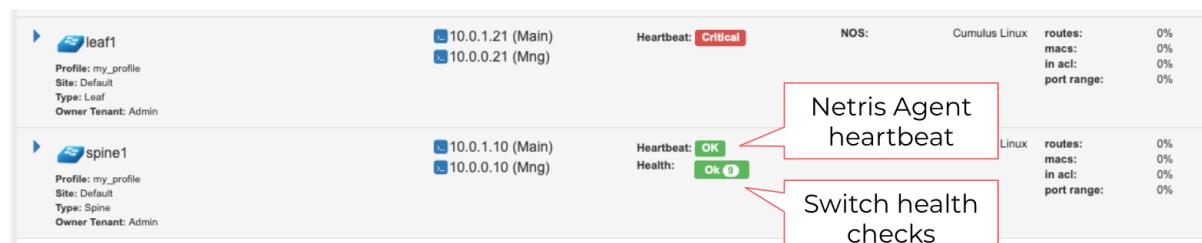
```
sudo /opt/netris/bin/netris-setup --auth=<authentication key> --controller=<IP or
FQDN> --hostname=<name> --lo=<loopback IP address> --type=<spine/leaf>
```

5. Reboot the switch

```
sudo reboot
```

Once the switch boots up you should see its heartbeat going from Critical to OK in Net→Inventory, Telescope→Dashboard, and switch color will reflect its health in Net→Topology

Screenshot: Net→Inventory



5.2 For Ubuntu SwitchDev

Note Further installation requires a Console and Internet connectivity via management port!

1. NOS Uninstall

Fist of all uninstall current NOS using **Uninstall OS** from grub menu:

```
+-----+  
| ONIE: Install OS  
| ONIE: Rescue  
| *ONIE: Uninstall OS  
| ONIE: Update ONIE  
| ONIE: Embed ONIE  
|  
|  
|  
|  
|  
|  
|  
+-----+
```

Once the uninstallation is completed, the switch will reboot automatically.

2. Update ONIE

Select **Update ONIE** from grub menu:

```
+-----+  
| ONIE: Install OS  
| ONIE: Rescue  
| ONIE: Uninstall OS  
| *ONIE: Update ONIE  
| ONIE: Embed ONIE  
|  
|  
|  
|  
|  
|  
|  
|  
+-----+
```

In case you don't have DHCP in the management network, then stop ONIE discovery service and configure IP address and default gateway manually:

```
onie-discovery-stop  
ip addr add <management IP address/prefix> dev eth0  
ip route add default via <gateway of management network>  
echo "nameserver 1.1.1.1" > /etc/resolv.conf
```

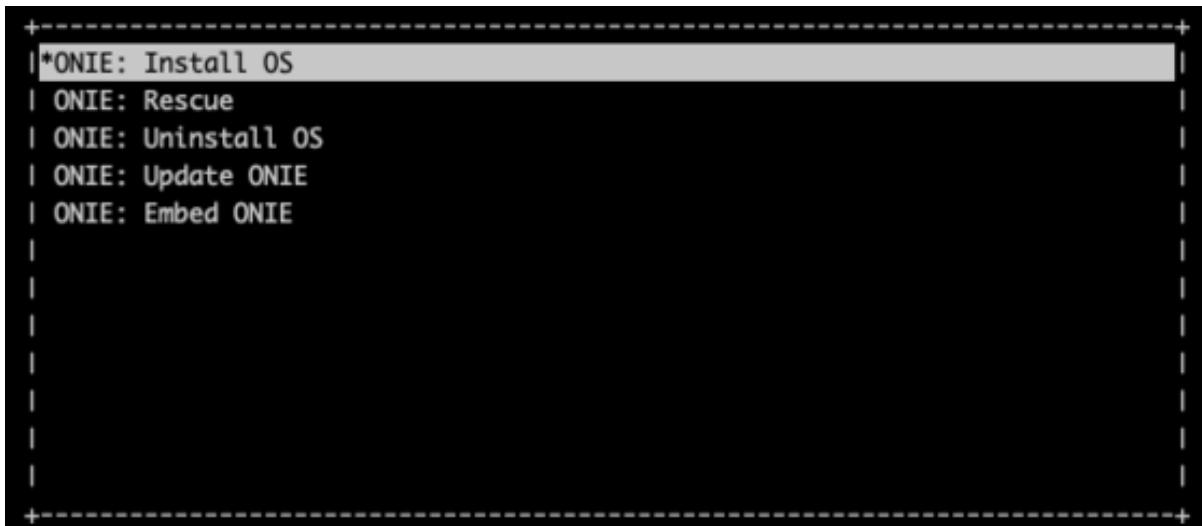
Update ONIE to the supported version.

Note ONIE image available for Mellanox switches only!

```
onie-self-update http://repo.netris.ai/repo/onie-updater-x86_64-mlnx_x86-r0
```

3. NOS Install

Select **Install OS** from grub menu:



In case you don't have DHCP in the management network, then stop ONIE discovery service and configure IP address and default gateway manually:

```
onie-discovery-stop
ip addr add <management IP address/prefix> dev eth0
ip route add default via <gateway of management network>
echo "nameserver 1.1.1.1" > /etc/resolv.conf
```

Install Ubuntu-SiwthDev from the Netris custom image:

```
onie-nos-install http://repo.netris.ai/repo/netris-ubuntu-18.04.1.bin
```

Default username/password

netris/newNet0ps

5.2.1 Configure the OOB Management IP address

Configure out of band management IP address, and in case Netris Controller is not in the same OOB network then configure a route to Netris Controller. No default route or other IP addresses should be configured.

```
sudo vim /etc/network/interfaces
```

```
# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
iface eth0 inet static
    address <Management IP address/prefix length>
```

```
up ip ro add <Controller address> via <Management network gateway> #delete  
this line if Netris Controller is located in the same network with the switch.  
  
source /etc/network/interfaces.d/*
```

```
sudo ifreload -a
```

5.2.2 Install the Netris Agent

1. Add netris repository using Netris Controller as an http proxy. Replace <Your Netris Controller address> with your actual Netris Controller address.

Note Netris Controller built-in proxy, by default, permits RFC1918 IP addresses (10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16). If your management network is using IP addresses outside these ranges you will need to configure iptables on the Netris Controller accordingly.

```
export http_proxy=http://<Your Netris Controller address>:3128  
  
wget -qO - http://repo.netris.ai/repo/public.key | sudo apt-key add -  
  
echo "deb http://repo.netris.ai/repo/ bionic main" | sudo tee  
/etc/apt/sources.list.d/netris.list
```

2. Update the apt

```
echo -e 'Acquire::http::Proxy "http://<Your Netris Controller  
address>:3128";\nAcquire::https::Proxy "http://<Your Netris Controller  
address>:3128";' | sudo tee -a /etc/apt/apt.conf.d/netris-proxy  
  
sudo apt update
```

3. Install Netris Agent and dependencies

```
sudo apt-get update && sudo apt-get install netris-sw
```

4. Initialize the switch using netris-setup

Description of netris-setup parameters

```
--auth - Authentication key, "6878C6DD88224981967F67EE2A73F092" is the default  
key.  
--controller - IP address or domain name of Netris Controller.  
--hostname - The hostname for the current switch, this hostname should match the  
name defined in the Controller.  
--lo - IP address for the loopback interface, as it is defined in the controller.  
--type - Role of the switch in your topology: spine/leaf
```

```
sudo /opt/netris/bin/netris-setup --auth=<authentication key> --controller=<IP or  
FQDN> --hostname=<name> --lo=<loopback IP address> --type=<spine/leaf>
```

5. Reboot the switch

```
sudo reboot
```


Netris SoftGate agent installation

6.1 Minimal hardware requirements

- 2 x Intel Silver CPU
- 96 GB RAM
- 300 GB HDD
- Nvidia Mellanox Connect-X 5 SmartNIC card

6.2 BIOS configuration

The following are some recommendations on BIOS settings. Different vendors will have different BIOS naming so the following is mainly for reference:

- Before starting consider resetting all BIOS settings to their defaults.
- Disable all power saving options such as: Power performance tuning, CPU P-State, CPU C3 Report and CPU C6 Report.
- Select Performance as the CPU Power and Performance policy.
- Disable Turbo Boost to ensure the performance scaling increases with the number of cores.
- Set memory frequency to the highest available number, NOT auto.
- Disable all virtualization options when you test the physical function of the NIC, and turn off VT-d.
- Disable Hyper-Threading.

6.3 Software installation

Requires freshly installed Ubuntu Linux 18.04 and network connectivity with your Netris Controller over the out-of-band management network.

1. Set environment variables to use Netris Controller as a proxy.

```
export http_proxy=http://<Your Netris Controller address>:3128 && export  
https_proxy=https://<Your Netris Controller address>:3128  
echo -e 'Acquire::http::Proxy "http://<Your Netris Controller
```

```
address>:3128";\nAcquire::https::Proxy "http://<Your Netris Controller  
address>:3128";' | sudo tee -a /etc/apt/apt.conf.d/netris-proxy
```

2. Config the apt for Mellanox repository.

```
wget -qO - https://www.mellanox.com/downloads/ofed/RPM-GPG-KEY-Mellanox | sudo  
apt-key add -  
  
wget  
http://linux.mellanox.com/public/repo/mlnx_ofed/5.0-2.1.8.0/ubuntu18.04/mellanox_mlnx_ofed.list  
-O /tmp/mellanox_mlnx_ofed.list && sudo mv /tmp/mellanox_mlnx_ofed.list  
/etc/apt/sources.list.d/
```

3. Config the apt for Netris repository.

```
wget -qO - http://repo.netris.ai/repo/public.key | sudo apt-key add -  
  
echo "deb http://repo.netris.ai/repo/ bionic main" | sudo tee  
/etc/apt/sources.list.d/netris.list
```

4. Install Mellanox drivers

```
sudo apt-get update && sudo apt-get install mlnx-ofed-dpdk
```

5. Install Netris agent package and dependencies, including specific Linux Kernel version.

```
sudo apt-get install netris-dpdk-mlnx
```

6. Configure Management IP address

Configure out of band management IP address. In case Netris Controller is not in the same OOB network then add a route to Netris Controller. No default route or other IP addresses should be configured.

```
sudo vim /etc/network/interfaces
```

```
# The loopback network interface  
auto lo  
iface lo inet loopback  
  
# The primary network interface  
auto eth0  
iface eth0 inet static  
    address <Management IP address/prefix length>  
    up ip ro add <Controller address> via <Management network gateway> #delete  
this line if Netris Controller is located in the same network with the SoftGate  
node.  
  
source /etc/network/interfaces.d/*
```

```
sudo ifreload -a
```

7. Initialize the SoftGate

netris-setup parameters, described below.

-auth - Authentication key, “6878C6DD88224981967F67EE2A73F092” is the default value, we strongly recommend to change this string in your controller as described in Controller initial configuration

section.

--controller - IP address or domain name of Netris Controller.

--hostname - Specify the hostname for the current switch, this hostname should match the name defined for particular switch in the Controller..

--lo - IP address for the loopback interface, as it is defined in the controller.

--node-prio - brief explanation of node priority goes here

Run netris-setup.

```
sudo /opt/netris/bin/netris-setup --lo=<SoftGate loopback IP address as defined in controller> --controller=<Netris Controller IP or FQDN> --hostname=<node name as defined in controller> --auth=<authentication key> --node-prio=<node priority 1 / 2>
```

Example: Running netris-setup

```
netris@ubuntu:~$ sudo /opt/netris/bin/netris-setup --lo=10.254.97.33
--controller=10.254.97.10 --hostname=softgate1
--auth=6a284d55148f81728f932b28e9d020736c8f78e1950b3d576f6e679d90516df1
--node-prio=1
* Setup Hostname
* Setup Hosts
* Setup Keepalived
* Setup Collectd
* Setup Loopback
* Get CPU List
* Setup FRR BGP Daemon
* Setup Netris Agent Config
* Setup DPDK Router Config
* Setup DPDK Router Systemd Unit
  * Setup Grub Config
* Update Grub
```

*** ATTENTION: You must reboot SoftGate to complete the installation

netris@ubuntu:~\$

8. Reboot the server

```
sudo reboot
```

When server boots up, you should see it's heartbeat status in Net→Inventory

Basic BGP

BGP neighbors can be declared in the Net→E-BGP section. Netris will automatically generate and inject the right configuration to meet your requirements as declared. See below description of E-BGP neighbor declaration fields.

- **Name** - Name for BGP session.
- **Description** - Free description.
- **Site** - Selects the site (data center) where this BGP session should be terminated on.
- **NFV Node** - Only if SoftGate nodes are in use, define on which node BGP session should be terminated on.
- **Neighbor AS** - Autonomous System number of the remote side. (Local AS is defined at Net→Sites section)
- **Terminate on switch** - Typically used for setups without SoftGate, for connecting with upstream routers. Instructs the system to terminate the BGP session directly on the switch.
- **Switch port** - Switch Port for the physical cable to the BGP neighbor. (any port on the fabric). Optionally can bind to a V-NET service, typically used for peering with IXPs or systems like GGC (Google Global Cache).
- **VLAN ID** - Optionally tag with a VLAN ID. (usually untagged)
- **IP Version** - IPv4 / IPv6
- **Local IP** - BGP peering IP address on Netris controlled side.
- **Remote IP** - BGP peering IP address on the remote end.
- **State** - Administrative state. (Enabled/Disabled)
- **Advanced** - Advanced policy settings are described in the next section.

Example: Declare a basic BGP neighbor.

❖ Edit E-BGP: Iris1 x

Name*	Iris1
Description	Description
Site*	Default-site
NFV Node*	softgate1
Neighbor AS*	100
Terminate on switch	No
Switch port*	swp17(port17)@spine1
VLAN ID*	Untagged
IP Version*	IPv4
Local IP*	50.117.59.66 /30
Remote IP*	50.117.59.65 /30
State*	Enabled

Which SoftGate node to terminate on.

Remote AS number.

Terminate the BGP session on the switch.

Switch Port for the neighbor cable.

BGP peering local/remote IPs (usually /30)

▶ Advanced

Cancel Save

Advanced BGP

BGP neighbor declaration can optionally include advanced BGP attributes and BGP route-maps for fine-tuning of BGP policies.

Click Advanced to expand the BGP neighbor add/edit window.

- **Neighbor address** - IP address of the neighbor when peering with the loopback IP address instead of the interface IP address. (aka Multihop).
- **Update source** - When Multihop BGP peering is used, it allows the operator to choose one of the loopback IP addresses of the SoftGate node as a BGP speaker source IP address.
- **BGP password** - Password for the BGP session.
- **Allow as-in** - Define the number of allowed occurrences of the self AS number in the received BGP NLRI to consider it valid. (normally 0)
- **Default Originate** - Originate default route to the current neighbor.
- **Prefix Inbound Max** - Drop the BGP session if the number of received prefixes exceeds this max limit. For switch termination maximum allowed is 1000 prefixes, while SoftGate termination can handle up to one million prefixes.
- **Inbound Route-Map** - Apply BGP policies described in a route-map for inbound BGP updates.
- **Outbound Route-Map** - Apply BGP policies described in a route-map for outbound BGP updates.
- **Local Preference** - Set local preference for all inbound routes for the current neighbor.
- **Weight** - Set weight for all inbound routes for the current neighbor.
- **Prepend Inbound(times)** - How many times to prepend self AS number for inbound routes.
- **Prepend Outbound(times)** - How many times to prepend self AS number for outbound routes.
- **Prefix List Inbound** - List of IP addresses prefixes to permit or deny inbound.
- **Prefix List Outbound** - List of IP addresses prefixes to permit or deny outbound.
- **Send BGP Community** - List of BGP communities to send to the current neighbor.

8.1 BGP objects

Under Net→E-BGP objects, you can define various BGP objects referenced from a route-map to declare a dynamic BGP policy.

Supported objects are:

- IPv4 Prefix
- IPv6 Prefix
- AS-PATH

- Community
- Extended Community
- Large Community

8.1.1 IPv4 Prefix.

Rules defined one per line.

Each line in IPv4 prefix list field consists of three parts:

- Action - Possible values are: permit or deny (mandatory).
- IP Prefix - Any valid IPv4 prefix (mandatory).
- Length - Possible values are: le <len>, ge <len> or ge <len> le <len>.

Example: Creating an IPv4 Prefix list.

Name*	Acme_Prefixes
Type*	IPv4 Prefix
IPv4 Prefix*	permit 198.51.100.0/24 le 32 permit 203.0.113.0/24 le 32

8.1.2 IPv6 Prefix.

Rules defined one per line.

Each line in IPv6 prefix list field consists of three parts:

- Action - Possible values are: permit or deny (mandatory).
- IP Prefix - Any valid IPv6 prefix (mandatory).
- Keyword - Possible values are: le <len>, ge <len> or ge <len> le <len>.

Example: Creating an IPv6 Prefix list.

Name*	Acme_IPv6_Prefixes
Type*	IPv6 Prefix
IPv6 Prefix*	permit 2001:DB8:1::/48 permit 2001:DB8:2::/48 le 64 deny 2001:DB8::/32

8.1.3 Community.

Community field has two parts:

- Action - Possible values: permit or deny (mandatory).
- Community string - format is AA:NN, where AA and NN are any number from 0 to 65535 range or alternatively well known string (local-AS | no-advertise | no-export | internet | additive).

Example: Creating community.

Name*	Acme_community
Type*	Community
Community*	permit 44395:666



8.2 BGP route-maps

Under the Net→E-BGP Route-maps section, you can define route-map policies, which can be associated with the BGP neighbors inbound or outbound.

Description of route-map fields:

- **Sequence Number** - Automatically assigned a sequence number. Drag and move sequences to organize the order.
- **Description** - Free description.
- **Policy** - Permit or deny the routes which match below all match clauses within the current sequence.
- **+Match** - Rules for route matching.
 - **Type** - Type of the object to match: AS-Path, Community, Extended Community, Large Community, IPv4 prefix-list, IPv4 next-hop, Route Source, IPv6 prefix-list, IPv6 next-hop, local-preference, MED, Origin, Route Tag.
 - **Object** - Select an object from the list.
- **Action** - Action when all match clauses are met.
 - **Action type** - Define whether to manipulate a particular BGP attribute or go to another sequence.
 - **Attribute** - The attribute to be manipulated.
 - **Value** - New attribute value.

Example: route-map

Name * route-map-1

+ Sequence

Sequence Number	5	 
Description	Prefer acme on this neighbor	
Policy	Permit	

+ Match

Match	IPv4 prefix-list		Acme_Prefixes		
Match	Origin		egp		

+ Action

Action	set		local-preference		300	
--------	-----	---	------------------	---	-----	---

Sequence Number 10

Description permit the rest

Policy Permit

+ Match

+ Action

Routes (static routing)

Located under Net→Routes is a method for describing static routing policies that Netris will dynamically inject on switches and/or SoftGate where appropriate. We recommend using the Routes only if BGP is not supported by the remote end.

Typical use cases for Routes

- To connect the switch fabric to an ISP or upstream router in a situation where BGP and dual-homing are not supported.
- Temporary interconnection with the old network for a migration.
- Routing a subnet behind a VM hypervisor machine for an internal VM network.
- Specifically routing traffic destined to a particular prefix through an out-of-band management network.

Add new static route fields description:

- **Prefix** - Route destination to match.
- **Next-Hop** - Traffic destined to the Prefix will be routed towards the Next-Hop. Note that static routes will be injected only on units that have the Next-Hop as a connected network.
- **Description** - Free description.
- **Site*** - Site where Route belongs.
- **State** - Administrative (enable/disable) state of the Route.
- **+Apply to** - Limit the scope to particular units. It's typically used for Null routes.

Example: Default route pointing to a Next-Hop that belongs to one of V-NETs.

❖ Edit static route x

Prefix*	0.0.0.0/0	
Next-Hop*	10.0.3.10	<input type="checkbox"/> Null ?
Description	Default route	
Site*	Default	▼
State	Enabled	▼

+Apply to ▼ ?

Cancel Save

Example: Adding a back route to 10.254.0.0/16 through an out-of-band management network.

❖ Add new static route x

Prefix*	10.254.0.0/16	
Next-Hop*	10.254.96.1	<input type="checkbox"/> Null ?
Description	OOB Management backroute	
Site*	Default	▼
State	Enabled	▼

+Apply to ▼ ?

Cancel Add

Screenshot: This Shows that my back route is actually applied on leaf1 and spine1.

Prefix	↓↑	Next-Hop	↓↑	State	↓↑	Applied	↓↑	Apply to	↓↑	Description	↓↑	Site
10.254.0.0/16		10.254.96.1		Active		leaf1 spine1				OOB Management backroute		Default

NAT

Netris SoftGate nodes are required to support NAT (Network Address Translation).

10.1 Enabling NAT

To enable NAT for a given site, you first need to attach NAT IP addresses and/or NAT IP pool resources to SoftGate nodes. NAT IP addresses can be used for SNAT or DNAT as a global IP address (the public IP visible on the Internet). NAT IP pools are IP address ranges that SNAT can use as a rolling global IP (for a larger scale, similar to carrier-grade SNAT). SNAT is always overloading the ports, so many local hosts can share one or just a few public IP addresses. You can add as many NAT IP addresses and NAT pools as you need, assuming it's configured as an allocation under Net→Subnets section.

1. Allocate a public IP subnet for NAT under Net→Subnets.

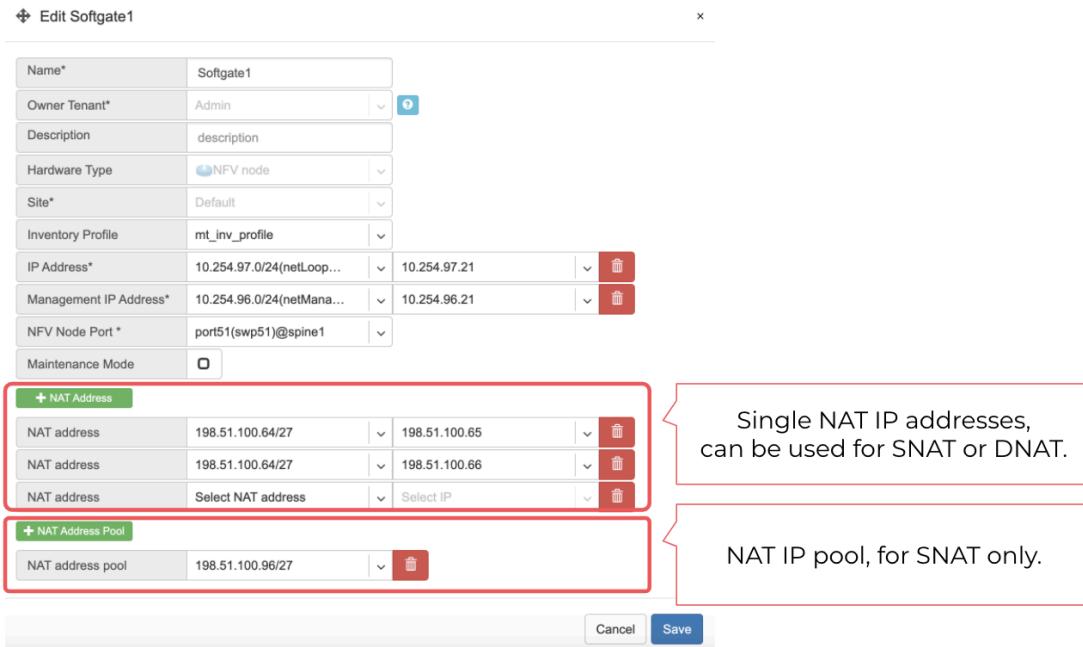
Example: Adding an IP allocation under Net→Subnets.

Name*	NAT IPs
Prefix*	198.51.100.64/27
Type	Assignment
Assign *	Admin
Purpose *	standard

Sites **Default** **Cancel** **Add**

2. Attach NAT IP addresses and/or NAT IP Pools to just one SoftGate node. Other SoftGate Nodes on the same site will automatically add the same NAT IP/Pool resources for proper consistency and high availability.

Example: Adding NAT IP addresses and NAT IP Address Pools to a SoftGate node.



10.2 Defining NAT rules

NAT rules are defined under Net→NAT.

NAT rule fields described:

- **Name** - Unique name.
- **Protocol**
 - **All** - Match any IP protocol.
 - **TCP** - Match TCP traffic and ports.
 - **UDP** - Match UDP traffic and ports
 - **ICMP** - Match ICMP traffic.
- **Action**
 - **SNAT** - Replace the source IP address with specified NAT IP.
 - **DNAT** - Replace the destination IP address and/or destination port with specified NAT IP.
 - **ACCEPT** - Silently forward, typically used to add an exemption to broader SNAT or DNAT rule.
- **Source**
 - **Address** - Source IP address to match.
 - **From port** - Source ports to match starting with this value (TCP/UDP)
 - **To port** - Source ports to match up to this value (TCP/UDP)
- **Destination**
 - **Address** - Destination IP address to match. In the case of DNAT it should be one of the predefined NAT IP addresses.
 - **Port** - For DNAT only, to match a single destination port.
 - **From port** - For SNAT/ACCEPT only. Destination ports to match starting with this value

(TCP/UDP)

- **To port** - For SNAT/ACCEPT only. Destination ports to much up to this value (TCP/UDP)
- **NAT IP** - The global IP address for SNAT to be visible on Public Internet. The internal IP address for DNAT to replace the original destination address with.
- **Status** - Administrative state (enable/disable).
- **Comment** - Free optional comment.

Example: SNAT all hosts on 10.0.0.0/8 to the Internet using 198.51.100.65 as a global IP.

The screenshot shows a 'Add Rule' dialog box with the following fields:

Name*	SNAT for all local nets	Action	SNAT
Protocol	ALL	Status	Enable
Source		Destination	
Source*	10.0.0.0/8	Destination*	0.0.0.0/0
Nat IP*	198.51.100.65/32(Default)	Comment	

At the bottom right are 'Cancel' and 'Add' buttons.

Example: Port forwarding. DNAT the traffic destined to 198.51.100.66:80 to be forwarded to the host 10.0.4.10 on port tcp/1080.

 Add Rule x

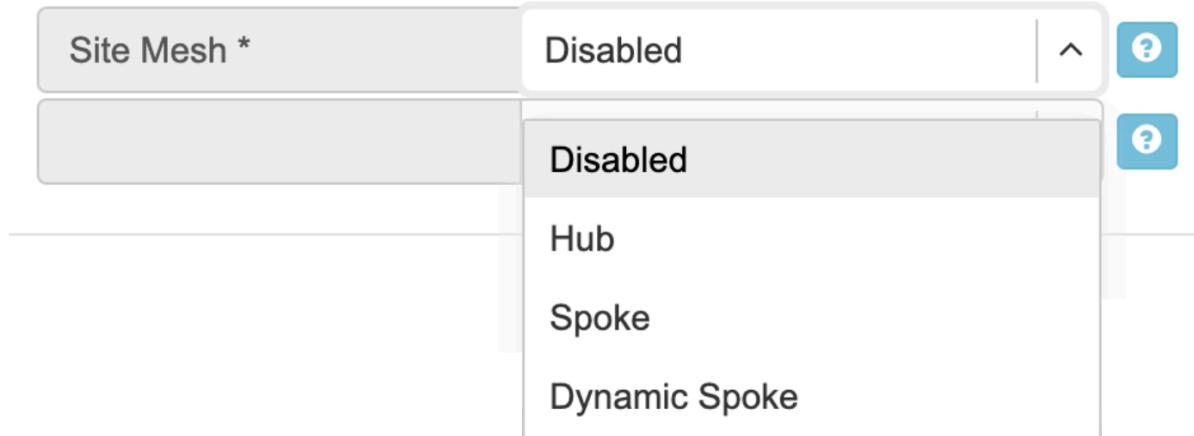
Name*	Port Forwarding	Action	DNAT	▼
Protocol	TCP			
Source		Destination		
Source*	0.0.0.0/0	Destination*	198.51.100.66/32(Defau...	▼
From port*	1	Port*	80	
To port*	65535			
NAT IP*	10.0.4.10	Status	Enable	▼
NAT Port*	1080			
Comment				
Cancel Add				

SiteMesh is a Netris service for site-to-site interconnects over the public Internet. SiteMesh automatically generates configuration for WireGuard to create encrypted tunnels between participating sites and automatically generates a configuration for FRR to run dynamic routing. Hence, sites learn how to reach each other over the mesh WireGuard tunnels. The SiteMesh feature requires a SoftGate node at each participating site.

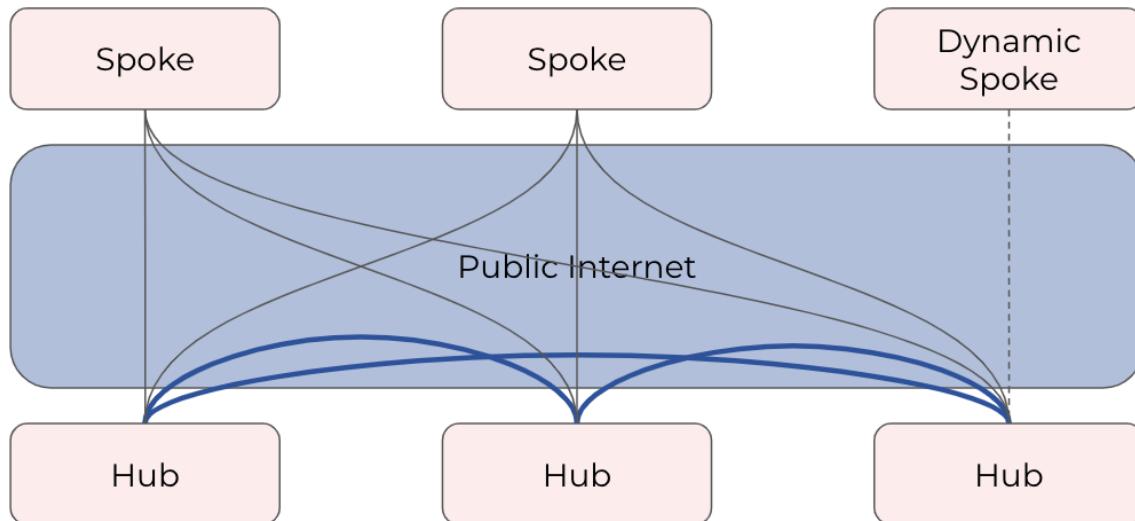
Edit Net->Sites, do declare what sites should form a SiteMesh. See SiteMesh types described below.

- **Disabled** - Do not participate in SiteMesh.
- **Hub** - Hub sites form full-mesh tunnels with all other sites (Hub and non-Hub) and can carry transit traffic for non-Hub sites. (usually major data center sites)
- **Spoke** - Spoke sites form tunnels with all Hub sites. Spoke to Spoke traffic will transit a Hub site. (small data center sites or major office sites)
- **Dynamic Spoke** - Dynamic Spoke is like Spoke, but it will maintain a tunnel only with one Hub site, based on dynamic connectivity measurements underneath and mathematical modeling. (small office sites)

Screenshot: Site Mesh parameter editing a Site under Net→Sites.



You only need to define your site-to-site VPN architecture policy by selecting SiteMesh mode for every site. Netris will generate the WireGuard tunnels (using randomly generated keys, and generate FRR rules to get the dynamic routing to converge).



Check the Net→Site Mesh section for the listing of tunnel statuses.

Screenshot: Listing of SiteMesh tunnels and BGP statuses (Net→Site Mesh)

Remote site	IT	Remote endpoint	IT	Status	IT	Last status cha...	IT	VPN enabled d...	IT
Silicon Valley	IT	SV-NFV1	IT	<div style="background-color: green; color: white; padding: 2px;">BGP: Established Prefix received: 1 Time: 4d13h33m</div>	IT	6/Sep/2020 18:05	IT	3/Sep/2020 04:56	IT

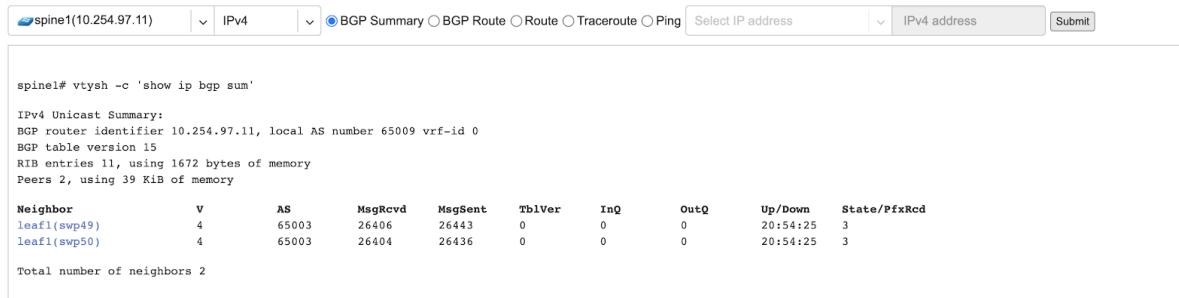
Looking Glass

The Looking Glass Is a GUI-based tool for looking up routing information from a switch or SoftGate perspective. You can access the Looking Glass either from Topology, individually for every device (right click on device → details → Looking Glass), or by navigating to Net→Looking Glass then selecting the device from the top-left dropdown menu.

Looking Glass controls described for IPv4/IPv6 protocol families.

- **BGP Summary** - Shows the summary of BGP adjacencies with neighbors, interface names, prefixes received. You can click on the neighbor name then query for the list of advertised/received prefixes.
- **BGP Route** - Lookup the BGP table (RIB) for the given address.
- **Route** - Lookup switch routing table for the given address.
- **Traceroute** - Conduct a traceroute from the selected device towards the given destination, optionally allowing to determine the source IP address.
- **Ping** - Execute a ping on the selected device towards the given destination, optionally allowing to select the source IP address.

Example: Spine1: listing BGP neighbors and number of received prefixes.



The screenshot shows the Looking Glass interface with the following details:

- Device: spine1(10.254.97.11)
- Protocol: IPv4
- Selected tab: BGP Summary (radio button selected)
- Buttons: Select IP address, IPv4 address, Submit

```

spine1# vtysh -c 'show ip bgp sum'

IPv4 Unicast Summary:
BGP router identifier 10.254.97.11, local AS number 65009 vrf-id 0
BGP table version 15
RIB entries 11, using 1672 bytes of memory
Peers 2, using 39 KiB of memory

Neighbor          V     AS   MsgRcvd   MsgSent    TblVer  InQ      OutQ     Up/Down  State/PfxRcd
leaf1(sw50)       4     65003  26406    26443      0        0        0        20:54:25  3
leaf1(sw49)       4     65003  26404    26436      0        0        0        20:54:25  3

Total number of neighbors 2
  
```

Example: BGP Route - looking up my leaf1 switch's loopback address from spine1's perspective. Spine1 is load balancing between two available paths.

The screenshot shows a web-based interface for a network device named 'spine1(10.254.97.11)'. The top navigation bar includes dropdown menus for 'IPv4' and 'BGP Summary', and buttons for 'BGP Route', 'Route', 'Traceroute', and 'Ping'. A search bar labeled 'Select IP address' contains the value '10.254.97.12', which is highlighted with a red box. A 'Submit' button is also visible.

```

spine1# vtysh -c 'show ip bgp 10.254.97.12'
BGP routing table entry for 10.254.97.12/32
Paths: (2 available, best #2, table default)
  Advertised to non peer-group peers:
    leaf1(swp49) leaf1(swp50)
  65003
    fe80::aa2b:b5ff:fed2:93e3 from leaf1(swp50) (10.254.97.12)
      (fe80::aa2b:b5ff:fed2:93e3) (used)
        Origin incomplete, metric 0, valid, external, multipath
        Community: 0:1
        AddPath ID: RX 0, TX 88
        Last update: Tue Jan 5 09:37:28 2021

  65003
    fe80::aa2b:b5ff:fed2:93df from leaf1(swp49) (10.254.97.12)
      (fe80::aa2b:b5ff:fed2:93df) (used)
        Origin incomplete, metric 0, valid, external, multipath, bestpath-from-AS 65003, best
        Community: 0:1
        AddPath ID: RX 0, TX 87
        Last update: Tue Jan 5 09:37:28 2021

```

Example: Ping.

The screenshot shows the same interface as above, with the 'Ping' button selected. The search bar still contains '10.254.97.12'. The output of the 'ping' command is displayed:

```

spine1# ping -c 5 -I 10.254.97.11 10.254.97.12
PING 10.254.97.12 (10.254.97.12) from 10.254.97.11 : 56(84) bytes of data.
64 bytes from 10.254.97.12: icmp_seq=1 ttl=64 time=0.271 ms
64 bytes from 10.254.97.12: icmp_seq=2 ttl=64 time=0.236 ms
64 bytes from 10.254.97.12: icmp_seq=3 ttl=64 time=0.250 ms
64 bytes from 10.254.97.12: icmp_seq=4 ttl=64 time=0.278 ms
64 bytes from 10.254.97.12: icmp_seq=5 ttl=64 time=0.259 ms

--- 10.254.97.12 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4000ms
rtt min/avg/max/mdev = 0.236/0.258/0.278/0.025 ms

```

Looking Glass controls described for the EVPN family.

- **BGP Summary** - Show brief summary of BGP adjacencies with neighbors, interface names, and EVPN prefixes received.
- **VNI** - List VNIs learned.
- **BGP EVPN** - List detailed EVPN routing information optionally for the given route distinguisher.
- **MAC table** - List MAC address table for the given VNI.

Example: Listing of adjacent BGP neighbors and number of EVPN prefixes received.

The screenshot shows the 'BGP Summary' section of the interface. The search bar contains '10.254.97.11'. The output of the 'show ip bgp sum' command is displayed:

```

spine1# vtysh -c 'show ip bgp sum'

IPv4 Unicast Summary:
BGP router identifier 10.254.97.11, local AS number 65009 vrf-id 0
BGP table version 15
RIB entries 11, using 1672 bytes of memory
Peers 2, using 39 KiB of memory

Neighbor          V       AS     MsgRcvd   MsgSent   TblVer  InQ     OutQ     Up/Down   State/PfxRcd
leaf1(swp49)      4       65003  26406    26443    0        0        0        20:54:25  3
leaf1(swp50)      4       65003  26404    26436    0        0        0        20:54:25  3

Total number of neighbors 2

```

Example: Listing MAC addresses on VNI 2.

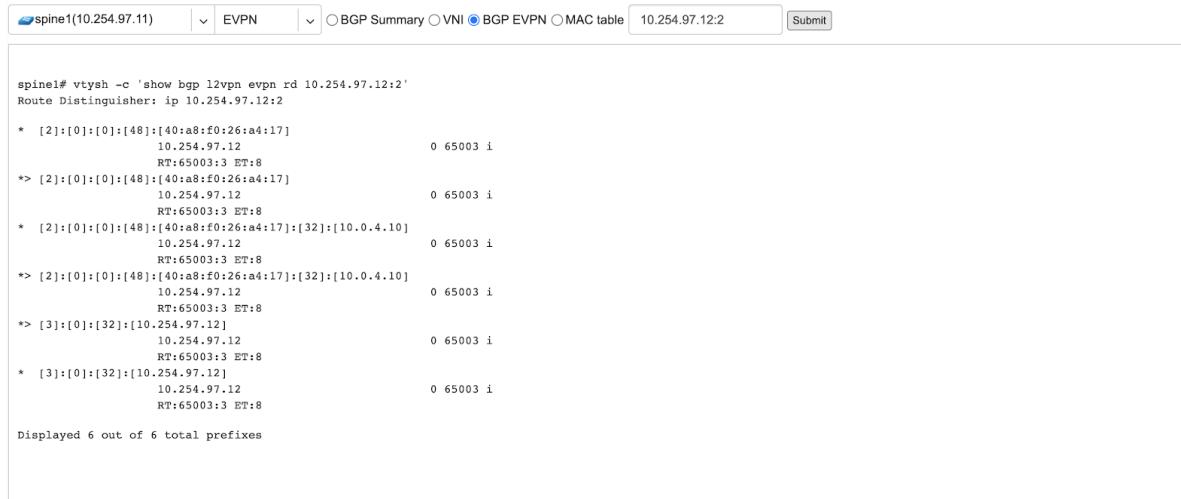
The screenshot shows the 'MAC table' section of the interface. The search bar contains '10.254.97.11' and the dropdown menu is set to 'EVPN'. The input field contains the value '2', which is highlighted with a red box. The 'Submit' button is also visible. The output of the 'show evpn mac vni 2' command is displayed:

```

spine1# vtysh -c 'show evpn mac vni 2'
Number of MACs (local and remote) known for this VNI: 2
MAC           Type   Intf/Remote VTEP      VLAN Seq #
02:00:5e:00:00:01 local   swp52.2          0/0
2c:76:8a:52:3c:4b remote  10.254.97.12      0/0

```

Example: EVPN routing information listing for a specified route distinguisher.



The screenshot shows a network configuration interface with the following details:

- Device: spine1(10.254.97.11)
- Selected tab: EVPN
- Filter: BGP Summary (selected)
- Route Distinguisher: ip 10.254.97.12:2
- Address: 10.254.97.12
- RT: 65003:3
- ET: 8
- Number of prefixes: 6
- Output: Detailed BGP route table output for the specified RD.

```

spine1# vtysh -c 'show bgp l2vpn evpn rd 10.254.97.12:2'
Route Distinguisher: ip 10.254.97.12:2
* [2]:[0]:[0]:[48]:[40:a8:f0:26:a4:17]
    10.254.97.12          0 65003 i
    RT:65003:3 ET:8
*> [2]:[0]:[0]:[48]:[40:a8:f0:26:a4:17]
    10.254.97.12          0 65003 i
    RT:65003:3 ET:8
* [2]:[0]:[0]:[48]:[40:a8:f0:26:a4:17]:[32]:[10.0.4.10]
    10.254.97.12          0 65003 i
    RT:65003:3 ET:8
*> [2]:[0]:[0]:[48]:[40:a8:f0:26:a4:17]:[32]:[10.0.4.10]
    10.254.97.12          0 65003 i
    RT:65003:3 ET:8
*> [3]:[0]:[32]:[10.254.97.12]
    10.254.97.12          0 65003 i
    RT:65003:3 ET:8
* [3]:[0]:[32]:[10.254.97.12]
    10.254.97.12          0 65003 i
    RT:65003:3 ET:8
Displayed 6 out of 6 total prefixes

```


V-NET

V-NET is a virtual networking service. V-NETs can be used for Layer-2 (unrouted) or Layer-3 (routed) virtual network segments involving switch ports anywhere on the switch fabric. V-NETs can be created and managed by a single tenant (single team) or created and managed collaboratively by multiple tenants (different teams inside and/or outside the organization).

Automatically, Netris will configure a VXLAN with an EVPN control plane over an unnumbered BGP Layer-3 underlay network and organize the high availability for the default gateway behind the scenes.

V-Net fields.

- **Name** - Unique name for the V-NET.
- **Owner** - Tenant, who can make any changes to current V-NET.
- **V-Net state** - Active/Disable state for entire V-NET.
- **VLAN aware** - Enable VLAN aware bridge, use only in rare cases, if otherwise is not possible.
- **Guest tenants** - List of tenants allowed to add/edit/remove ports to the V-Net but not manage other parameters.
- **Sites** - Ports from these sites will be allowed to participate in the V-Net. (Multi-site circuits would require backbone connectivity between sites).
- **IPv4 Gateway** - IPv4 address to be used as a default gateway in this V-NET. Should be configured under Net→Subnets as an assignment, assigned to the owner tenant, and available in the site where V-NET is intended to span.
- **IPv6 Gateway** - IPv6 address to be used as a default gateway in this V-NET. Should be configured under Net→Subnets as an assignment, assigned to the owner tenant, and available in the site or sites where V-NET is intended to span.
- **Port** - Physical Switch Port anywhere on the network. Switch Port should be assigned to the owner or guest tenant under Net→Switch Ports.
 - **Enabled** - Enable or disable individual Switch Port under current V-NET
 - **Port Name** - Switch Port format: <alias>(swp<number>)@<switch name>
 - **VLAN ID / Untag** - Specify a VLAN ID for tagging traffic on a per-port basis or set Untag not to use tagging on a particular port. VLAN tags are only significant on each port's ingress/egress unless VLAN aware mode is used.
 - **LAG Mode** - Allows for active-standby dual-homing, assuming LAG configuration on the remote end. Active/active dual homing will be enabled in future releases (dependence on SVI support by NOSes).

Tip: Many switches can't autodetect old 1Gbps ports. If attaching hosts with 1Gbps ports to 10Gbps switch ports, you'll need to change the speed for a given Switch Port from Auto(default) to 1Gbps. You can edit a port in Net→Switch Ports individually or in bulk.

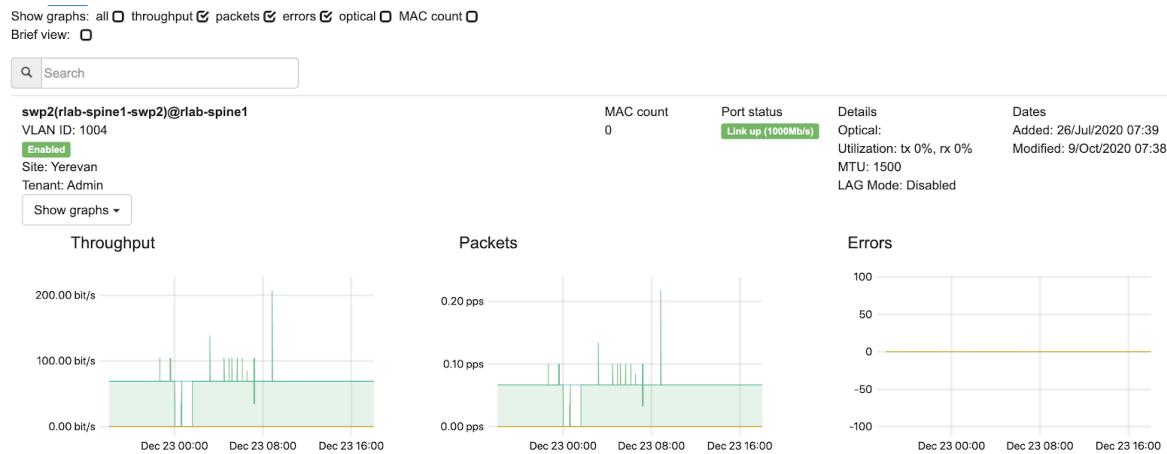
Example: Adding a new V-NET.

The screenshot shows the 'Add new V-Net' dialog. The 'Name*' field is set to 'Servers1'. The 'Owner*' field is set to 'Admin'. The 'V-Net state' is 'Active'. The 'VLAN aware' checkbox is unchecked. The 'Guest tenants' dropdown is set to 'Select Tenants'. The 'Sites*' dropdown is set to 'All selected'. Below this, there's an 'IPv4 Gateway' section with an entry '10.0.3.0/24(Servers1) 10.0.3.1'. Another 'IPv6 Gateway' section is present. At the bottom, there's a table for 'Port' configuration with three entries: 'port1(port1)@leaf1 (Admin)', 'port1(port1)@leaf2 (Admin)', and 'port2(port2)@leaf1 (Admin)'. The table includes columns for Enabled, MAC address, Untag, LAG Mode, and delete icons. A message at the bottom says '0 - 3 (3) records.' and a 'Rows' dropdown is set to '5'. At the bottom right are 'Cancel' and 'Add' buttons.

Example: Listing of V-NETs.

The screenshot shows the V-Net listing page. The table has columns for V-Net name, State, IP address, Created Date, Modified Date, and a more button. The first row is 'i-NET' (State: Active, IP: 194.61.91.17/29, Created: 26/Jul/2020 07:38, Modified: 26/Jul/2020 08:08). The second row is 'k8s-VNET' (State: Active, IP: 192.168.100.1/24, Created: 2/Nov/2020 07:31, Modified: 23/Dec/2020 04:24). The third row is 'K82' (State: Active, IP: 192.168.101.1/24, Created: 14/Apr/2020 06:38, Modified: 10/Jun/2020 02:13). Each row has a 'Ports count' detail below it. The top bar includes filters for Clear, Search, All Sites, and buttons for Export as csv, Collapse All, Add, and a user icon.

Example: Expanded view of a V-NET listing.



Kubenet

Kubenet is a network service purpose-built for Kubernetes cluster nodes. Netris integrates with Kube API to provide on-demand load balancer and other Kubernetes specific networking features. Netris Kubenet is designed to complement Kubernetes CNI networking and provide a cloud-like user experience to local Kubernetes clusters.

The Gateway and Switch Port part of Kubenet is similar to the V-NET. In fact, it is leveraging a V-NET. Kubeconfig is for granting Netris Controller access to your Kube API. Kubenet therefore, dynamically leverages Netris L4LB and other services based on events that Netris kube-watcher (Kube API integration adapter) watches in your Kube API.

Description of Kubenet fields.

- **Name** - Unique name for the Kubenet.
- **Tenant** - Tenant, who can make any changes to current Kubenet.
- **Site** - Site where Kubernetes cluster belongs.
- **State** - Active/Disable state for particular Kubenet service.
- **IPv4 Gateway** - IPv4 address to be used as a default gateway for current Kubenet.
- **Port** - Physical Switch Port anywhere on the switch fabric. Switch Port should be assigned to the owner tenant under Net→Switch Ports.
 - **Enabled** - Enable or disable individual Switch Port under current Kubenet.
 - **Port Name** - Switch Port format: <alias>(swp<number>)@<switch name>
 - **VLAN ID / Untag** - Specify a VLAN ID for tagging traffic on a per-port basis or set to Untag not to use tagging on a particular port.
- **Kubeconfig** - After installing the Kubernetes cluster, add your Kube config for granting Netris at least read-only access to the Kube API.

Tip: Many switches can't autodetect old 1Gbps ports. If attaching hosts with 1Gbps ports to 10Gbps switch ports, you'll need to change the speed for a given Switch Port from Auto(default) to 1Gbps. You can edit a port in Net→Switch Ports individually or in bulk.

Example: Adding a new Kubenet service.

❖ Add Kubenet

Port	Status	Tagging
port3(port3)@leaf1	Enabled	Untagged
port4(port4)@leaf1	Enabled	Untagged
port5(port5)@leaf1	Enabled	Untagged

Once Netris Controller establishes a connection with Kube API, status will reflect on the listing.

Screenshot: Listing of Kubenet services. Kube API connection is successful.

Screenshot: Physical Switch Port statuses.

Screenshot: Statuses of on-demand load balancers (type: load-balancer)

ROH (Routing on the Host)

To create more resilient and higher-performance data centers, some companies leverage the Linux ecosystem to run routing protocols directly to their servers. Known as ROH (Routing on the Host).

In ROH architectures, servers use a routing demon to establish a BGP adjacency with the switch fabric on every physical link. ROH can run on bare metal servers, VMs, and even containers. The most commonly used routing daemon is FRR.

Hosts connected to the network in ROH architecture don't have IP addresses on a shared Ethernet segment; instead IP address is configured on the loopback interface and advertised over all BGP links towards switch fabric. Thus, leveraging the Layer-3 network throughout the entire network down the servers.

ROH architecture with Netris allows for leveraging ECMP load balancing capabilities of the switching hardware for the high-performance server load balancing (described in L3 Load Balancer section). For each instance of ROH, you'll need to create an ROH entry in Netris Controller.

Description of ROH instance fields:

- **Name** - Unique name for the ROH instance.
- **Site** - Site where the current ROH instance belongs.
- **Type** - Physical Server, for all servers forming a BGP adjacency directly with the switch fabric. Hypervisor, for using the hypervisor as an interim router. Proxmox is currently the only supported hypervisor.
- **ROH Routing Profile** - ROH Routing profile defines what set of routing prefixes to be advertised to ROH instances.
 - **Default route only (a most common choice)** - Will advertise 0.0.0.0/0 + loopback address of the physically connected switch.
 - **Default + Aggregate** - Will add prefixes of defined assignments + "Default" profile.
 - **Full table** - Will advertise all prefixes available in the routing table of the connected switch.
 - **Inherit** - will inherit policy from site objects defined under Net→Sites.
- **Legacy Mode** - Switch from default zero-config mode to using /30 IP addresses. Use for MS Windows Servers or other OS that doesn't support FRR.
- **+Port** - Physical Switch Ports anywhere on the network.
- **+IPv4** - IPv4 addresses for the loopback interface.
- **+Inbound Prefix List** - List of additional prefixes that the ROH server may advertise. Sometimes used to advertise container or VM networks.

Tip: Many switches can't autodetect old 1Gbps ports. If attaching hosts with 1Gbps ports to 10Gbps switch ports, you'll need to change the speed for a given Switch Port from Auto(default) to 1Gbps. You can edit a port in Net→Switch Ports individually or in bulk.

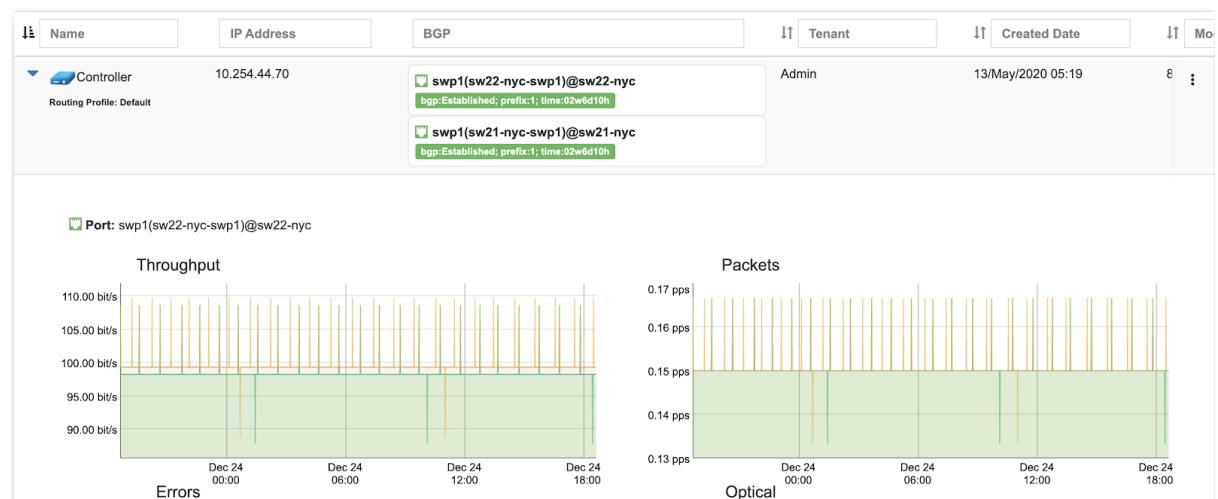
Example: Adding an ROH instance. (Yes, you can use A.B.C.0/32 and A.B.C.255/32)

❖ Add new Instance ×

Name*	<input type="text" value="Name"/>	?								
Site*	<input type="text" value="Default"/>	▼								
Type*	<input type="text" value="Physical Server"/>	▼								
ROH Routing Profile*	<input type="text" value="Inherit"/>	?								
Legacy Mode	<input type="checkbox"/>									
+Port* ▼										
<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;"><input type="text" value="port1(port1)leaf1"/></td> <td style="width: 10%; text-align: right;">▼</td> <td style="width: 10%; text-align: right;">trash</td> <td style="width: 60%; text-align: right;">BGP: Unknown</td> </tr> <tr> <td><input type="text" value="port1(port1)leaf2"/></td> <td style="text-align: right;">▼</td> <td style="text-align: right;">trash</td> <td style="text-align: right;">BGP: Unknown</td> </tr> </table>			<input type="text" value="port1(port1)leaf1"/>	▼	trash	BGP: Unknown	<input type="text" value="port1(port1)leaf2"/>	▼	trash	BGP: Unknown
<input type="text" value="port1(port1)leaf1"/>	▼	trash	BGP: Unknown							
<input type="text" value="port1(port1)leaf2"/>	▼	trash	BGP: Unknown							
+IPv4 ▼										
<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;"><input type="text" value="10.0.5.0/24(ROH)"/></td> <td style="width: 10%; text-align: right;">▼</td> <td style="width: 30%; text-align: right;"><input type="checkbox"/> Anycast</td> <td style="width: 30%; text-align: right;">trash</td> </tr> </table>			<input type="text" value="10.0.5.0/24(ROH)"/>	▼	<input type="checkbox"/> Anycast	trash				
<input type="text" value="10.0.5.0/24(ROH)"/>	▼	<input type="checkbox"/> Anycast	trash							
+Inbound Prefix List ?										
<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;"><input type="text" value="permit"/></td> <td style="width: 10%; text-align: right;">▼</td> <td style="width: 30%; text-align: right;"><input type="checkbox"/> le/ge range</td> <td style="width: 30%; text-align: right;">trash</td> </tr> </table>			<input type="text" value="permit"/>	▼	<input type="checkbox"/> le/ge range	trash				
<input type="text" value="permit"/>	▼	<input type="checkbox"/> le/ge range	trash							

Cancel Save

Screenshot: Expanded view of ROH listing. BGP sessions are up, and the expected IP is in fact received from the actual ROH server. Traffic stats are available per port.



L3 Load Balancer (Anycast LB)

L3 (Anycast) load balancer is leveraging ECMP load balancing and hashing capability of spine and leaf switches to deliver line-rate server load balancing with health checks.

ROH servers, besides advertising their unicast (unique) loopback IP address, need to configure and advertise an additional anycast (the same IP) IP address. Unicast IP address is used for connecting to each individual server.

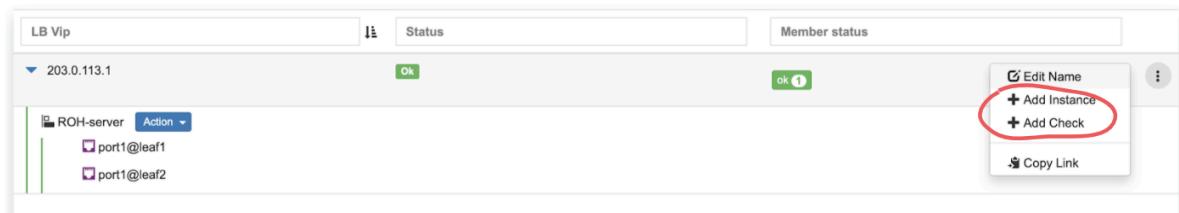
End-user traffic should be destined to the anycast IP address. Switch fabric will ECMP load balance the traffic towards every server, as well as will hash based on IP/Protocol/Port such that TCP sessions will keep complete between given end-user and server pair. Optionally health checks are available to reroute the traffic away in the event of application failure.

To configure L3 (Anycast) load balancing, edit an existing ROH instance entry and add an extra IPv4 address, and select Anycast. This will create a service under Services→Load Balancer and permit using the Anycast IP address in multiple ROH instances.

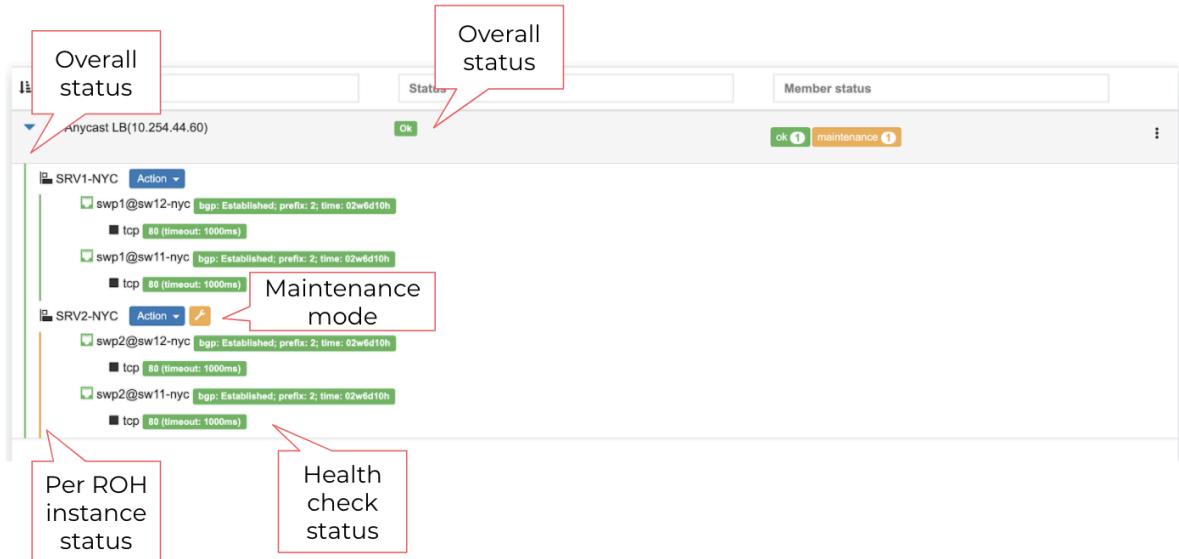
Example: Adding an Anycast IPv4 address

Name*	ROH-server	
Site*	Default	
Type*	Physical Server	
ROH Routing Profile*	Default	
Legacy Mode	<input type="checkbox"/>	
+Port*		
port1(port1)@leaf1		BGP: Unknown
port1(port1)@leaf2		BGP: Unknown
+IPv4		
10.0.5.0/24(ROH)		10.0.5.0 <input type="checkbox"/> Anycast
203.0.113.0/28(ROH ...)		203.0.113.1 <input checked="" type="checkbox"/> Anycast
<input type="button" value="Inbound Prefix List"/>		
<input type="button" value="Cancel"/> <input type="button" value="Save"/>		

Example: Under Services→Load Balancer, you can find the listing of L3 (Anycast) Load Balancers, service statuses, and you can add/remove more ROH instances and/or health checks.



Screenshot: L3 (Anycast) Load Balancer listing.



L4 Load Balancer (L4LB)

Netris L4 Load Balancer (L4LB) is leveraging SoftGate(Linux router) nodes for providing Layer-4 load balancing service, including on-demand cloud load balancer with native integration with Kubernetes.

16.1 Enabling L4LB service

L4 Load Balancer service requires at least one SoftGate node to be available in a given Site, as well as at least one IP address assignment (purpose=load balancer).

The IP address pool for L4LB can be defined in the Net→Subnets section by adding an Allocation and setting the purpose field to 'load-balancer.' You can define multiple IP pools for L4LB at any given Site. See the below example.

Example: Adding a load-balancer IP pool assignment.

❖ Add Assignment

Name*	L4LB IP pool	?
Prefix*	203.0.113.192/26	?
Type	Assignment	?
Assign *	Admin	?
Purpose *	load-balancer	?

+Sites [?](#) Default [X](#)

This IP pool is dedicated for the Site Default.

IP resource is assigned to Admin tenant, but L4LB service can be requested by any tenant.

Mark the IP prefix to be used for L4 Load Balancer service.

Cancel [Add](#)

Screenshot: Listing of Net→Subnets after adding a load-balancer assignment

Description	Subnets	Tenant	Site	Purpose	
controller_lo	169.254.254.254/32		Default		⋮
private	10.0.0.0/16		Default		⋮
netManagement	10.0.0.0/24	Admin	Default	standard	⋮
netLoopbacks	10.0.1.0/24	Admin	Default	standard	⋮
Public	203.0.113.0/24		Default		⋮
L4LB IP pool	203.0.113.192/26	Admin	Default	load-balancer	⋮

16.2 Consuming L4LB service

This guide describes how to request an L4 Load Balancer using GUI. For Kubernetes integration, check the Kubenet section.

Click +add under Services→L4 Load Balancer to request an L4LB service.

Add new L4 Load Balancer fields are described below:

General fields

- **Name*** - Unique name.
- **Protocol*** - TCP or UDP.
- **Tenant*** - Requestor Tenant should have access to the backend IP space.
- **Site*** - Site where L4LB service is being requested for. Backends should belong on this site.
- **State*** - Administrative state.

Frontend

- **Address*** - Frontend IP address to be exposed for this L4LB service. “Assign automatically” will provide the next available IP address from the defined load-balancer pool. Alternatively, users can select manually from the list of available addresses.
- **Port*** - TCP or UDP port to be exposed.

Health-check

- **Type*** - Probe backends on service availability.
 - **None** - load balance unconditionally.
 - **TCP** - probe backend service availability through TCP connect checks.
 - **HTTP** - probe backend service availability through http GET checks.
- **Timeout(ms)*** - Probe timeout in milliseconds.
- **Request path*** - Http request path.

Backend

- **+Add** - add a backend host.
- **Address** - IP address of the backend host.
- **Port** - Service port on the backend host.
- **Enabled** - Administrative state of particular backend.

Example: Requesting an L4 Load Balancer service.

Add L4 Load Balancer

Name *	Name	TCP or UDP
Protocol *	TCP	Tenant requesting the L4LB
Tenant *	Admin	
Site *	Default	Site with L4LB support
State *	Active	

Frontend		Health-check
Address *	Assign Automatically	Provide frontend IP address automatically from the pool
Port *	80	Port to be exposed on the frontend
Type *	TCP	Timeout(ms) * 500
		Request path * /

Backend		
+ Add		
10.0.3.11	1080	Enabled
10.0.3.10	1080	Enabled

Backend addresses

Service port backend is listening

Cancel Add

Example: Listing of L4 Load Balancer services

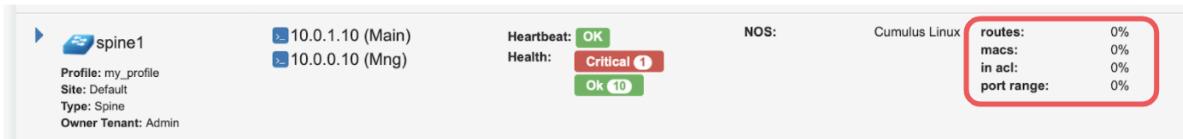
Service name		Frontend	Status	Backends	Tenant	Site	Created Date
83e4ba36-8aab-42b3-...	10.200.0.1:80	OK	Active (3)	Admin	San Francisco	17/Dec/20...	⋮
Protocol: TCP Kubenet: k8s							
Health-check: TCP Timeout: 2000 Backend addresses 192.168.100.100:32176 192.168.100.101:32176 192.168.100.102:32176 Active connection succeed Active connection succeed Active connection succeed							
83e4ba36-8aab-42b3-...	10.200.0.1:443	OK	Active (3)	Admin	San Francisco	17/Dec/20...	⋮
Protocol: TCP Kubenet: k8s							
Art-test	10.200.0.150:150	OK	Active (2)	Admin	San Francisco	23/Nov/20...	⋮
Protocol: TCP							
Art-test-2	10.200.0.222:222	Disabled	Inactive (2)	Admin	San Francisco	23/Nov/20...	⋮
Protocol: TCP							
0 - 5 (5) records.							
100 ▲ Rows							

Access Control Lists (ACL)

Netris supports ACLs for switch network access control. (ACL and ACL2.0) ACL is for defining network access lists in a source IP: Port, destination IP: Port format. ACL2.0 is an object-oriented service way of describing network access.

Both ACL and ACL2.0 services support tenant/RBAC based approval workflows. Access control lists execute in switch hardware providing line-rate performance for security enforcement. It's important to keep in mind that the number of ACLs is limited to the limited size of TCAM of network switches.

Screenshot: TCAM utilization can be seen under Net→Inventory



Netris is applying several optimization algorithms to minimize the usage of TCAM while achieving the user-defined requirements.

17.1 ACL Default Policy.

The ACL default policy is to permit all hosts to communicate with each other. You can change the default policy on a per Site basis by editing the Site features under Net→Sites. Once the "ACL Default Policy" is changed to "Deny," the given site will start dropping any traffic unless specific communication is permitted through ACL or ACL2.0 rules.

Example: Changing "ACL Default Policy" for the site "siteDefault".

❖ Edit Site siteDefault

Name *	siteDefault	
Border(Leaf) Switch ASN *	65008	
Spine Switch ASN *	65009	
TOR(Leaf) Switch ASN *	65003	
Hypervisor ASN *	65002	
ROH instance ASN *	65500	
ROH virtual instance ASN *	65501	
ROH Routing Profile *	Default	
Site Mesh *	Disabled	
ACL Default Policy	Permit	

17.2 ACL rules

ACL rules can be created, listed, edited, approved under Services→ACL.

Description of ACL fields. General

- **Name*** - Unique name for the ACL entry.
- **Protocol*** - IP protocol to match.
 - All - Any IP protocols.
 - IP - Specific IP protocol number.
 - TCP - TCP.
 - UDP - UDP.
 - ICMP ALL - Any IPv4 ICMP protocol.
 - ICMP Custom - Custom IPv4 ICMP code.
 - ICMPv6 ALL - Any IPv6 ICMP protocol.
 - ICMPv6 Custom - Custom IPv6 ICMP code.
- **Active Until** - Disable this rule at the defined date/time.
- **Action** - Permit or Deny forwarding of matched packets.
- **Established/Reverse** - For TCP, also match reverse packets except with TCP SYN flag. For non-TCP, also generate a reverse rule with swapped source/destination.

Source/Destination - Source and destination addresses and ports to match.

- **Source*** IPv4/IPv6 - IPv4/IPv6 address.
- **Ports Type***
 - Port Range - Match on the port or a port range defined in this window.

- **Port Group** - Match on a group of ports defined under Services→ ACL Port Group.
- **From Port*** - Port range starting from.
- **To Port*** - Port range ending with.
- **Comment** - Descriptive comment, commonly used for approval workflows.
- **Check button** - Check if Another ACL on the system already permits the described network access.

Example: Permit hosts in 10.0.3.0/24 to access hosts in 10.0.5.0/24 by SSH, also permit the return traffic (Established).

Name*	commonServers_to_ROH																									
Protocol	TCP	<input type="button" value="▼"/>																								
<input type="checkbox"/> Active Until Mon Dec 28 2020 18:45:58 GMT																										
Action	Permit																									
<input checked="" type="checkbox"/> Established																										
Source <table border="1"> <tr> <td>Source*</td> <td colspan="2">10.0.3.0/24</td> </tr> <tr> <td>Ports Type</td> <td>Port Range</td> <td><input type="button" value="▼"/></td> </tr> <tr> <td>From port*</td> <td colspan="2">1</td> </tr> <tr> <td>To port*</td> <td colspan="2">65535</td> </tr> </table> Destination <table border="1"> <tr> <td>Destination*</td> <td colspan="2">10.0.5.0/24</td> </tr> <tr> <td>Ports Type</td> <td>Port Range</td> <td><input type="button" value="▼"/></td> </tr> <tr> <td>From port*</td> <td colspan="2">22</td> </tr> <tr> <td>To port*</td> <td colspan="2">22</td> </tr> </table>			Source*	10.0.3.0/24		Ports Type	Port Range	<input type="button" value="▼"/>	From port*	1		To port*	65535		Destination*	10.0.5.0/24		Ports Type	Port Range	<input type="button" value="▼"/>	From port*	22		To port*	22	
Source*	10.0.3.0/24																									
Ports Type	Port Range	<input type="button" value="▼"/>																								
From port*	1																									
To port*	65535																									
Destination*	10.0.5.0/24																									
Ports Type	Port Range	<input type="button" value="▼"/>																								
From port*	22																									
To port*	22																									
Comment <input type="text"/>																										
<input type="button" value="Cancel"/> <input type="button" value="Check"/> <input type="button" value="Save"/>																										

Example: “Check” shows that requested access is already provided by a broader ACL rule.

Name*	commonServers_to_ROH		<input type="checkbox"/> Active Until	Mon Dec 28 2020 19:04:55 GMT	
Protocol	TCP		Action	Permit	
<input checked="" type="checkbox"/> Established					
Source			Destination		
Source*	10.0.3.0/24		Destination*	10.0.5.15/32	
Ports Type	Port Range		Ports Type	Port Range	
From port*	1		From port*	22	
To port*	65535		To port*	22	
Comment					
<p>commonServers_to_R 10.0.3.0/24 1-65535 10.0.5.0/24 22 tcp permit</p>					
<input type="button" value="Cancel"/> <input type="button" value="Check"/> <input type="button" value="Save"/>					

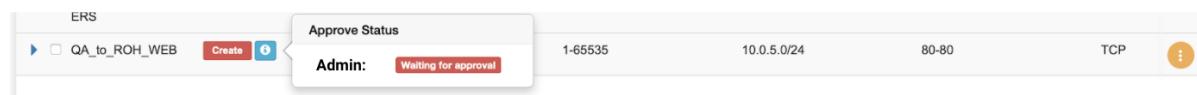
17.3 ACL approval workflow

When one Tenant (one team) needs to get network access to resources under the responsibility of another Tenant (another team), an ACL can be created but will activate only after approval of the Tenant responsible for the destination address resources. See the below example.

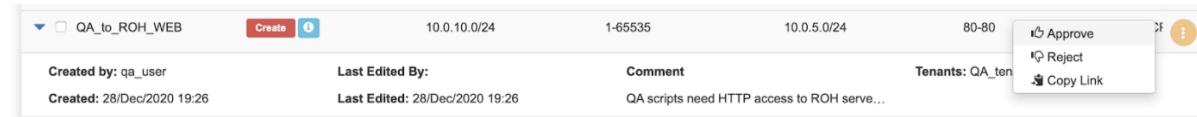
Example: User representing QA_tenant is creating an ACL where source belongs to QA_tenant, but destination belongs to the Admin tenant.

Name*	QA_to_ROH_WEB	
Protocol	TCP	<input type="checkbox"/> Active Until Mon Dec 28 2020 19:26:35 GMT
<input checked="" type="checkbox"/> Established		Action Permit
Source	Destination	
Source*	10.0.10.0/24	
Ports Type	Port Range	<input type="checkbox"/> Port Range
From port*	1	
To port*	80	
Comment		
QA scripts need HTTP access to ROH servers for auto testing.		
<input type="button" value="Cancel"/> <input type="button" value="Check"/> <input type="button" value="Save"/>		

Screenshot: ACL stays in “waiting for approval” state until approved.



Screenshot: Users of tenant Admin, receive a notification in the GUI, and optionally by email. Then one can review the access request and either approve or reject it.



Screenshot: Once approved, users of both tenants will see the ACL in the “Active” state, and soon Netris Agents will push the appropriate config throughout the switch fabric.



17.4 The sequence order of ACL rules

1. User-defined Deny Rules
2. User-defined Permit Rules
3. Deny the rest

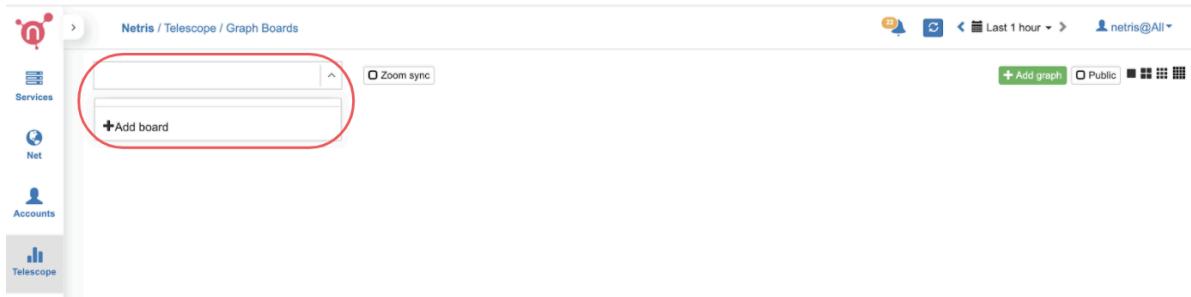
Visibility (Telescope)

18.1 Graph Boards

You can create custom graph boards with data sources available in different parts of the system. You can even sum multiple graphs and visualize them in a single view.

To start with Graph Boards, first, you need to add a new Graph Board.

1. Navigate to Telescope→Graph Boards, open the dropdown menu in the top left corner, then click +Add board.



2. Type a name and assign it to one of the tenants that you manage. Later on, you can optionally mark the Graph Board as public if you want the particular board to be visible to all users across multiple tenants.

Create board	
Name	my_graph_board
Tenant	Admin
<input type="button" value="Cancel"/> <input type="button" value="Add"/>	

Now you can add graphs by clicking +Add graph.

Description of +Add graph fields:

- **Title** - Title for the new graph.
- **Type** - Type of data source.

- Bps - Traffic bits per second.
- Pps - Traffic packets per second.
- Errors - Errors per second.
- Optical - Optical signal statistics/history.
- MAC Count - History of the number of MAC addresses on the port.
- **Function** - Currently, only summing is supported.
- **+Member** - Add data sources by service (E-BGP, V-NET, etc..) or by Switch Port.

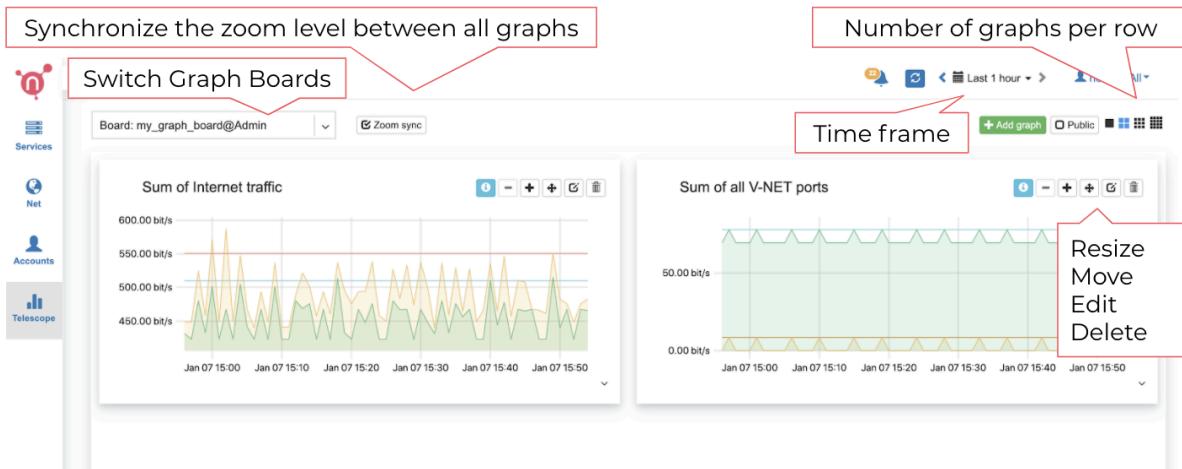
Example: Sum of traffic on two ISP(Iris1 + Iris2) links.

Title*	Sum of Internet traffic	
Type*	Bps	<input type="button" value="▼"/>
Function*	Sum	<input type="button" value="▼"/>
+Member <input type="button" value="▼"/>		
BGP bps	Iris1	<input type="button" value="Delete"/>
BGP bps	Iris2	<input type="button" value="Delete"/>

Example: Sum of the traffic on all ports under the service called “my V-NET”

Title*	Sum of all V-NET ports	
Type*	Bps	<input type="button" value="▼"/>
Function*	Sum	<input type="button" value="▼"/>
+Member <input type="button" value="▼"/>		
V-Net bps	my V-NET	<input type="button" value="Delete"/>

Screenshot: Listing of a Graph Board with the explanation of the controls.



18.2 API Logs

Comprehensive logging of all API calls sent to Netris Controller with the ability to search by various attributes, sort by each column, and filter by method type.

18.3 Dashboard

Netris, besides automatic configuration, also provides automatic monitoring of the entire network without the need for configuration of the monitoring systems.

Telescope→Dashboard summarizes Network Health, which can also be accessed by clicking on the Netris icon in the top left corner.

Description of the pie charts.

- **Hardware Health** - summary of CPU, RAM, disk utilization. Statuses of power supplies, fans, temperature sensors, critical system services, and time synchronization. Statuses of switch port link, utilization, optical signal levels, and BGP sessions.
- **E-BGP** - Statuses of external BGP sessions.
- **LB VIP** - Statuses of Load Balancer frontend / VIP availability.
- **LB Members** - Statuses of Load Balancer backend members.

By clicking on each title you can see the details of the checks on the right side.

Screenshot: Dashboard showing details of “Hardware Health.”

Monitoring summary grouped by type. Click on title to see details on the right column.

Filter by status: OK, Warning, Critical.

Name	Check name	Duration	Last check	Message
controller	check_port	12h 47m 23s	7/Jan/2021 17:41	swp10 port is UP, 0% RX Utilized of 1 Gbps, 0% TX Utilized of 1 Gbps
leaf1	check_port	12h 47m 23s	7/Jan/2021 17:41	swp11 port is DOWN, 0% RX Utilized of 1 Gbps, 0% TX Utilized of 1 Gbps
spine1	check_port	12h 47m 23s	7/Jan/2021 17:41	swp12 port is DOWN, 0% RX Utilized of 1 Gbps, 0% TX Utilized of 1 Gbps
spine1	check_port	12h 47m 23s	7/Jan/2021 17:41	swp13 port is DOWN, 0% RX Utilized of 1 Gbps, 0% TX Utilized of 1 Gbps
spine1	check_port	12h 47m 23s	7/Jan/2021 17:41	swp14 port is DOWN, 0% RX Utilized of 1 Gbps, 0% TX Utilized of 1 Gbps

Port up/down state can be set to “Save as normal.” So the system will alarm only if the actual state is different from the saved as the normal state.

Screenshot: “Save as normal” on selected ports.

Save as normal

Name	Check name	Duration	Last check	Message
controller	check_port	1h 45m 56s	29/Dec/2020 15:16	swp10 port is DOWN
spine1	check_port	1h 45m 56s	29/Dec/2020 15:16	swp11 port is DOWN
spine1	check_port	1h 45m 56s	29/Dec/2020 15:16	swp12 port is DOWN
spine1	check_port	1h 45m 56s	29/Dec/2020 15:16	swp13 port is DOWN
spine1	check_port	1h 45m 56s	29/Dec/2020 15:16	swp14 port is DOWN
spine1	check_port	1h 45m 56s	29/Dec/2020 15:16	swp15 port is DOWN
spine1	check_port	1h 45m 56s	29/Dec/2020 15:16	swp16 port is DOWN
spine1	check_port	1h 45m 56s	29/Dec/2020 15:16	swp17 port is DOWN

Accounts

The accounts section is for the management of user accounts, access permissions, and tenants.

19.1 Users

Description of User account fields:

- **Username** - Unique username.
- **Full Name** - Full Name of the user.
- **E-mail** - The email address of the user. Also used for system notifications and for password retrieval.
- **E-mail CC** - Send copies of email notifications to this address.
- **Phone Number** - User's phone number.
- **Company** - Company the user works for. Usually useful for multi-tenant systems where the company provides Netris Controller access to customers.
- **Position** - Position within the company.
- **User Role** - When using a User Role object to define RBAC (role-based access control), Permissions Group and Tenant fields will deactivate.
- **Permission Group** - User permissions for viewing and editing parts of the Netris Controller. (if User Role is not used)
- **+Tenant** - User permissions for viewing and editing services using Switch Port and IP resources assigned to various Tenants. (if User Role is not used)

Example: Creating a user with full access to all sections of Netris Controller, read-only access to resources managed by any Tenant, and full access to resources assigned to the Tenant Admin.

Add new User

Username*	acme
Full Name	Acme
E-mail*	acme@netris.ai
E-mail CC	
Phone Number	
Company	Netris
Position	
User Role *	
Permission Group *	Permit All

+Tenant Add

Read-only access to resources of any Tenant

Full access to resources of Tenant Admin

Permit full access to all sections of Netris Controller

Cancel Add

Password: To set a password or email the user for a password form, go to the listing of usernames and click the menu on the right side.

Example: Listing of user accounts.

Clear Search Export as csv Add

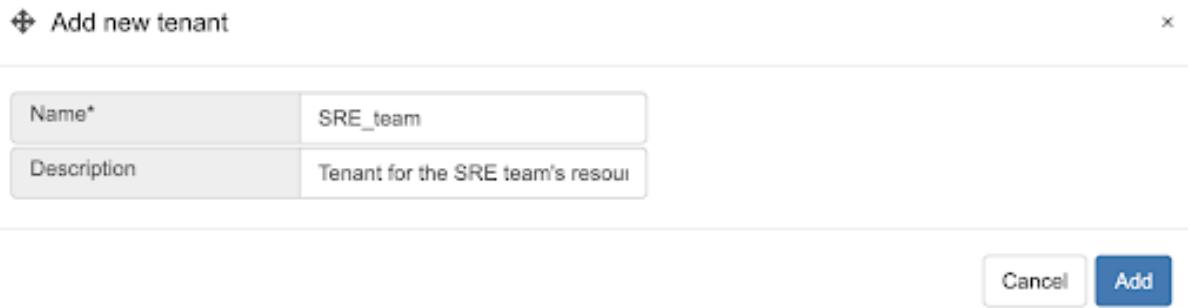
Username	Full Name	User Role	Permission Gr...	Tenants	Details	Created Date
acme	Acme	Permit All	<input type="checkbox"/> Edit All Tenants <input checked="" type="checkbox"/> Edit Admin	<input type="checkbox"/> Edit All Tenants <input checked="" type="checkbox"/> Edit Admin	<input type="checkbox"/> Edit Copy Link Set Password Send Password recovery Delete	2023-09-01
netris		Permit All	<input checked="" type="checkbox"/> Edit All Tenants	<input checked="" type="checkbox"/> Edit All Tenants	<input type="checkbox"/> Edit Copy Link Set Password Send Password recovery Delete	2023-09-01

19.2 Tenants

IP addresses and Switch Ports are network resources that can be assigned to different Tenants to have under their management. Admin is the default tenant, and by default, it owns all the resources. The concept of Tenants can be used for sharing and delegation of control over the network resources, typically used by network teams to grant access to other teams for requesting & managing network services using the Netris Controller as a self service portal or programmatically (with Kubernetes CRDs) as part of DevOps/NetOps pipeline.

A Tenant has just two fields, the unique name and custom description.

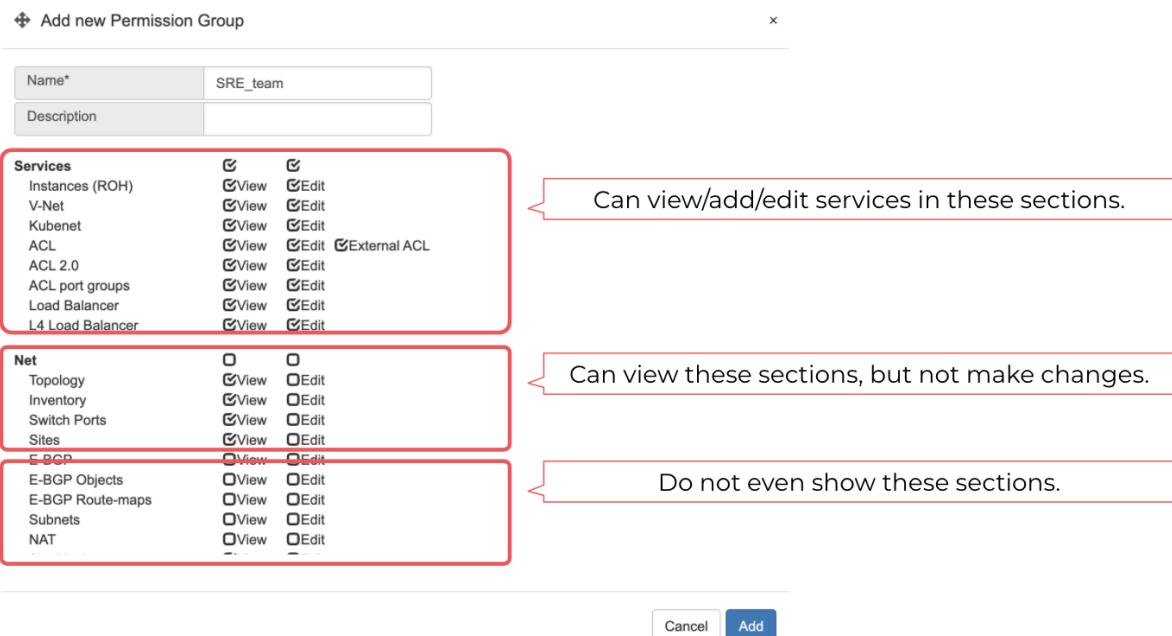
Example: Adding a tenant.

A screenshot of a "Add new tenant" dialog. It has fields for "Name*" (SRE_team) and "Description" (Tenant for the SRE team's resources). Buttons at the bottom are "Cancel" and "Add".

19.3 Permission Groups

Permission Groups are a list of permissions on a per section basis that can be attached individually to a User or a User Role. Every section has a View and Edit attribute. The view defines if users with this Permission Group can see the particular section at all. Edit defines if users with this Permission Group can edit services and policies in specific sections.

Example: Permission Group.

A screenshot of a "Add new Permission Group" dialog. It has fields for "Name*" (SRE_team) and "Description". Below is a table of sections and their permissions:

	View	Edit
Services	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Instances (ROH)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
V-Net	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kubenet	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
ACL	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
ACL 2.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
ACL port groups	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Load Balancer	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
L4 Load Balancer	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Net	<input type="checkbox"/>	<input type="checkbox"/>
Topology	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Inventory	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Switch Ports	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Sites	<input checked="" type="checkbox"/>	<input type="checkbox"/>
E-BGP	<input type="checkbox"/>	<input type="checkbox"/>
E-BGP Objects	<input type="checkbox"/>	<input type="checkbox"/>
E-BGP Route-maps	<input type="checkbox"/>	<input type="checkbox"/>
Subnets	<input type="checkbox"/>	<input type="checkbox"/>
NAT	<input type="checkbox"/>	<input type="checkbox"/>

Annotations explain the permissions:

- Services section: "Can view/add/edit services in these sections."
- Net and E-BGP sections: "Can view these sections, but not make changes."
- E-BGP Objects, E-BGP Route-maps, Subnets, NAT sections: "Do not even show these sections."

Buttons at the bottom are "Cancel" and "Add".

19.4 User Roles

Permission Groups and Tenants can be either linked directly to an individual username or can be linked to a User Role object which then can be linked to an individual username.

❖ Add new User Role x

Name*	SRE_users
Description	
Permission Group*	SRE_team ?

+ Tenant ?

<input type="checkbox"/>	<input checked="" type="checkbox"/> Edit SRE_team Delete	Full access to resources of Tenant SRE_team
<input type="checkbox"/>	<input type="checkbox"/> Edit Admin Delete	Read-only access to resources of Tenant Admin

Cancel Add

Release notes

- DPDK data plane support for SoftGate nodes. - Provides higher SoftGate performance. Up to 27Mpps, 100Gbps for L3 routing, 12Mpps with NAT rules on.
- L4 Load Balancer. - In addition to switch-based Anycast Load Balancer, we now support a Soft-Gate/DPDK-based L4 Load Balancer. L4LB integrates with Kubernetes providing cloud-like load balancer service (type: load-balancer).
- Kubenet - a network service purpose-built for Kubernetes cluster nodes. Kubenet integrates with Kube API to provide an on-demand load balancer and other Kubernetes specific networking features. Netris Kubenet is designed to complement Kubernetes CNI networking with modern physical networking.
- API logs - Comprehensive logging of all API calls sent to Netris Controller with the ability to search by various attributes, sort by each column, and filter by method type.
- SiteMesh - a Netris service for automatically configuring site-to-site interconnect over the public Internet. SiteMesh supports configuration for WireGuard to create encrypted tunnels between participating sites and automatically generates configuration for FRR to run dynamic routing. In a few clicks, services in one site get connectivity to services in other sites over a mesh of WireGuard tunnels.
- Ubuntu/SwitchDev updates - Removed the requirement for a hairpin loop cable. Removed the need for IP address reservation for V-NET, switching entirely to the anycast default gateway.
- Controller distributions - Netris controller is now available in three deployment forms. 1) On-prem KVM virtual machine. 2) Kubernetes application. 3) Managed/Hosted in the cloud.
- Inventory Profiles - A construct for defining access security, timezone, DNS, NTP settings profiles for network switches and SoftGate nodes.
- Switch/SoftGate agents - New installer with easy initial config tool. Support for IP and FQDN as a controller address. Authentication key.
- GUI - Improved Net → Topology section, becoming the main and required place for defining the network topology. All sections got a column organizer, so every user can order and hide/show columns to their comfort.

