

cloudSwXtch

Version 3.1.0



Table of Contents



cloudSwXtch > Getting Started

Getting Started	5
What is cloudSwXtch?	6
What is an xNIC?	7

cloudSwXtch > Getting Started > cloudSwXtch Licensable Features

cloudSwXtch Licensable Features	8
Multicast	10
Source Specific Multicast	11
Broadcast	12
High Availability	13
Bridge	14
Precision Time Protocol	15
Protocol Conversion and Fanout	16
Tachyon LIVE on cloudSwXtch	17

cloudSwXtch > Installing cloudSwXtch

cloudSwXtch System Requirements	18
---------------------------------	----

cloudSwXtch > Installing cloudSwXtch > cloudSwXtch on AWS

cloudSwXtch on AWS	20
Validate Subnets on AWS	21
Verify Security Groups	23
Create SSH Key Pair	25
Install cloudSwXtch on AWS	27
Deploy cloudSwXtch with Terraform on AWS	33
Check Health of cloudSwXtch Instance on AWS	36
Delete cloudSwXtch on AWS	38

cloudSwXtch > Installing cloudSwXtch > cloudSwXtch on Azure

cloudSwXtch on Azure	39
Validate Subnets on Azure	41
Create an Azure cloudSwXtch Template	43
Install cloudSwXtch on Azure	45
Install cloudSwXtch via Azure Marketplace	48
Deploy cloudSwXtch with Terraform on Azure	54
Install cloudSwXtch for an Air-Gapped Environment	57

cloudSwXtch > Installing cloudSwXtch > cloudSwXtch on GCP

cloudSwXtch on GCP	64
Install cloudSwXtch via GCP Marketplace	65

cloudSwXtch > Installing cloudSwXtch > cloudSwXtch on OCI

cloudSwXtch on OCI	70
Install cloudSwXtch via OCI Marketplace	71

cloudSwXtch > Installing cloudSwXtch

Cloud agnostic cloudSwXtch VM Install	77
---------------------------------------	----

cloudSwXtch > Upgrading cloudSwXtch

Upgrading cloudSwXtch	78
-----------------------	----

cloudSwXtch > Installing cloudSwXtch Bridge

Installing cloudSwXtch Bridge	79
cloudSwXtch Bridge System Requirements	80
Install cloudSwXtch Bridge Type 3	82
Install cloudSwXtch Bridge Type 2	89
Install cloudSwXtch Bridge Type 1	91

cloudSwXtch > Installing xNIC

Installing xNIC	92
xNIC System Requirements	93

cloudSwXtch > Installing xNIC > Install xNIC on Linux

Install xNIC on Linux	94
xNIC Linux Uninstall	97
xNIC Linux Upgrade	98

cloudSwXtch > Installing xNIC > Install xNIC on Windows

Install xNIC on Windows	99
Uninstall xNIC on Windows	103

Upgrade xNIC on Windows	104
cloudSwXtch > Installing xNIC > Install xNIC on Kubernetes	
Install xNIC on Kubernetes	105
Install xNIC on K8s Cluster	106
Uninstall xNIC DaemonSet on K8s	112
Test xNIC with K8s	113
Upgrade xNIC nodes on K8s	117
cloudSwXtch > Using wXcked Eye for cloudSwXtch	
Using wXcked Eye for cloudSwXtch	118
cloudSwXtch > Using wXcked Eye for cloudSwXtch > Monitor cloudSwXtch with wXcked Eye	
Monitor cloudSwXtch with wXcked Eye	119
wXcked Eye cloudSwXtch Page	120
wXcked Eye xNICs Page	124
wXcked Eye Topology Graph	127
wXcked Eye Topology Table	139
wXcked Eye Protocol Fanout Stats	142
wXcked Eye Timing Nodes	144
wXcked Eye Support Page	146
cloudSwXtch > Using wXcked Eye for cloudSwXtch > Configure cloudSwXtch with wXcked Eye	
Configure cloudSwXtch with wXcked Eye	148
General	149
High Availability with wXcked Eye	151
Protocol Conversion and Fanout with wXcked Eye	155
Protocol Conversion and Fanout Example	162
Channels	165
Aliases	168
Streams	171
Components	179
cloudSwXtch > Configuring cloudSwXtch > High Availability	
Configuring cloudSwXtch for HA	183
Configuring xNIC on Endpoints for HA	186
Configuring cloudSwXtch Bridge for HA	195
cloudSwXtch > Configuring cloudSwXtch	
Bridge Type 2 and Type 3	197
Bridge Type 1	200
Protocol Conversion and Fanout	201
cloudSwXtch > Monitoring cloudSwXtch	
Azure Monitoring	202
Prometheus Monitoring	204
cloudSwXtch > Testing cloudSwXtch	
Testing cloudSwXtch	213
swxtch-perf	214
swxtch-top	217
swxtch-tcpdump	227
swxtch-where	228
Universal Third-Party Testing Tools	231
cloudSwXtch > cloudSwXtch How To's	
How to View cloudSwXtch Logs for Troubleshooting	238
How to Install xNIC Dependencies in an Air-Gapped Environment	241
How to View cloudSwXtch Bridge Logs	250
How to Find xNIC Logs	252
How to License a cloudSwXtch	256
How to set MTU size	257
How to Peer between VPCs in Different Regions for AWS	261
How to Modify CPU Core Usage in Icore for Large Instances	265
cloudSwXtch > Market Use Cases > Media Use Cases	
Media Use Cases	266
Media Multicast made easy with cloudswXtch	267
Hitless Merge - 2022-7	268
Media support for Compressed and Uncompressed Workflows	269
Protocol Fanout	271
Disaster Recovery	272
cloudSwXtch > cloudSwXtch API	
Monitoring API	273

Configuration API	301
cloudSwXtch	
Release Notes	337
cloudSwXtch > Resources	
Resources	344
cloudSwXtch	
Quotas	345

Getting Started

Quick Start Guide (for those in a hurry)

Introduction

swXtch.io implements a software-based network switch called cloudSwXtch. cloudSwXtch consists of a software network switch and virtual NIC service called xNIC . Together, these components create an overlay network on top of a standard cloud network. This overlay network adds many valuable network features, one of which is a seamless IP multicast experience. With cloudSwXtch, existing user applications and services, that expect standards-based IP multicast, will work on any cloud without requiring any code changes. This enables performance to approach that of bare metal.

Installing cloudSwXtch and xNIC

WHAT TO EXPECT

- In this section, users will be able to learn more about installing cloudSwXtch and the xNIC on AWS, Azure, GCP and OCI.
- Users **must** install an xNIC on every VM that needs to send or receive cloudSwXtch multicast or broadcast traffic.

Before you install cloudSwXtch, please review the [System Requirements](#).

[AWS cloudSwXtch Installation Guide](#)

[Azure cloudSwXtch Installation Guide](#)

[GCP cloudSwXtch Installation Guide](#)

[OCI cloudSwXtch Installation Guide](#)

Before you install the xNIC, please review [xNIC System Requirements](#).

[xNIC Installation Guide for Windows and Linux](#)

[xNIC Installation Guide for Kubernetes](#)

wXcked Eye for Monitoring and Orchestration

wXcked Eye is a web-based [monitoring](#) and [orchestration](#) tool for cloudSwXtch. It presents users with a high-level view of their cloudSwXtch environment with an interactive network graph detailing connections to different endpoints. With an expansive look at performance metrics, users can ensure that their data is flowing as expected.

In addition, wXcked Eye unlocks the ability to configure Mesh, High Availability, Protocol Fanout, and Precision Time Protocol (PTP) from the comfort of a user's web browser.

For more information, see [Using wXcked Eye for cloudSwXtch](#).

Testing

The xNIC installation includes the following utilities that can be used to verify both the functionality and performance of the network:

- **swxtch-top**: This utility shows detailed switch statistics in the console. For more information, click [here](#).
- **swxtch-perf**: This utility can be used to produce and consume multicast traffic for testing. For more information, click [here](#).
- **swxtch-topdump**: This utility helps capture multicast packets sent to and from the cloudSwXtch with logic to decode our own head and display the original MC payload.
- **swxtch-where**: This utility allows users to call for hardware information regarding their cloudSwXtch VM.

Each of the utilities can be run from a VM that has the xNIC software installed. Detailed usage information can be found for each by entering in the **—help** command-line argument.

Multicast Examples

Users can find examples of the multicast by scrolling down to the Multicast Example section in the [swxtch-perf article](#).

What is cloudSwXtch?

WHAT TO EXPECT

In this article, users will get a deeper understanding of **cloudSwXtch** and how it can improve their networking capabilities. This article also gives users a preliminary introduction to the main features available while using cloudSwXtch.

Meet cloudSwXtch

cloudSwXtch creates a virtual overlay network that lets users add high performance networking to their cloud or edge applications with the touch of a button, requiring no code changes!

cloudSwXtch is available on AWS, Azure, GCP and OCI. It can be instantiated via their respective Marketplaces. cloudSwXtch is also available as a BYOL software install.

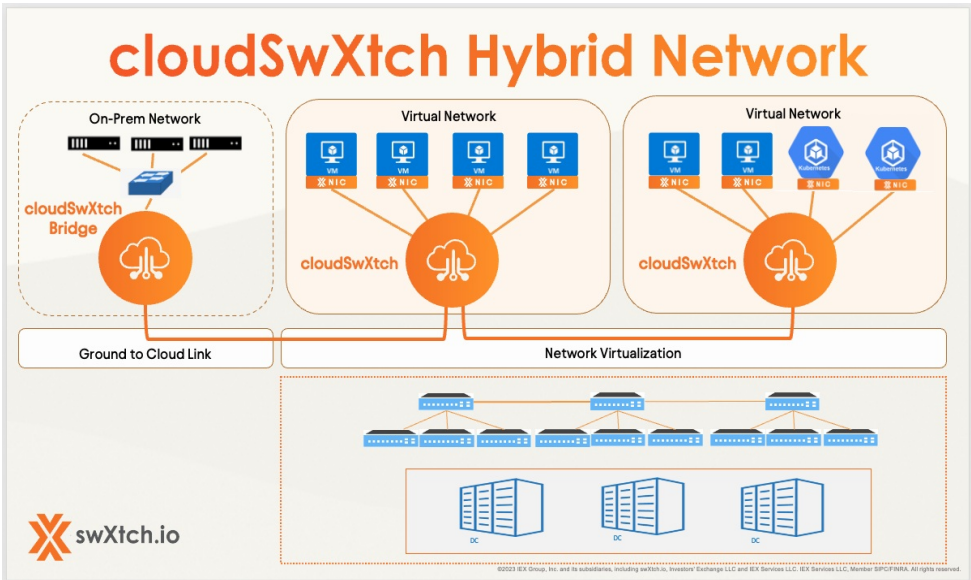
Supported Environments

- Microsoft's [Azure](#)
- Amazon's [AWS](#)
- Google's [GCP](#)
- Oracle's [OCI](#)
- On-Premises with [cloudSwXtch Bridge](#)

What is a Virtual Overlay Network?

swXtch.io provides an application that implements a **Cloud Based Virtual Switch - cloudSwXtch**. It consists of a software-based network switch and a virtual NIC service (xNIC). Together, these components create an overlay network on top of the standard cloud network.

This overlay network adds many valuable, high-performance network features that aren't traditionally available in the cloud; one of which is a **seamless IP multicast** experience.



cloudSwXtch Instance

A cloudSwXtch instance running on a user's virtual machines will provide extremely low latency ($< 3\mu s$), high determinism, and elastic scalability. A user can build a 1,000-port switch or create a cloudSwXtch mesh of switches to optimize network reliability.

With cloudSwXtch, existing user applications and services that expect standards-based IP **multicast** will work in the cloud **without requiring any code changes**. This enables performance to approach that of bare metal.

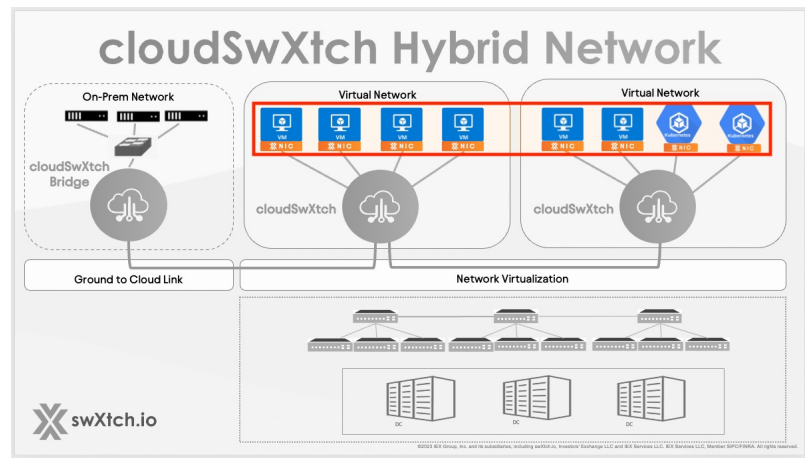
Benefits of cloudSwXtch

- **Unblock Cloud Migrations** – Migrate critical workloads that couldn't move to the cloud because of missing network features or performance limitations.
- **Extend On-Prem Networks to the Cloud** – Create a single data plane across private networks and the cloud, traversing virtual networks, availability zones, and regions.
- **Massive Scale** – Extended networks with unlimited endpoints share identical features and sub-millisecond performance.
- **Enhanced Packet Monitoring** – The cloudSwXtch architecture provides a unique view into low level network traffic across the entire extended network.
- **Simplified and Flexible Network Configuration** – Add and remove endpoints dynamically from global networks as conditions dictate. Eliminate the need to reconfigure individual workloads.

What is an xNIC?

WHAT TO EXPECT

In this article, you will learn how the xNIC, our virtual NIC, fits into the cloudSwXtch infrastructure.



A Fully Virtualized NIC

The **xNIC** is a software service deployed by swXtch.io that creates a fully virtualized **Network Interface Controller (NIC)** within a virtual machine's (VM) operating system. In tandem with the cloudSwXtch, the xNIC provides users with wicked fast network connectivity and enables applications running on the VM to send and receive IP multicast traffic. It is seamless and transparent to all workloads, requiring no code changes.

xNIC installation for cloudSwXtch is required for multicast to work across all major clouds — **AWS**, **Azure**, **GCP**, **OCI**. It is available on both **Linux** and **Windows** machines as well as **Kubernetes**. You can find a more detailed list of **xNIC system requirements**, [here](#).

In addition to multicast, the xNIC can be configured to perform high availability. At an endpoint, the xNIC compares packet reception from the multiple paths, detects dropped packets, and reconstructs the output stream in the correct order. For more information, see [Configuring xNIC for High Availability](#).

Q: Can xNIC work on any IP within the multicast address space?

A: We recommend referring to the IANA [IPv4 Multicast Address Space Registry](#). Currently, the xNIC can't join a multicast stream if it's on 224.0.0.1.

cloudSwXtch Licensable Features

WHAT TO EXPECT

The following section gives a preliminary introduction to the main features available while using a cloudSwXtch.

Licensing cloudSwXtch Features

In addition to [Multicast](#) and [wXcked Eye](#), users can license the following features for their cloudSwXtch:

Protocol Fanout	Protocol Conversion (e.g. SRT to Multicast)	Ground to Cloud/Cloud to Ground	SMPTE ST 2022-7 & High Availability
Precision Time Protocol (PTP)	Tachyon LIVE!	Increase Bandwidth Capacity (Ingress)	Additional Endpoint Connections

Multicast

cloudSwXtch enables true and seamless IP-multicast. Using **multicast**, instead of unicast, optimizes a user's network configuration, reducing their cloud distribution and egress costs. In addition, receivers can dynamically subscribe and unsubscribe to a user's streams as workflows dictate. cloudSwXtch alleviates the need to have to constantly reconfigure unicast streams to accommodate downstream receivers. cloudSwXtch uses the industry standard IGMPv2/v3 for its management of multicast group membership.

For more information, check out the [Multicast](#) feature article.

Single Source Multicast (SSM)

Traditionally, Single Source Multicast (SSM) is a method for delivering multicast packets in which the only packets that are delivered to the receiver are those originating from a specific source address requested by the receiver. This can be accomplished as either a consumer command for swtch-perf, the cloudSwXtch-based tool for simulating traffic movement, or via an external application.

Protocol Conversion and Fanout

cloudSwXtch supports a unique feature called **protocol conversion and fanout**. This feature is useful when a user's multicast application needs to stream to an endpoint that does not support multicast or it is not possible to install an xNIC in the endpoint. cloudSwXtch can map a multicast group address to a unicast address. Similarly, a unicast input to cloudSwXtch can be mapped to a multicast group enabling multiple endpoints to consume the original unicast input stream. Protocol Fanout converts many packet protocols and distributes them out as if they were multicast; freely integrating multicast, unicast and Secure Reliable Transport (SRT) streams while making the network more efficient and reducing egress costs.

For more information, check out the [Protocol Conversion and Fanout](#) feature article.

SMPTE 2022-7 and High Availability (HA)

High Availability (HA) protects users against data path errors by sending the same stream through as many as eight different distributed data paths. It compares packet reception from the multiple paths, detects dropped packets and reconstructs the output stream in the correct order. This feature is compliant with SMPTE 2022-7 for media workflows.

For more information, check out the [High Availability](#) feature article.

Ground to Cloud <==> Cloud to Ground

A user can connect their On-Prem network to their cloudSwXtches in the Cloud via the [bridge application](#).

For more information, check out the [Installing cloudSwXtch Bridge](#) guide.

wXcked Eye for Monitoring and Configuration

cloudSwXtch also provides its users with visibility down to the packet level for enhanced Monitoring and Quality of Service (QoS). **wXcked Eye** is the cloudSwXtch monitoring UI tool that enables users to deeply audit the performance of their cloudSwXtch network. Each cloudSwXtch performs complete packet capture.

In addition, wXcked Eye also provides users with an additional avenue to configure their cloudSwXtch environment for mesh, high availability, protocol conversion and fanout, and precision time protocol.

A REST API is provided to help users manage and control their network in their own way.

For more information, please see the [Using wXcked Eye for cloudSwXtch](#) article.

Precision Time Protocol (PTP)

Precision Time Protocol (PTP) is a cloudSwXtch feature that facilitates clock synchronization between agents connected to the network. The cloudSwXtch acts as the **Master Node**, passing on the information gained from the true clock source to the **Follower Nodes** or agent end points.

For more information, please see the [Precision Time Protocol \(PTP\)](#) feature article.

Tachyon LIVE!

Tachyon LIVE! is a live, high-quality standards, format, and frame rate converter software stack that maximizes video quality across all conversions. It performs standards conversion including PAL/NTSC frame rate and format conversions, high-quality deinterlacing, and up/down rescaling from SD through HD, and it can process the highest-quality conversions for HD 59.97p to HD 50p, faster than real time in a VM with NVIDIA GPU-accelerated infrastructure.

This feature is an exclusive offer for cloudSwXtch from Cinnafilm and available as either an HD or UHD add-on.

For more information, please see the [Tachyon LIVE](#) feature article.

Multicast

Multicast

Software defined multicast (SDMC™) is a feature of the **cloudSwXtch** overlay network. With SDMC, existing applications and services that expect standards-based, IP multicast will work **without requiring any code changes** and with performance that approaches that of bare metal.

At a high level, **cloudSwXtch** implements a **software switch** that serves the same role as a hardware switch. **cloudSwXtch** receives multicast packets from producers and sends a copy of each packet to every destination VM. The **cloudSwXtch** control plane uses the industry standard IGMPv2/3 specification for the management of group membership.

The **xNIC** service handles multicast traffic between the switch and the VM operating system. The **xNIC** service must be installed on every VM that needs to send or receive multicast traffic.

SUMMARY

The **cloudSwXtch** system consists of a software switch instantiated within a virtual network and a set of virtual machines that have an **xNIC** virtual interface.

Applications can send and receive IP multicast by targeting the virtual network interface. IGMP control packets are generated by the local operating system and the **xNIC** virtual interface seamlessly picks these up and sends them to the **cloudSwXtch** instance. Local applications will work in this environment just as they would on a similar bare-metal network.

Source Specific Multicast

WHAT TO EXPECT

In this section, we will go into detail about Source Specific Multicast, or SSM, and how it improves security when sending/receiving multicast packets.

Source Specific Multicast, or SSM, as defined by IGMPv3, is a method for delivering multicast packets in which the only packets that are delivered to the receiver are those originating from a specific source address requested by the receiver. Not only does this improve security within the cloudSwitch but it also alleviates strain on the network since the sender will know to only send a multicast stream from the specified single source and not via other source addresses.

This feature can be tested using swtch-perf, a cloudSwitch-based tool for simulating consumer and producer traffic, as well as external applications that support SSM. For more information on using it for SSM, please see the [swtch-perf](#) article under Testing cloudSwitch.

Broadcast

Broadcast is a feature of the cloudSwXtch overlay network. With Broadcast, existing applications and services that expect standards-based broadcast will work without requiring any code changes and with performance that approaches that of bare metal.

At a high level, cloudSwXtch implements a software switch that serves the same role as a hardware switch. cloudSwXtch receives broadcast packets from producers and sends a copy of each packet to every destination VM.

The xNIC 2 service handles tunneling broadcast traffic between the cloudSwXtch and the VM operating system. The xNIC 2 service must be installed on every VM that needs to send or receive broadcast traffic.

SUMMARY

The cloudSwXtch system consists of a software switch instantiated within a virtual network and a set of virtual machines that have an xNIC 2 virtual interface.

Applications can send and receive broadcast by targeting the virtual network interface. Broadcast packets are generated by the local operating system and the xNIC 2 virtual interface seamlessly picks these up and sends them to the cloudSwXtch instance. Local applications will work unchanged in this environment just as they would on a similar bare-metal network.

High Availability

WHAT TO EXPECT

High Availability (HA) is an implementation of data path redundancy and stream duplication. It protects users from data loss by replicating and sending packets through multiple network paths. xNIC compares packets received from those multiple paths and automatically reconstructs the original stream.

In this section, users will learn more about the benefits of implementing the High Availability feature in their cloudSwXtch and understand how to leverage it for their future needs.

Creating A More Resilient Network

With High Availability, critical workloads can be configured to be more resilient, stretching across regions or availability zones in a single cloud. In addition, it can be used across multiple cloud providers. Although there can only be up to eight redundant paths, there are no limits to the number of consumers that can receive the HA stream, other than bandwidth constraints.

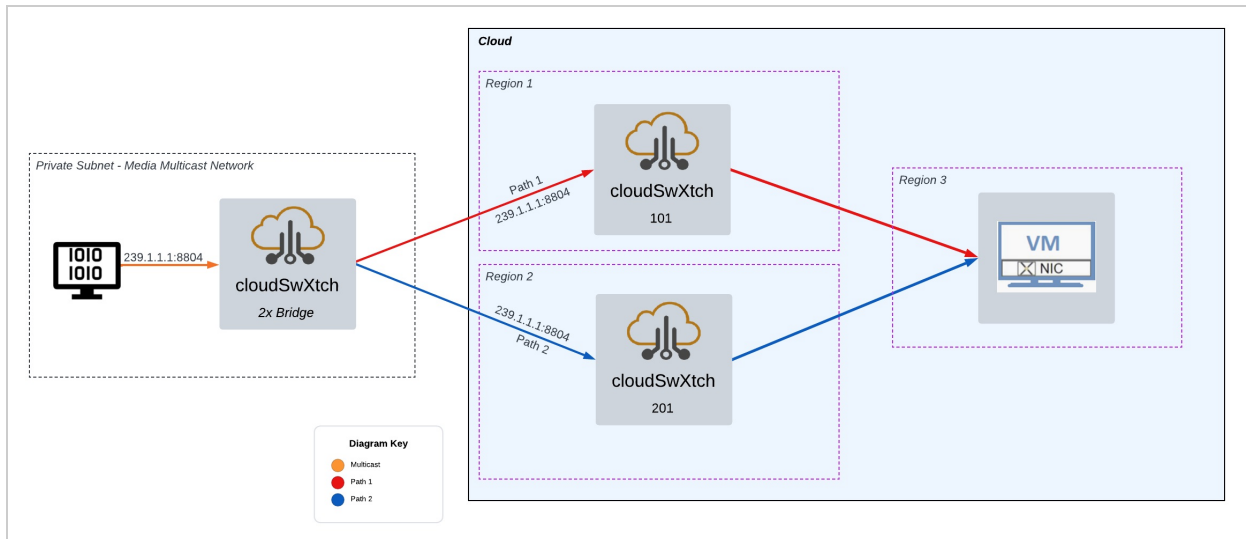
In addition, there is no limit to the number of multicast groups per data path. If one cloud, availability zone or region should go down, then the data is still sent in the other 2-8 paths, ensuring that the consumer gets the necessary data. Consumers can also be put into different clouds, availability zones or regions so that if a consumer becomes unavailable, users can still sign into a different cloud, availability zone or region and get the data desired.

The HA feature forwards packets to the receiving application from any of the configured paths as soon as the "next" expected packet is received. Redundant packets from other paths are discarded. There is no additional latency imposed by the HA feature.

IMPORTANT

A cloudSwXtch configured in a HA path cannot be used in a cloudSwXtch mesh. They are mutually exclusive.

High Availability Example



The simple diagram above shows high availability with one multicast group 239.1.1.1:8804 originating from an on prem source. From the bridge, two paths are created with redundant packets being sent to alternate cloudSwXtches in different regions. The number of regions and cloud providers needed for High Availability will vary depending upon the customer's environment.

Independent path redundancy ensures no packet loss if every packet arrives at the consumer from at least one path. For example:

- In the event that cloudSwXtch101 goes offline, the consumer will still get the multicast traffic via cloudSwXtch201 (or vice versa).
- In the event that there are network issues in Region 1 where some of the packets are lost in path one, the consumer can still get the multicast traffic with High Availability pulling data from Region 2 in path two.
- In the event that there are network issues in Region 1 and 2 where some of the packets are lost in both paths, both consumers can still get the multicast since the high availability function will take the valid packets and reconstruct the multicast stream from Region 1 and 2.

In each example, despite losing paths, multicast data was still able to get to the end point using high availability with no packet loss. Configuring more paths will ensure higher availability of the multicast group.

HA can be monitored via swtch-top, see [swtch-top](#) section 4-6.

To configure the system for high availability, refer to: [High Availability Configuration](#).

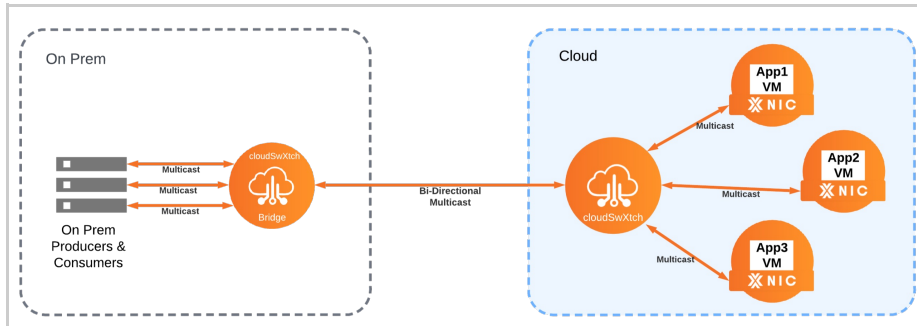
Installing cloudSwXtch - Firewall Exceptions

When installing the cloudSwXtch, high availability requires special firewall exceptions. To learn more, see [cloudSwXtch System Requirements](#).

Bridge

cloudSwXtch Bridge

The **cloudSwXtch Bridge** application enables bi-directional multicast traffic between a non-cloud and cloud network. The source network can be bare-metal and on-premises. The destination network can be a cloud virtual network with a **cloudSwXtch** instance deployed. With **cloudSwXtch**, multicast traffic generated from the on-prem network can be received and processed in the cloud which then in turn can be sent back to the on-prem network.



The cloudSwXtch Bridge is bi-directional. It sends multicast traffic from the on-premises network to the cloud and from the cloud to on-premises.

From on-prem to the cloud, the bridge is dynamic. This means that users in the cloud can subscribe to a multicast group via IGMP joins. Then, the bridge will allow that traffic through. This ensures that only necessary traffic goes through the VPN or Express Route/Gateway into the cloud. It guarantees the best use of the gateway and incurs less ingress bandwidth into the cloud.

Operation

The operation of the cloudSwXtch Bridge varies based on direction.

Ground-->Cloud

For Ground to Cloud, a mesh must be configured between the cloudSwXtch and the cloudSwXtch Bridge at the ground. From then on, the operation is dynamic, meaning the user does not need to map multicast addresses to go into the cloud. Instead, when a user is in an application and use an IGMP join then a message is sent to the cloudSwXtch Bridge via the cloudSwXtch through the mesh and then the Bridge allows that traffic through. When the user stops using multicast group and does an IGMP leave, then the bridge stops sending multicast data.

Cloud-->Ground

For Cloud to ground there is no current support to propagate IGMP joins and leaves from cloudSwXtch to on-prem. In this case, multicast groups must be explicitly configured to let the bridge know what traffic is allowed.

See [Bridge Installation](#) on how to install the Bridge and the differences between Bridge Type 1 and Type2. See our configuration pages for [Bridge Type 1](#) and [Bridge Type 2](#).

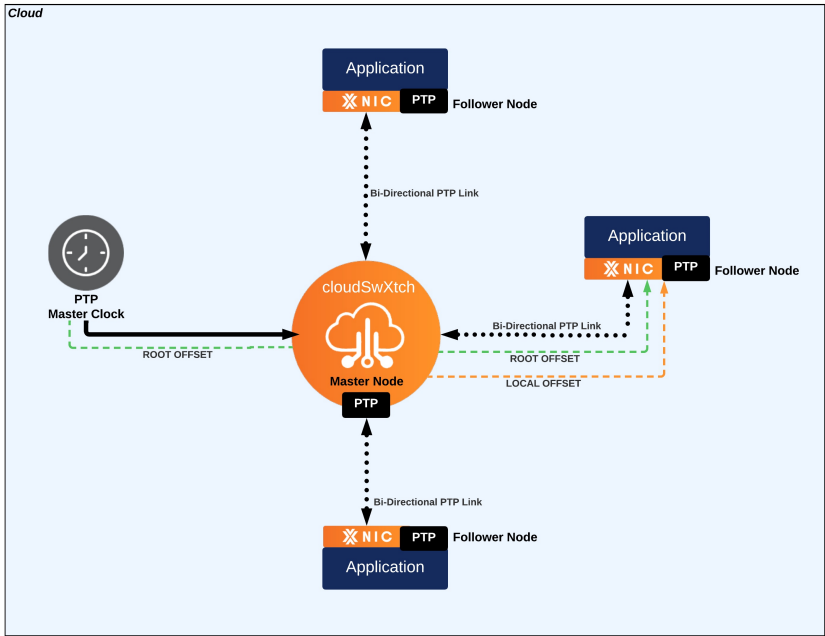
Precision Time Protocol

WHAT TO EXPECT

In this article, users will learn how Precision Time Protocol (PTP) works in a cloudSwXtch environment when the feature is activated.

What is Precision Time Protocol?

Precision Time Protocol (PTP) is a cloudSwXtch feature that facilitates clock synchronization between agents connected to the network. The cloudSwXtch acts as the **Master Node**, passing on the information gained from the true clock source to the **Follower Nodes** or agent end points.



Information regarding PTP will display in both **swXtch-top** under the PTP page and **wXcked Eye** under Timing Nodes. Both cloudSwXtch tools will show the local and root offset. The **local offset** denotes the offset in time from the cloudSwXtch to the xNIC. The **root offset** denotes the offset in time from the True Clock Source and the cloudSwXtch's follower nodes (xNICs). The root value will always be larger than the local since the distance between the follower node and the True Clock Source is greater than the offset between a cloudSwXtch and xNIC.

Protocol Conversion and Fanout

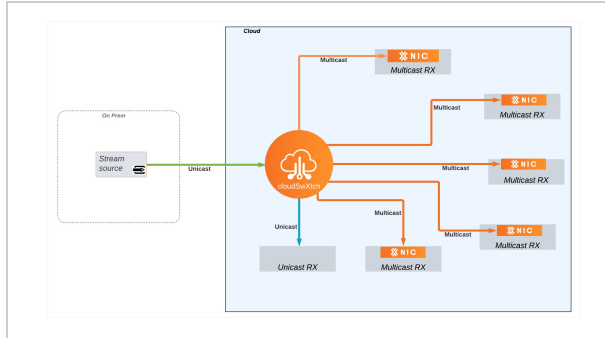
WHAT TO EXPECT

Protocol Conversion and Fanout allows users to send copies of a single input stream in any supported protocol to multiple destinations. This gives each destination the option of being a different protocol from the input stream. An example would be a UDP input being sent to a set of multicast destination and, additionally, to one using SRT.

In this section, users will become more familiar with how Protocol Conversion and Fanout works in a cloudSwXtch-enabled environment.

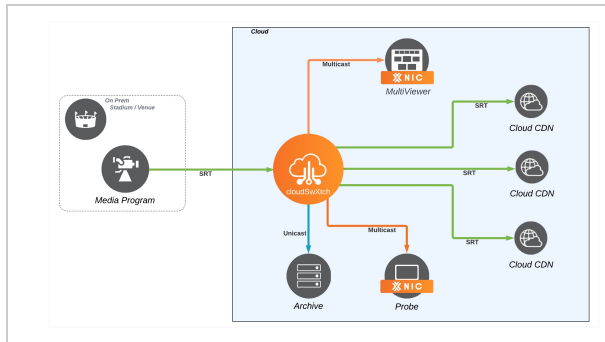
What is Protocol Conversion and Fanout?

It is not usual for workflows to have many endpoints with each having their own protocols: SRT, Unicast and Multicast. Configuring each device can be a difficult and time consuming endeavor and up until now, impossible in the cloud. However, with cloudSwXtch, that is no longer the case. Users can convert protocols and send out multiple copies of payloads to different receivers regardless of protocol type without the need to add custom software or hardware.



This is a generic depiction of a Protocol Conversion and Fanout configuration.

In the above example, unicast data flows from the stream source into the cloudSwXtch. With Protocol Conversion and Fanout enabled, the cloudSwXtch can convert the unicast stream into multicast and fan it out to multiple receivers. In addition, the cloudSwXtch is sending out the stream to a unicast receiver. **Please note:** While it is not depicted in the above example, the cloudSwXtch can send the stream out to multiple unicast receivers.



This is a media depiction of a Protocol Fanout configuration.

In the alternative example, SRT data flows from the stream source into the cloudSwXtch. With Protocol Fanout enabled, the cloudSwXtch can convert the SRT stream into multicast and fan it out to multiple receivers. In addition, the cloudSwXtch is sending out the stream to a unicast receiver and to multiple SRT receivers. **Please note:** While it is not depicted in the above example, the cloudSwXtch can send the stream out to multiple unicast receivers.

Understanding Endpoints

Workflows often have many endpoints: some requiring unicast, and some requiring multicast. Configuring for each device can be difficult and supporting both unicast and multicast for the same stream requires custom software or hardware. cloudSwXtch has the ability to map multicast streams to unicast streams and unicast streams to multicast streams allowing non-xNIC endpoints to participate in the cloudSwXtch network. This feature actualizes two different scenarios:

1. **Non-xNIC producers**, such as, those external to the cloud can send traffic to the cloudSwXtch, via unicast or SRT. The cloudSwXtch, then, can map that unicast or SRT stream to a multicast group for consumption within the cloudSwXtch network.
2. **Non-xNIC consumers** can receive traffic from a cloudSwXtch, as multicast streams can be mapped to unicast or SRT endpoints. This implies that non-xNIC consumers can receive packets created from a xNIC producer.

xNIC consumers and producers can consume SRT, unicast, and/or multicast based on consumer/producer workflow. For example, a VM may have 3 applications installed with each requiring a different protocol. The cloudSwXtch can send all three in the event that all three are needed.

Configuring Protocol Conversion and Fanout for cloudSwXtch

Users can configure Protocol Conversion and Fanout using two methods:

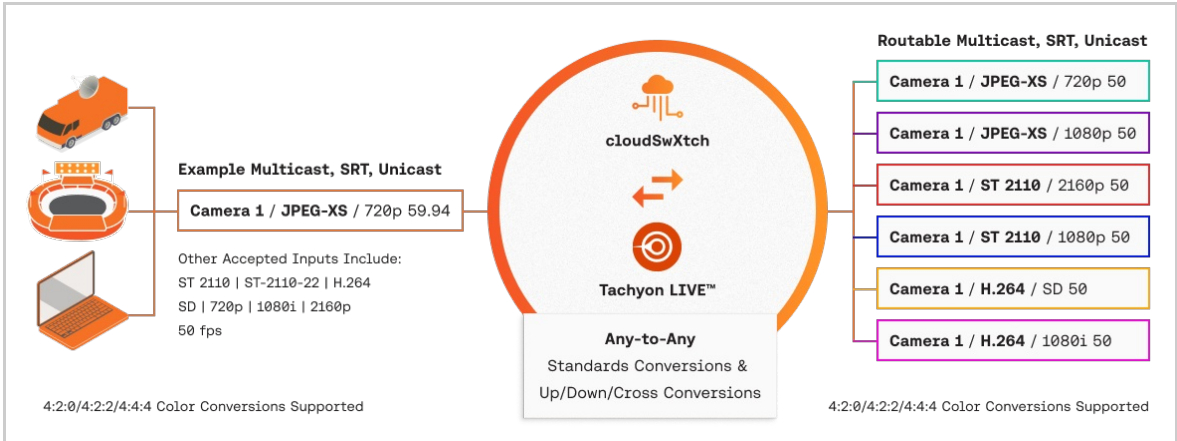
- [Via wXcked Eye](#)
- [Via API](#) - Please see the section on Protocol Fanout.

Tachyon LIVE on cloudSwXtch

WHAT TO EXPECT

In this article, users will learn about the various video transformations available with Tachyon LIVE on cloudSwXtch.

What is Tachyon LIVE?



Tachyon LIVE is a live, high-quality standards, format, and frame rate converter software stack that maximizes video quality across all conversions. It performs standards conversion including PAL/NTSC frame rate and format conversions, high-quality deinterlacing, and up/down rescaling from SD through HD. It can process the highest-quality conversions for HD 59.97p to HD 50p, faster than real time in a VM with NVIDIA GPU-accelerated infrastructure.

With cloudSwXtch, this combined solution can take in LIVE multicast, RTP, UDP, NDI, SRT, and SMPTE 2110 flows for conversion and output them to multiple destinations in any of the supported network protocols. This enables simple integration into existing broadcast workflows both on public cloud or on-premises.

Supported Media Formats and Stream Capacity

Protocol	Supported Input Codecs & Formats	Supported Output Video Codecs & Formats	Supported Output Audio Codecs & Formats
SRT	H.264 & HEVC 4:2:0/4:2:2/4:4:4 SD to UHD resolutions to 60 FPS Uncompressed & AAC Audio	H.264 SD & HD to 60fps, 4:2:0/4:2:2/4:4:4 H.264 UHD 4:2:0 to 60fps HEVC SD to UHD 4:2:0 to 60fps	AAC and Uncompressed Audio (Stereo or 5.1 – TBC)
NDI	All supported video formats SD to UHD resolutions to 60 FPS All Supported Audio Formats	All supported video formats SD to UHD resolutions to 60 FPS	All Supported Audio Formats
2110	Uncompressed or JpegXS Video SD to UHD resolutions to 60 FPS Uncompressed Audio	Uncompressed or JpegXS Video SD to UHD resolutions to 60 FPS	Uncompressed Audio

Installing Tachyon LIVE for cloudSwXtch.

To learn more about how to install Tachyon LIVE for your cloudSwXtch, please contact support@swxtch.io for more information.

cloudSwXtch System Requirements

WHAT TO EXPECT

In this article, users will learn about the system requirements needed to successfully deploy a cloudSwXtch. It is recommended for a user to review this page before installing a cloudSwXtch any of the four cloud platforms.

cloudSwXtch Sizing Guidelines

Sizing and Feature Selection For Your cloudSwXtch

The number of endpoints and bandwidth dictate cloudSwXtch sizing requirements. It is recommended for users to **contact a swXtch.io sales representative** to discuss cloudSwXtch sizing and additional features so that the appropriate license can be distributed. **Please note:** A cloudSwXtch BYOL offering will not work without a license.

- **Sizing:** For bandwidth greater than 2 Gb/s and endpoints greater than 100, you will need different virtual CPUs/NIC sizing.
- **Adding Features:** Many additional licensable features are available for cloudSwXtch. For more information, see [cloudSwXtch Features](#).

To contact sales, please visit [swXtch.io/contact](#).

cloudSwXtch BYOL (Marketplace)

# Endpoints	Bandwidth	Core	Memory	Hard Drive
Up to 100	2 Gb/s (max)	16+	16GB DDR	64GB SSD
Up to 200	More than 2Gb/s	64+	16GB DDR	64GB SSD

cloudSwXtch PAYG for POCs/Trial

# Endpoints	Bandwidth	Core	Memory	Hard Drive
Unlimited	Unlimited	12+	48GB DDR	*

*Hard Drive is dependent on the user's needs. **Note:** cloudSwXtch Pay As You Go (PAYG) is a \$30/hr offering.

Internet Connection

Installing and upgrading cloudSwXtch **requires** internet connection. Alternatively, if a user **does not have access** to the internet, they can use the [Air-Gapped installation guide for Azure](#).

Supported Cloud Environments

- Amazon's [AWS](#) Cloud
- Microsoft's [Azure](#) Cloud
- Google's [GCP](#) Cloud
- Oracle's [OCI](#) Cloud

Virtual Network

A cloudSwXtch instance **must have 2 NICs**. However, both NICs can share a single subnet for control and data plane communications. This is the preferred method.

In the event that a user needs higher performance, a user can separate their subnets as described below.

- Contain a subnet for control plane traffic (referred to as the **ctrl-subnet** from here on).
- Contain a subnet for data plane traffic (referred to as the **data-subnet** from here on).

Please note: GCP does not allow for single subnet configuration. A user must have 2 separate subnets for their data and control NICs.

Subnet Selection

The subnets must be the same subnets used for the xNIC installations.

The virtual network and subnets may be shared with other services in addition to the **cloudSwXtch**. The size of each subnet should include at least 32 addresses.

Minimum CPU and Memory

A cloudSwXtch must be a minimum of 8 cores and 16 GiB memory.

Firewall and Security Group Rules

The xNIC software and the cloudSwXtch communicate with each other using the following protocols and ports. These firewall exceptions must be allowed in the xNIC VMs and the cloudSwXtch VM.

Subnet	Protocol	Ports	VM
ctrl-subnet	http	80	cloudSwXtch
ctrl-subnet	udp	10800-10803	all
data-subnet	udp	9999	all

Mesh and High Availability

Both Mesh and High Availability need special firewall exceptions in order to properly work in a user's cloudSwXtch environment. If you plan on using either features, please allow the following:

Mesh

Subnet	Protocol	Ports	VM
ctrl-subnet	tcp+udp	37856	cloudSwXtch

High Availability

Subnet	Protocol	Ports	VM
ctrl-subnet	tcp+udp	37856	cloudSwXtch

Reminder: HA Mesh are mutually exclusive and cannot be used together.

PTP

PTP needs special firewall exceptions in order to properly work in a user's cloudSwXtch environment. If you plan on using the feature, please allow the following:

Subnet	Protocol	Ports	VM
ctrl-subnet	http	80	cloudSwXtch
ctrl-subnet	udp	10800-10803	all
data-subnet	udp	9999	all

cloudSwXtch on AWS

Pre-Creation Steps

Before creating an EC2 instance with cloudSwXtch installed for AWS, users must already have an AWS account ***and*** a VPC (Virtual Private Cloud) already created.

Installation Method:

1. [Review system requirements](#)
2. [Validate subnets on AWS](#)
3. [Verify security groups](#) *Optional*
4. [Create SSH key pair](#)
5. [Install cloudSwXtch on AWS](#)

Disclaimers

- swtch.io does not handle any policy access rights for deployment nor does it have any special IAM roles or policies that are needed. That being said, swtch.io suggests using a policy of least privilege for all access granted as part of the deployment. Please refer to AWS for best practices for policy rights and IAM roles and policies: [AWS Identity](#)
- swtch.io does not require any public resources for deployment such as Amazon S3 buckets.
- swtch.io cloudSwXtch installation does not use any AWS Secrets in Secret Manager as swtch.io does not natively store any customer sensitive data. Customers can encrypt their traffic and the cloudSwXtch will still be able to handle the network traffic.
- swtch.io does not encrypt data. It pass through any data sent in the multicast which may be encrypted.

Validate Subnets on AWS

WHAT TO EXPECT

A virtual network must be created before deploying a cloudSwXtch EC2 instance.

- It must contain at least one subnet that's used for both the control and data plane communication.
 - It is recommended that it is **private facing** and **does not auto-assign public IPs**.
 - This single subnet will be used for xNIC installation.

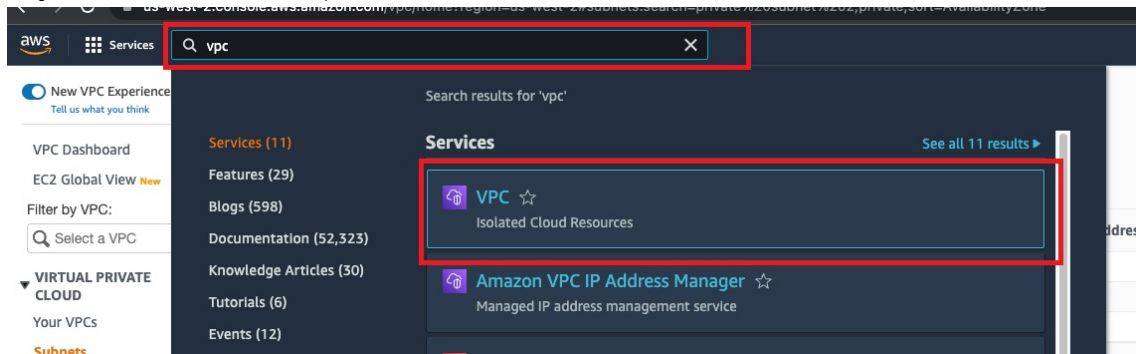
In this section, users will learn how to validate whether a subnet exists to be used as both the control and the data plane for their virtual network. This is in preparation for cloudSwXtch installation on AWS. We will also walk through an alternative method of using 2 subnets, separating the control and data plane.

Method #1: Single-subnet

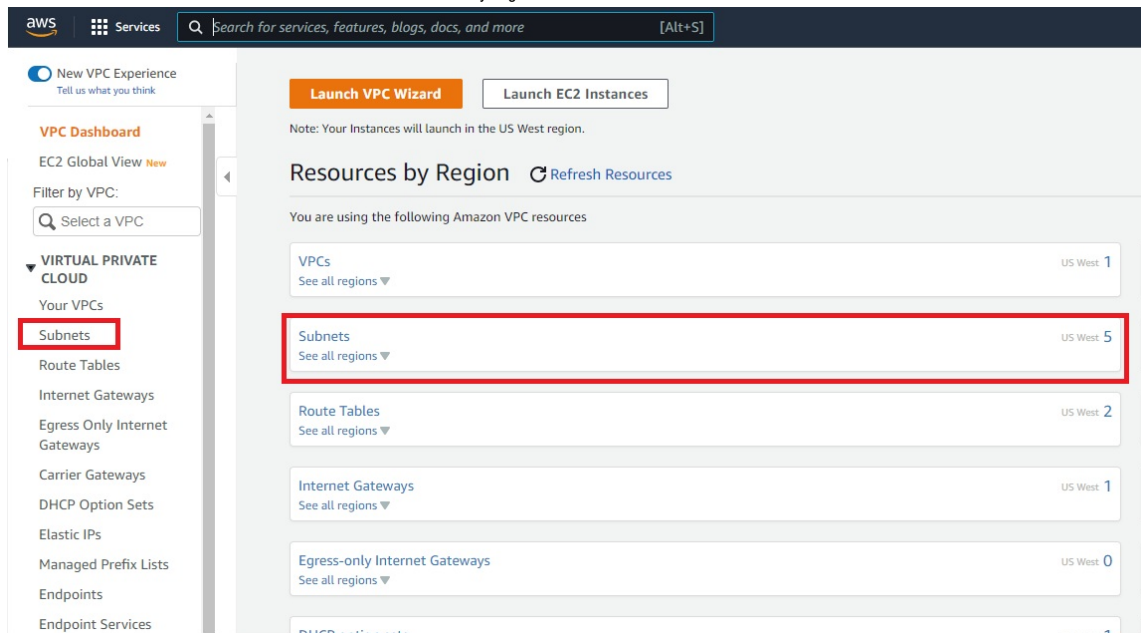
Typically, when deploying a VPC, a user will automatically create a subnet. During the main installation process, this subnet can be used for both control and data plane communications. This is the **preferred** method and will be used by a majority of users. Before installing cloudSwXtch, users should validate that the control subnet exists.

To validate:

1. **Navigate** to the VPC Console in AWS. In the example below, the user entered VPC in search field to find it under Services.



2. **Select "Subnets"** under the Virtual Private Cloud tab or under Resources by Region in the VPC Dashboard.



3. **Check that the subnet you wish to use for the cloudSwXtch is listed.** In addition to the cloudSwXtch installation, this single subnet will be used during xNIC installation.

Method #2: Two Subnets

Alternatively, a user may decide that they want to have two separate subnets for their cloudSwXtch: one for the control plane and another for data. In addition, the same subnets must be used for the xNIC installations. This method is recommended for individuals who want higher performance.

To accomplish this:

1. **Navigate to the VPC Console in AWS.**
2. **Select Subnets** under the **Virtual Private Cloud** tab or under **Resources by Region** in the VPC Dashboard.
3. **Check that 2 subnets exist:** one for the data and another for the control plane. Ensure that both subnets are in the same **Availability Zone**. This allows both NICs to be connected on the EC2 instance at the same time.

Naming your subnets

For ease of use, name the subnets ctrl-subnet and data-subnet to distinguish between them when creating an EC2 instance with cloudSwXtch installed.

Subnets (2) Info

Filter subnets

Network ACL: acl-016f53e386af5b3e7 X Availability Zone: us-west-2b X Route table: rtb-0f2f26b2092f9d9d5 X Clear filters

	Name	Subnet ID	State	VPC	IPv4 CIDR	IPv6...	Aval...	Availability Zone
<input type="checkbox"/>	ctl_subnet	subnet-026043d5e098216ef	Available	vpc-032478a302937fd9e SA-Test-V...	172.31.16.0/20	-	4073	us-west-2b
<input type="checkbox"/>	data_subnet	subnet-04c26b015009b4c3f	Available	vpc-032478a302937fd9e SA-Test-V...	172.31.132.0/22	-	1005	us-west-2b

4. If a second subnet does not exist, select the orange **Create Subnet** button in the top right corner of the page.

Subnets (1/2) Info

Filter subnets

Network ACL: acl-016f53e386af5b3e7 X Availability Zone: us-west-2b X Route table: rtb-0f2f26b2092f9d9d5 X Clear filters

	Name	Subnet ID	State	VPC	IPv4 CIDR	IPv6...	Aval...	Availability Zone
<input checked="" type="checkbox"/>	ctl_subnet	subnet-026043d5e098216ef	Available	vpc-032478a302937fd9e SA-Test-V...	172.31.16.0/20	-	4073	us-west-2b
<input type="checkbox"/>	data_subnet	subnet-04c26b015009b4c3f	Available	vpc-032478a302937fd9e SA-Test-V...	172.31.132.0/22	-	1005	us-west-2b

5. Fill in the **Create Subnet** form like the example shown below, ensuring that the subnet is in the same VPC ID and **Availability Zone** as your other subnet. In the example below, the user is creating their data subnet.

VPC > Subnets > Create subnet

Create subnet Info

VPC

VPC ID
Create subnets in this VPC.

vpc-032478a302937fd9e

Associated VPC CIDRs

IPv4 CIDRs

172.31.0.0/16

Subnet settings

Specify the CIDR blocks and Availability Zone for the subnet.

Subnet 1 of 1

Subnet name
Create a tag with a key of 'Name' and a value that you specify.

data_subnet

Availability Zone Info
Choose the zone in which your subnet will reside, or let Amazon choose one for you.

US West (Oregon) / us-west-2b

IPv4 CIDR block Info

.31.133.0/22

Tags - optional

Key Value - optional

Name data_subnet Remove

Add new tag

You can add 49 more tags.

Remove

Add new subnet

Cancel Create subnet

6. Click **"Create Subnet."** You should now have a new subnet on your list.

Verify Security Groups

The security group contains the firewall settings for EC2 instances and interfaces (xNICs).

To ensure security groups are set up properly for cloudSwXtch:

1. Navigate to the VPC console.
2. Select the "Security Groups" link as shown below. (Note: There are multiple ways to get to the "Security Groups" page.)

The screenshot shows the AWS VPC console interface. On the left-hand navigation menu, under the 'SECURITY' section, the 'Security Groups' link is highlighted with a red rectangle. The main content area displays 'Resources by Region' for the US West region, listing various VPC resources. The 'Security Groups' resource is also highlighted with a red rectangle, showing it has 5 resources in the US West region.

3. Select the Security Group that is normally used to create your EC2 instances for your application. (Note: The names in the example will be different in your environment.)

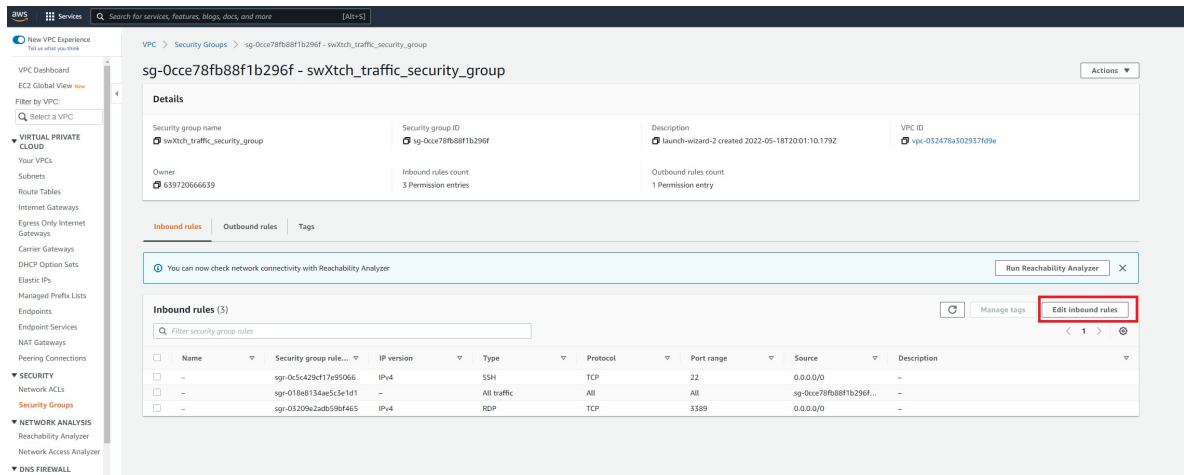
The screenshot shows the 'Security Groups' page in the AWS VPC console. A table lists several security groups. One security group, 'sg-0c5429c17e95066', is highlighted with a red rectangle. The table columns include Name, Security group ID, Security group name, VPC ID, Description, Owner, Inbound rules count, and Outbound rules count.

4. In order for certain features to work in your cloudSwXtch, you will need to add inbound rules to open specific ports originating from that security group. You can find the ports outlined in the [cloudSwXtch System Requirements](#) article under "Firewall and Security Group Rules."

The screenshot shows the 'Inbound rules' tab for a specific security group, 'sg-0c5429c17e95066'. The page displays details about the security group, including its name, ID, description, and owner. Below the details, there is a section for 'Inbound rules (3)'. Three inbound rules are listed and highlighted with a red rectangle:

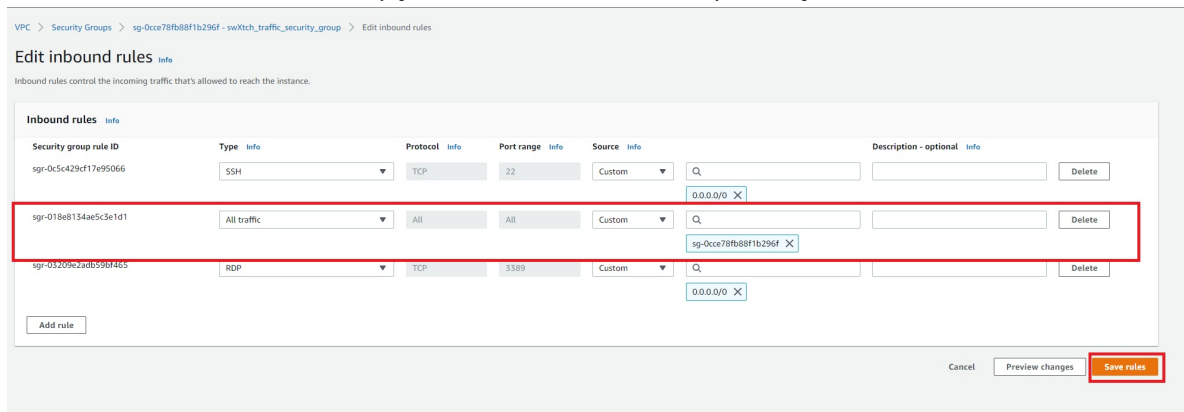
Name	Security group rule name	IP version	Type	Protocol	Port range	Source	Description
	sg-0c5429c17e95066	IPv4	SSH	TCP	22	0.0.0.0/0	
	sg-018a8154ae5c3e161	IPv4	All traffic	All		sg-0c5429c17e95066	
	sg-03209c2adb59f465	IPv4	RDP	TCP	3389	0.0.0.0/0	

5. If an inbound rule does not exist, create it by selecting "Edit inbound rules."



6. Select "Add Rule."

7. Enter the information like the screenshot shown below verifying that the ID of the SG on Source matches the SG you are editing.



8. Save the rule.

Additional Rules

Mandatory Inbound Rule For Mesh

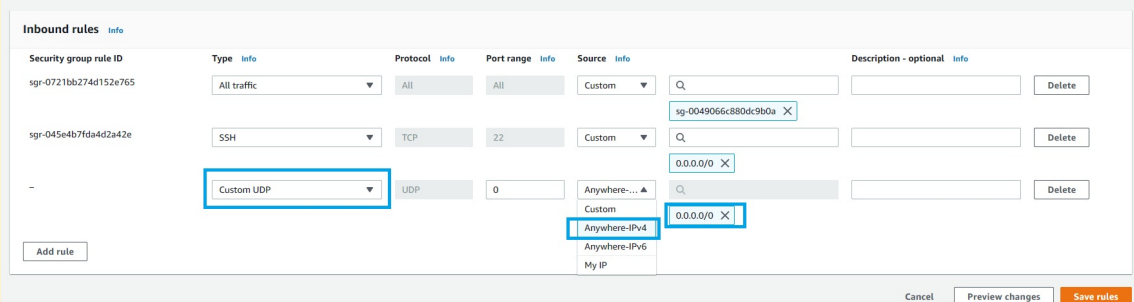
In order to use the Mesh feature bidirectionally between VPCs, users must also add the following inbound rule to each SG:

- **Type:** Custom UDP
- **Protocol:** UDP
- **Port Range:** 9999
- **Source:** Custom/Anywhere-IPv4 0.0.0.0/0

EC2 > Security Groups > sg-0049066c880d9b0a - SWLJ2L00A60-CreatedSwXtchSecurityGroup-1ES7NW0Z1UDWZ > Edit inbound rules

Edit inbound rules

Inbound rules control the incoming traffic that's allowed to reach the instance.



Create SSH Key Pair

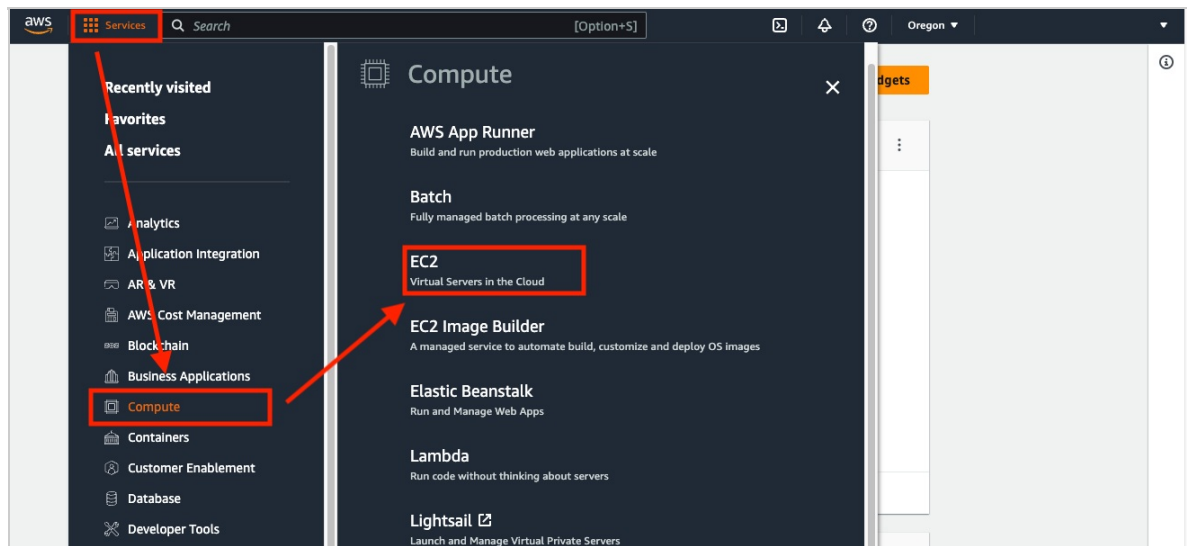
WHAT TO EXPECT

An SSH key pair is necessary when accessing a cloudSwXtch EC2 instance. If you do not already have one imported, please create an SSH key pair before beginning the cloudSwXtch on AWS creation process.

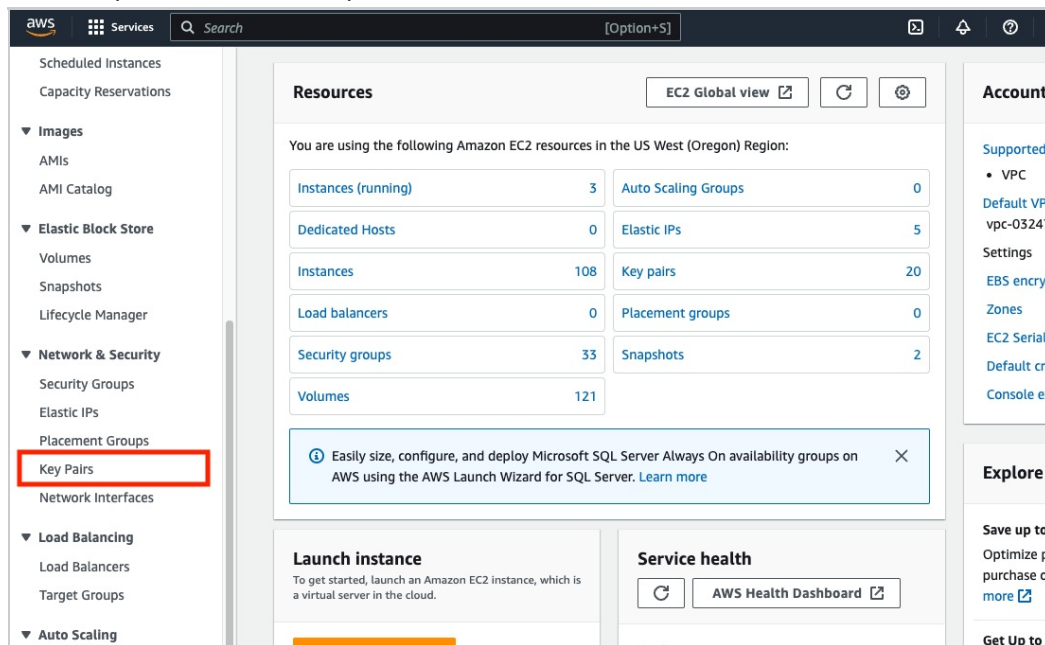
In the *AWS Management Console*, make sure you are in the region where you plan to use the cloudSwXtch instance.

1. Navigate to EC2

- Select the "Services" menu in the AWS Management Console.
- Click "Compute"
- Select "EC2"



2. In EC2, click "Key Pairs" under the "Network & Security" tab in the menu on the left-hand side.



3. Click "Create Key Pair". A new window should open.

4. Under **Name**, enter something meaningful and descriptive for the key.

5. Depending on your needs, you have to choose **RSA** or **ED25519**, and **.pem** or **.ppk** (OpenSSH or PuTTY access).

6. Click on **Create Key Pair**.

- a. A file with the chosen extension will be downloaded to your computer (secret private key), and the other half of the pair will be store on AWS for later use (public key, used in conjunction with your private key to validate the access).

Create key pair [Info](#)

Key pair

A key pair, consisting of a private key and a public key, is a set of security credentials that you use to prove your identity when connecting to an instance.

Name

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type [Info](#)

☒ RSA

☐ ED25519

Private key file format

☒ .pem

For use with OpenSSH

☐ .ppk

For use with PuTTY

Tags - *optional*

No tags associated with the resource.

[Add new tag](#)

You can add up to 50 more tags.

[Cancel](#)

[Create key pair](#)

Install cloudSwXtch on AWS

WHAT TO EXPECT

Deployment of a cloudSwXtch consists of two parts: the creation of an EC2 instance containing cloudSwXtch and the installation of the xNIC software. The cloudSwXtch is considered "installed" once while the xNIC is installed on each agent instance that is a part of the network.

In this section, users will learn how to deploy cloudSwXtch for their AWS environment.

NOTE:

Root privileges are not required for deployment or operation. Our CloudFormation template allows an automated mechanism to update the installed cloudSwXtch version. This will deploy the latest version of the cloudSwXtch instead of the one packaged in the AMI, which requires root privileges to trigger the update from the product side. For upgrades, please see [Upgrade cloudSwXtch on AWS](#) on how to perform an upgrade from the client side. An upgrade from the client side does not require root privileges.

Creating a cloudSwXtch EC2 Instance

Prerequisites

Before starting, a user must do the following:

1. Review [cloudSwXtch System Requirements](#).
2. Ensure that you already have an AWS account.
3. Create a virtual network (VPC). This *must* be created before deploying a cloudSwXtch.
4. Validate you have at least one subnet for your virtual network. A single subnet can be used for the control and data plane.
5. Verify a Security Group that allows access to all traffic inside the VPC. If one is not created, use default when creating a cloudSwXtch.
6. Create an SSH Key Pair.

Post-Installation

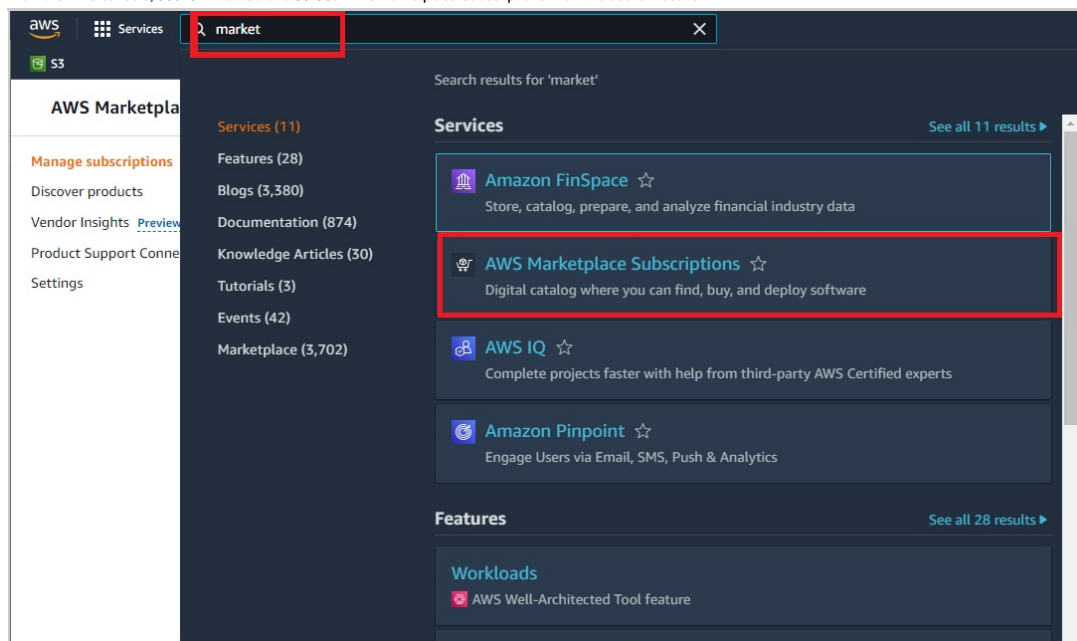
The following instructions detail how to deploy a cloudSwXtch BYOL instance. If a user decides to deploy a BYOL instance, they will need to complete [the additional step of licensing their cloudSwXtch](#).

If all prerequisites are met, a cloudSwXtch can be created via the Marketplace in any region in approximately 10 minutes. If multi-AZ or multi-region is required then see [Mesh](#) for details. The installer will create a CloudFormation Stack to include the following resources:

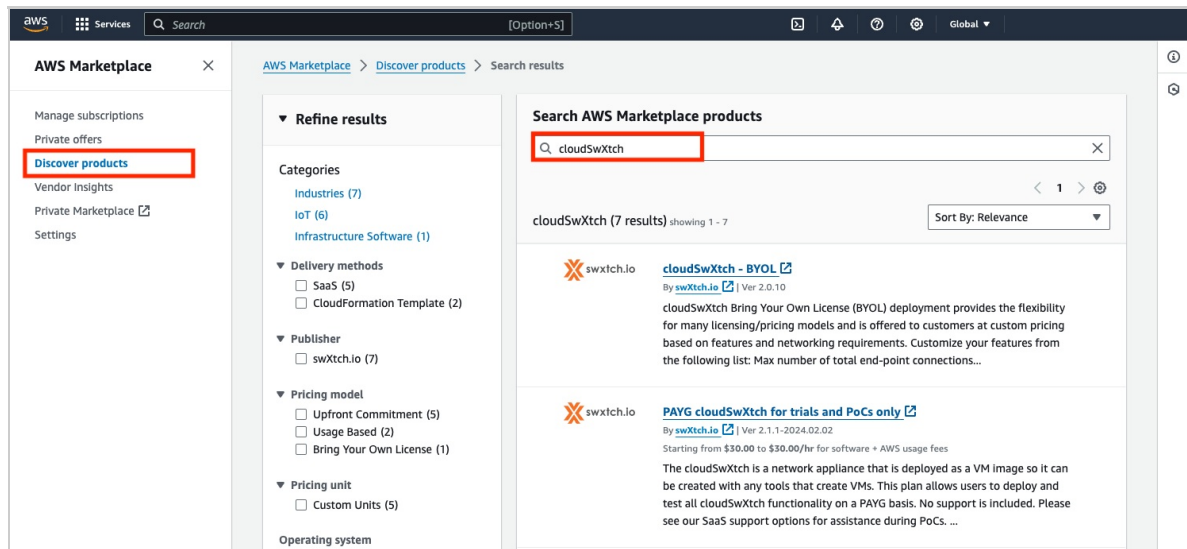
- ControlEni Networking Interface for control data
- DataEni Networking Interface for data such as Multicast
- EC2Instance in Linux for the cloudSwXtch to run on

In order to create a cloudSwXtch, please do the following steps.

1. Sign into AWS.
2. From the AWS console, search "Market" and select "AWS Marketplace Subscriptions" from the search results.



3. Select "Discover Products" in the AWS Marketplace menu on the left hand side.
4. Search for "cloudSwXtch."



5. Select a Tier (cloudSwXtch - BYOL or cloudSwXtch - PAYG) based on your usage requirements and features needed.

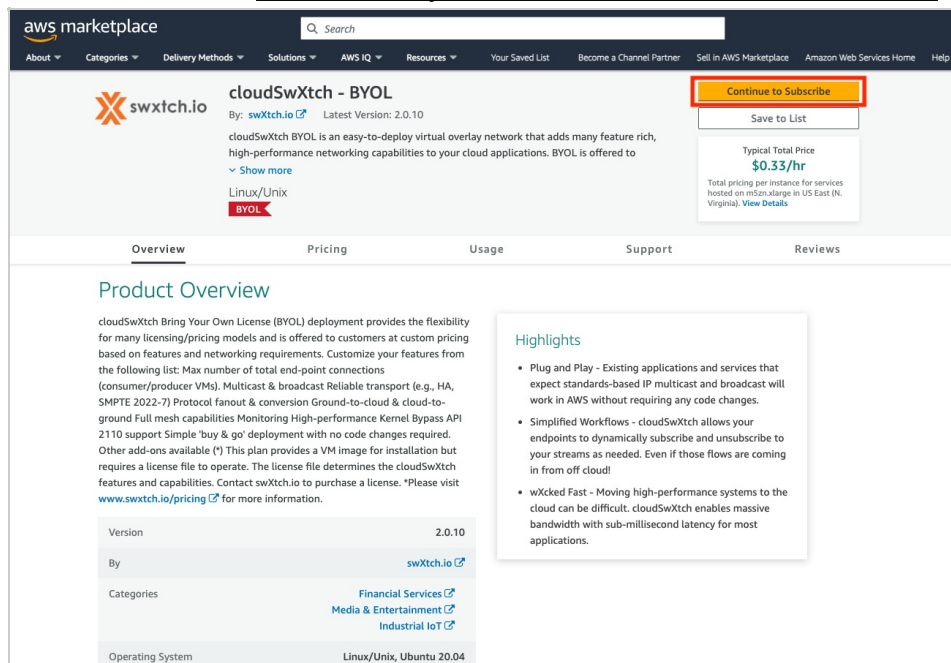
- Please read the [cloudSwXtch System Requirements](#) article for more information regarding cloudSwXtch sizing.
- For the purpose of this guide, the next screenshots will be for a cloudSwXtch BYOL deployment.

Endpoint Connections Limit

Be mindful of the number of endpoints you connect to your cloudSwXtch after creation. For cloudSwXtch BYOL, you are limited to the number of endpoints allowed in your license.

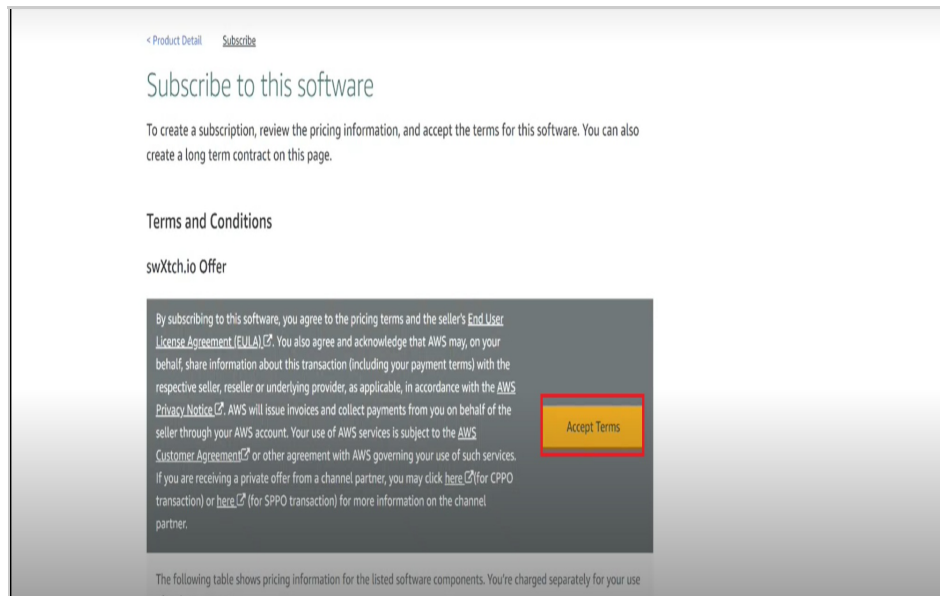
If you need to increase the number of endpoints, please view the AWS instructions [here](#). Note that if your new instance type exceeds the size of your tier, you must contact support@swxtch.io to update your license.

6. Select "Continue to Subscribe" after reviewing the product information. Note: The "Typical Total Price" is calculated with the recommended instance size included in the final monthly value and a utilization of 24x7. **Please note: The cost in "Software Pricing Details" is for the cloudSwXtch and does not include costs for the AWS instance.**

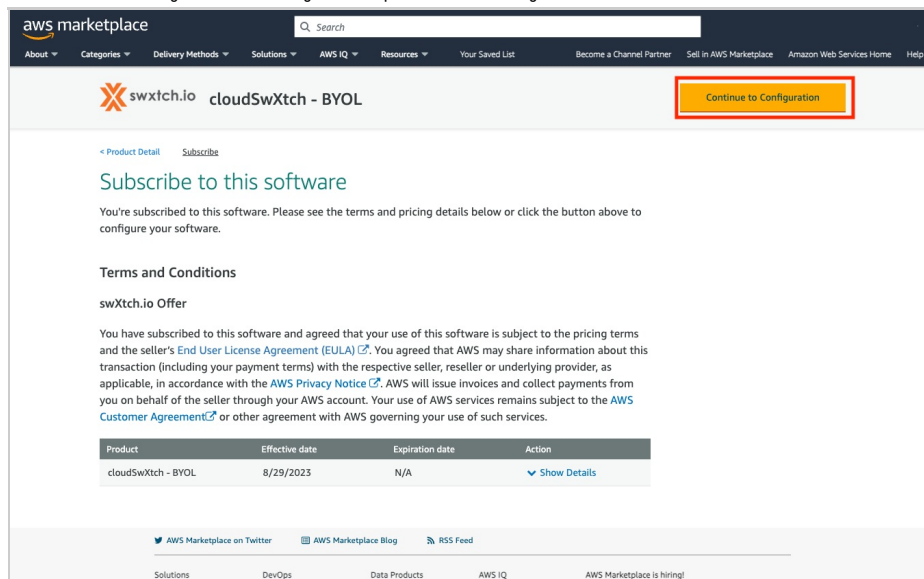


7. Review the Terms and Conditions.

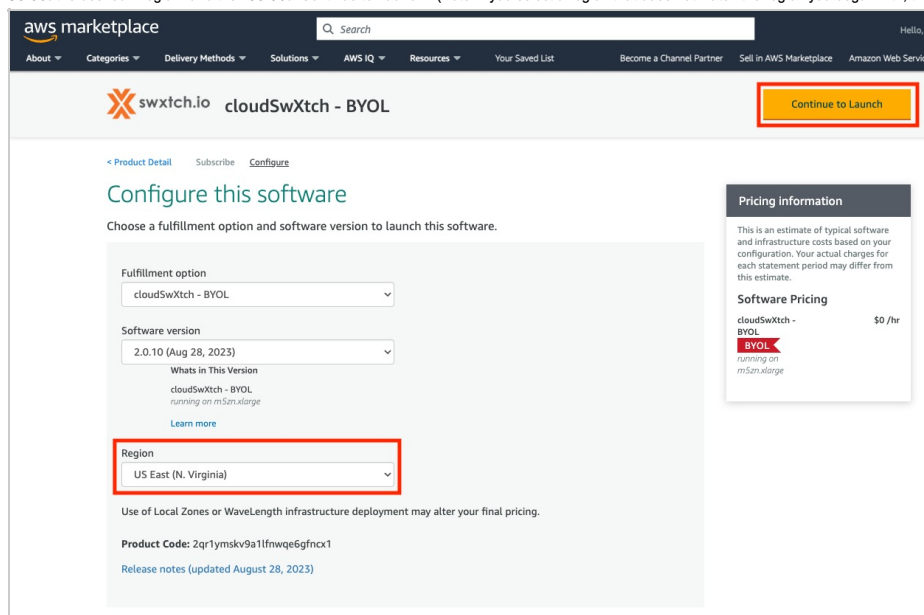
8. Select "Accept Terms" if they are acceptable.



9. Select "Continue to Configuration" after reading the subscription and license management.



10. Select the desired "Region" and then select "Continue to Launch". (Note: If you select a region that does not match the region you began with, then it may not work even if selected here.)



INSTANCE TYPES

It is recommended to use an instance type in the **c6in** or **m6in** families. Remember that a cloudSwXtch requires a minimum of 8 cores. Note how the cloudSwXtch Marketplace install selects the appropriate VM size in the Fulfillment section based on the cloudSwXtch tier.

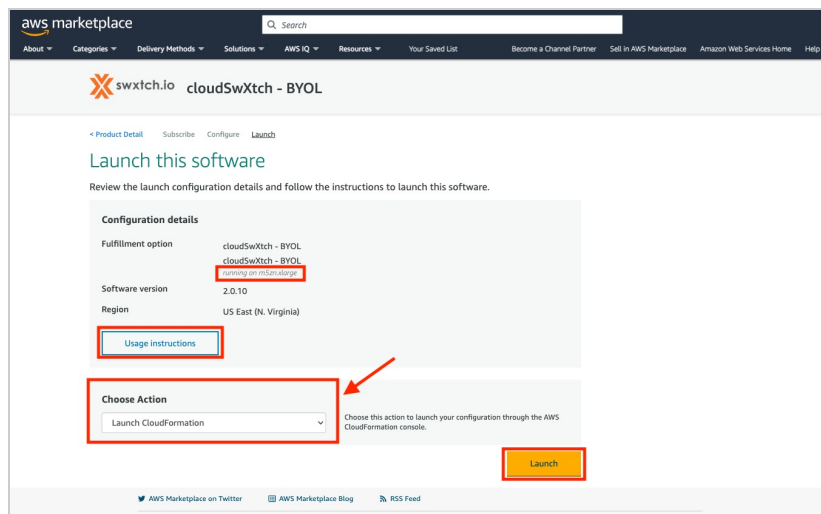
Please ensure that the instance type matches one of the options below:

- c6in.2xlarge
- c6in.4xlarge
- c6in.8xlarge
- c6in.12xlarge
- c6in.16xlarge
- c6in.24xlarge
- c6in.32xlarge
- m6in.2xlarge
- m6in.4xlarge
- m6in.8xlarge
- m6in.12xlarge
- m6in.16xlarge
- m6in.24xlarge
- m6in.32xlarge

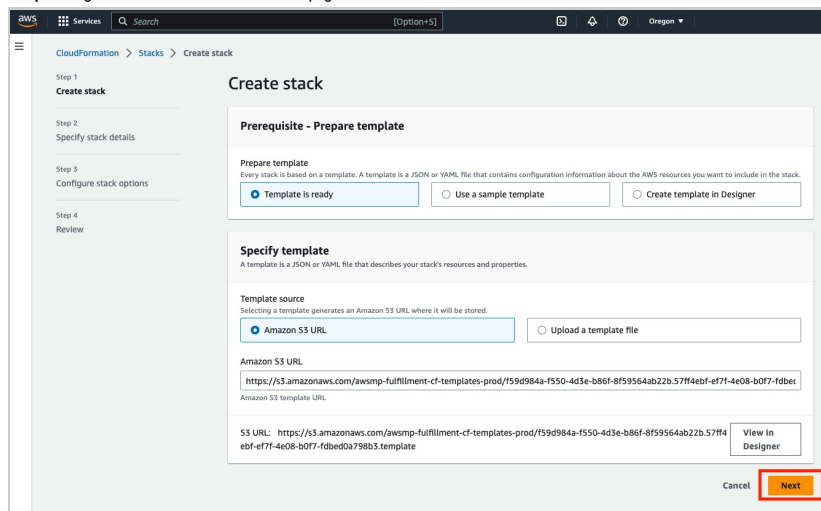
11. Read "Usage Instructions" if you desire.

12. Use the "Choose Action" dropdown menu and select "Launch CloudFormation".

13. Click "Launch".



14. Keep Settings on default on the "Create Stack" page and select "Next."



15. On the Specify stack details page, complete the following:

- Under "Stack name," enter your desired name. Keep in mind that this will be used for everything added to the stack. For example: "resource name," "security groups," "EC2 instance name," etc.
- Under "CidrIpForInboundOutboundTraffic," use 0.0.0.0/0 so that you can SSH to the virtual machine from any IP address. You can also pick a more restrictive range if desired.
- Under "ControlSubnet," use the dropdown to find the control subnet you created (recommended: *ctlr-subnet*).
- Under "DataSubnet," use the dropdown to find the control subnet you created. Both control and data can share the same subnet.
 - Alternatively, for better performance, a user can assign a separate subnet for their data subnet.
- For "InstanceType," there should be "Fulfillment" data from the earlier step.
- Under "KeyName," use the dropdown to find your previously created or imported SSH key.

- g. In "PassedSwTchSecruityGroup," use "default" and one will be created during the installation process. Alternatively, you can enter the ID of an already created security group. It will be something similar to "sg-009273855418af38d."
- h. Under "VpcId," select from the dropdown to find the already created VPC id.
- i. Here is an example of how your template would look like:

Specify stack details

Stack name

Stack name

Test-Swxtch-01

Stack name can include letters (A-Z and a-z), numbers (0-9), and dashes (-).

Parameters

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

CidrIpForSSHAccess

For SSH access when using default security group, please set CIDR to x.x.x.x/32 to allow one specific IP address access, 0.0.0.0/0 to allow all IP addresses access, or another CIDR range

0.0.0.0/0

ControlSubnet

Used to create control ENI assigned to the swtXtch control subnet.

subnet-0f956735b224e2178

DataSubnet

Used to create control ENI assigned to the swtXtch control subnet.

subnet-07de773cb1faeec38

InstanceType

EC2 instance type to use for swtXtch creation

m5zn.3xlarge

KeyName

SSH key name for swtXtch SSH access

PassedSwtchSecurityGroup

Name of the security group that should be used by swtXtch and created ENIs

sg-0702ad5e67a99c4b8

VpcId

The VPC id your swtXtch will be placed in

vpc-0941089258f89c9a2

Cancel

Previous

Next

16. Click "Next."
17. The "Configuring stack options" page is completely optional. You can assign tags for your stack, set additional IAM permissions, stack failure options, etc.
18. Click "Next" if you don't need to make any changes.
19. Verify that your parameters are accurate on the final "Review" page. If you need to change anything, select "Edit."

CloudFormation > Stacks > Create stack

Step 1: Specify template

Step 2: Specify stack details

Step 3: Configure stack options

Step 4: Review

Review Test-Swxtch-01

Step 1: Specify template

Template

Template URL

https://s3.amazonaws.com/awsump-f fulfillment-cf-templates-prod/7530984a-f550-4d3a-b86f-8f39564ab32b/x5509f5-0008-448b-b365-170a0b5c4865/template

Stack description

-

Estimate cost

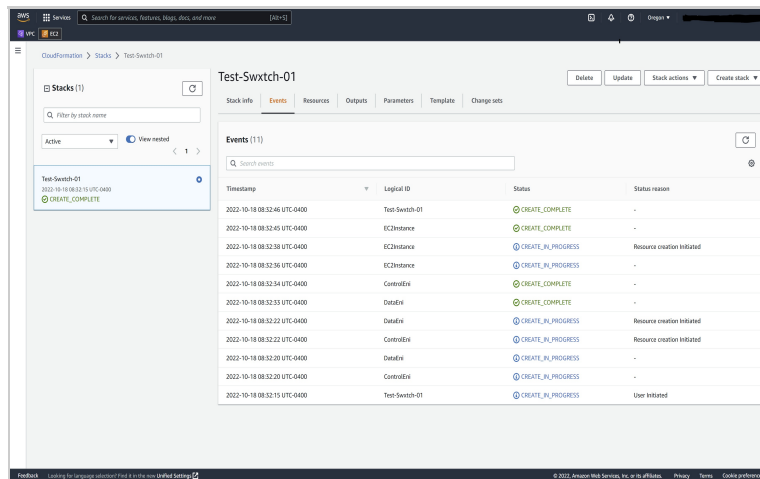
Step 2: Specify stack details

Parameters (7)

CL Filter

Key	Value	Resolved value
CidrIpForSSHAccess	0.0.0.0/0	-
ControlSubnet	subnet-0f956735b224e2178	-
DataSubnet	subnet-07de773cb1faeec38	-
InstanceType	m5zn.3xlarge	-
KeyName	eme-agent	-
PassedSwTchSecurityGroup	sg-0702ad5e67a99c4b8	-
VpcId	vpc-0941089258f89c9a2	-

20. Click "Submit." On the next page, you can view the creation of your stack.



Your EC2 instance has now been created. You can view it on the EC2/Instances list and connect to your cloudSwXtch from there.

21. Once you have connected with SSH to your cloudSwXtch as root user (sudo su), navigate to the cloudSwXtch directory (cd /swxtch) then run the following command:

```
Bash
Copy
sudo swxtch/swxtch-top dashboard --swxtch <cloudSwXtch-IP>
```

NOTE

Use the cloudSwXtch-name in place of the IP address if DNS resolution is setup or "localhost."

This will display the cloudSwXtch's swxtch-top dashboard. In "Status," you should see "OK." This will let you know that your cloudSwXtch has been successfully deployed. You can review more information regarding swxtch-top in the [swxtch-top article](#).

INSTALLING AN xNIC

If this is a new installation, then each client that is expected to receive or transmit to the cloudSwXtch will need an xNIC installed.

If this is an existing cloudSwXtch replacement, then each client with an xNIC already installed will need to be upgraded to match the current cloudSwXtch version.

You can find more information about xNIC installation, [here](#).

Required Step for BYOL: Contact swXtch.io for a license

Users deploying a BYOL instance of cloudSwXtch will need to contact swXtch.io for a license file. For more information, see [How to License a cloudSwXtch](#).

Checking the Health of Your cloudSwXtch Instance

It is important to ensure your AWS system is healthy. AWS provides AWS CloudWatch as a way to check on the health of your system. To check on the cloudSwXtch EC2 instance, read more [here](#).

Upgrading cloudSwXtch on AWS

It is important that your cloudSwXtch instance is up to date. To learn how to upgrade your cloudSwXtch, you can read more [here](#).

Deleting cloudSwXtch on AWS

To learn how to delete your cloudSwXtch, you can read more [here](#).

Deploy cloudSwXtch with Terraform on AWS

WHAT TO EXPECT

In this article, you will learn how to deploy a cloudSwXtch instance on AWS using a Terraform script.

Prerequisites:

For this script to work, you will need to have already provisioned your VPC, Subnets, and SSH Keys. You will plug those parameter values into your [AWS/terraform/terraform.tfvars](#) file.

Deploying a Terraform Script

1. Choose what platform you would like to run Terraform on. For this example, the user is on a Linux machine. Download intructions can be found at: [terraform.io/downloads](#)
2. Clone the repository using SSH. **Please note:** You will need an SSH key set up with GitHub.

ConsoleCopy

\$ git clone git@github.com:swxtchio/cloudSwXtch-support

Or, alternatively, you can clone with the HTTPS URL: ([here on GitHub](#))

ConsoleCopy

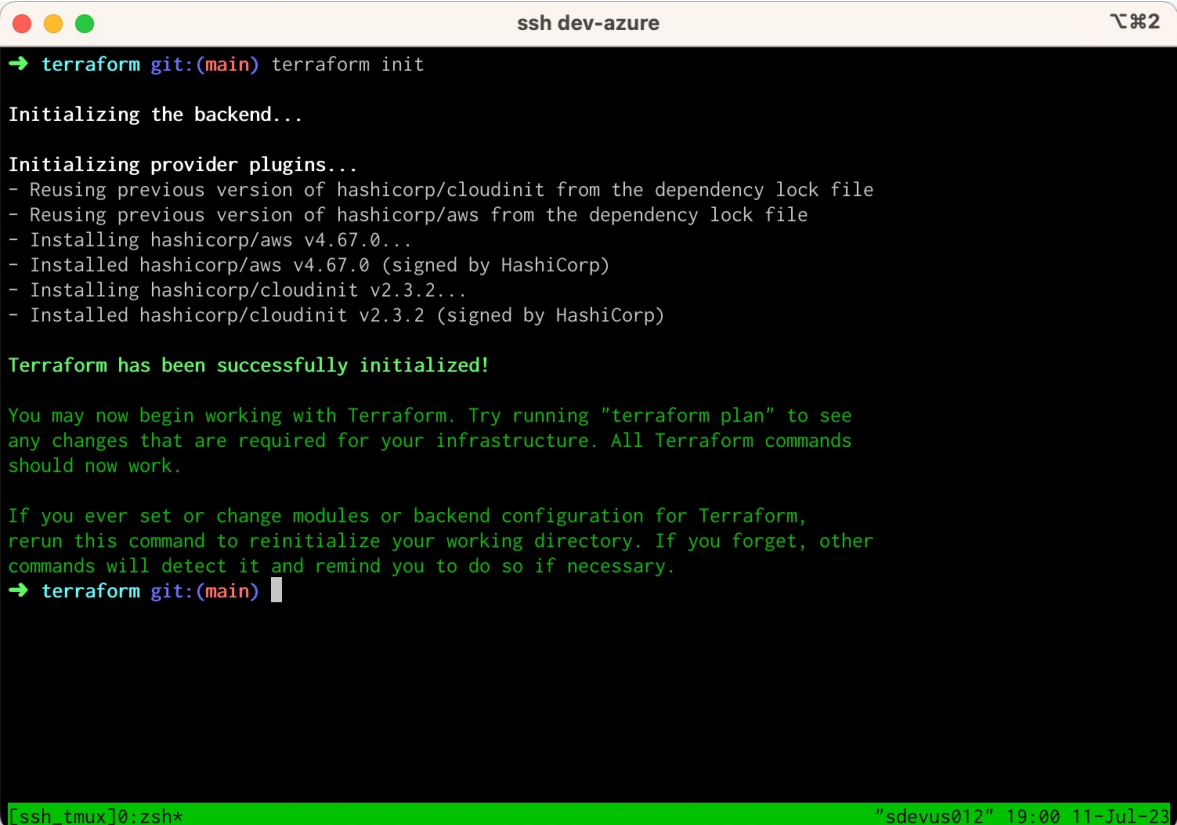
\$ git clone https://github.com/swxtchio/cloudSwXtch-support.git

Then:

ConsoleCopy

\$ cd cloudSwXtch-support/AWS/terraform

3. Update the values in the [AWS/terraform/terraform.tfvars](#) file to match your existing AWS resources such as: VPC id, Subnet IDs, and SSH Key names.
4. Run `terraform init` inside the [AWS/terraform/](#) directory.



5. For this step you'll need to have authenticated your AWS credentials inside the console. Or you can pass the credentials with environmental variables. One simple way to do this is using an Access Key (AK). If you don't have one, you can generate your AK in Amazon's IAM->Users->(your user), on the Security Credentials tab, Access Keys. Then, you have to export the following:

ConsoleCopy

\$ export AWS_ACCESS_KEY_ID="anaccesskey"
\$ export AWS_SECRET_ACCESS_KEY="asecretkey"
\$ export AWS_REGION="us-west-2"

Now that Terraform has been initialized and you're authenticated, run this command to evaluate the config and confirm the desired output which will be shown:

Console

Copy

\$ terraform plan

ssh dev-azure

2

```
+ tags                = {
  + "Name" = "swxtch_traffic_swxtch-tf-example_sg"
}
+ tags_all            = {
  + "Name" = "swxtch_traffic_swxtch-tf-example_sg"
}
+ vpc_id              = "vpc-06974b4b531c0f697"
}
```

Plan: 7 to add, 0 to change, 0 to destroy.

Changes to Outputs:

```
+ swxtches = [
  + {
    + ctrl_ip = (known after apply)
    + data_ip = (known after apply)
    + username = "ubuntu"
  },
  + {
    + ctrl_ip = (known after apply)
    + data_ip = (known after apply)
    + username = "ubuntu"
  },
]
```

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.

→ terraform git:(main)

[ssh_tmux]0:zsh* "sdevus012" 19:02 11-Jul-23

This is the result of running `terraform plan` with 2 cloudSwxtches. The provided terraform example creates a security group for all of the deployed resources: two `aws_network_interface` for each of the cloudSwxtches and an `aws_instance` for each cloudSwxtch.

6. Run the Terraform apply command (followed by "yes" when prompted) to approve the action.

Console

Copy

terraform apply
yes

7. Once the resources have been applied successfully, you should see an output similar to this:


```
ssh dev-azure 2

aws_network_interface.swxtch_data[1]: Creating...
aws_network_interface.swxtch_data[0]: Creating...
aws_network_interface.swxtch_ctrl[1]: Creation complete after 1s [id=eni-099a6704b48871b65]
aws_network_interface.swxtch_data[0]: Creation complete after 1s [id=eni-05c53da54a04ea3be]
aws_network_interface.swxtch_data[1]: Creation complete after 1s [id=eni-0c364aa6898a7e2a2]
aws_network_interface.swxtch_ctrl[0]: Creation complete after 1s [id=eni-0c765fb2ad379495b]
aws_instance.swxtch[0]: Creating...
aws_instance.swxtch[1]: Creating...
aws_instance.swxtch[0]: Still creating... [10s elapsed]
aws_instance.swxtch[1]: Still creating... [10s elapsed]
aws_instance.swxtch[1]: Creation complete after 14s [id=i-06032053dbf5a744e]
aws_instance.swxtch[0]: Creation complete after 14s [id=i-08ba3f7cfa686764c]

Apply complete! Resources: 7 added, 0 changed, 0 destroyed.

Outputs:

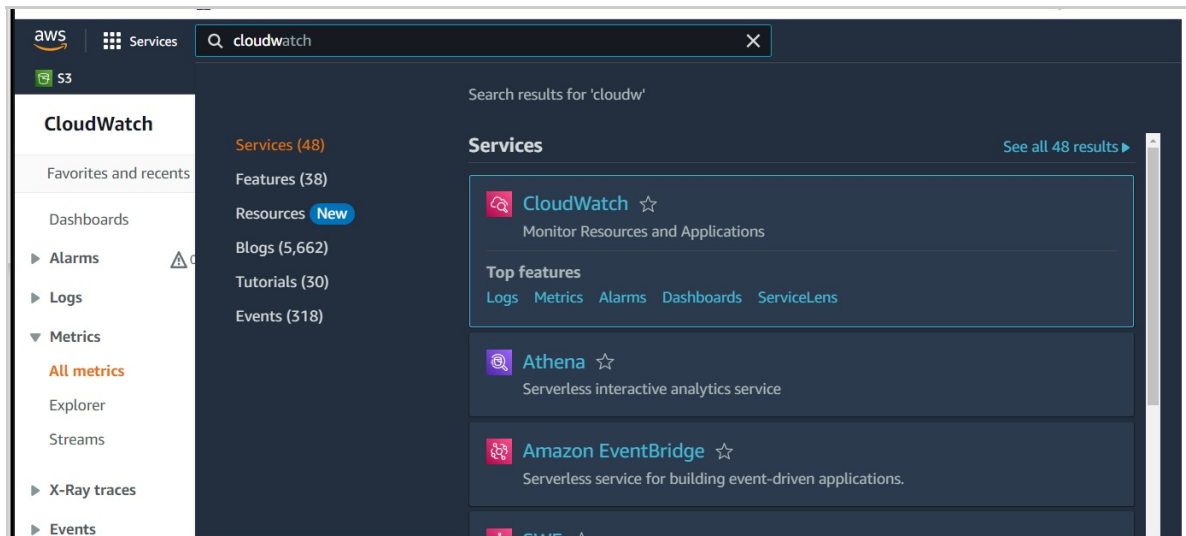
swxtches = [
  {
    "ctrl_ip" = "172.31.33.152"
    "data_ip" = "172.31.132.216"
    "username" = "ubuntu"
  },
  {
    "ctrl_ip" = "172.31.46.220"
    "data_ip" = "172.31.133.206"
    "username" = "ubuntu"
  },
]
→ terraform git:(main) [ssh_tmux]0: terraform* "sdevus012" 19:05 11-Jul-23
```

You can view the resources created from your AWS portal as confirmation of a successful deployment.

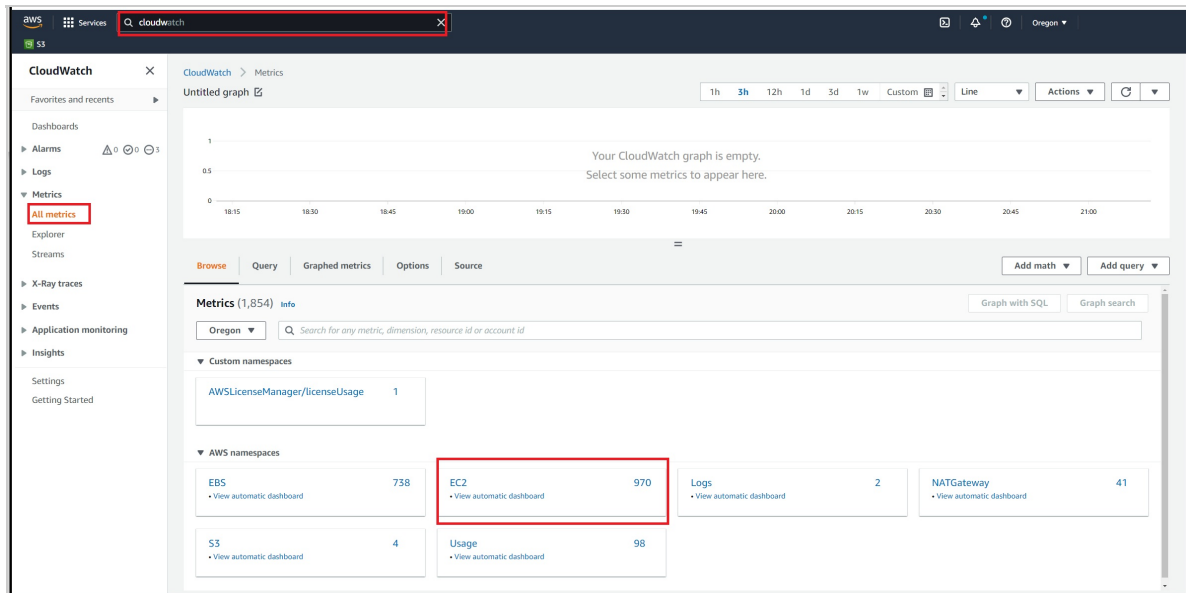
Check Health of cloudSwXtch Instance on AWS

It is important to ensure your AWS system is healthy. AWS provides AWS CloudWatch as a way to check on the health of your system. To check on the cloudSwXtch EC2 instance:

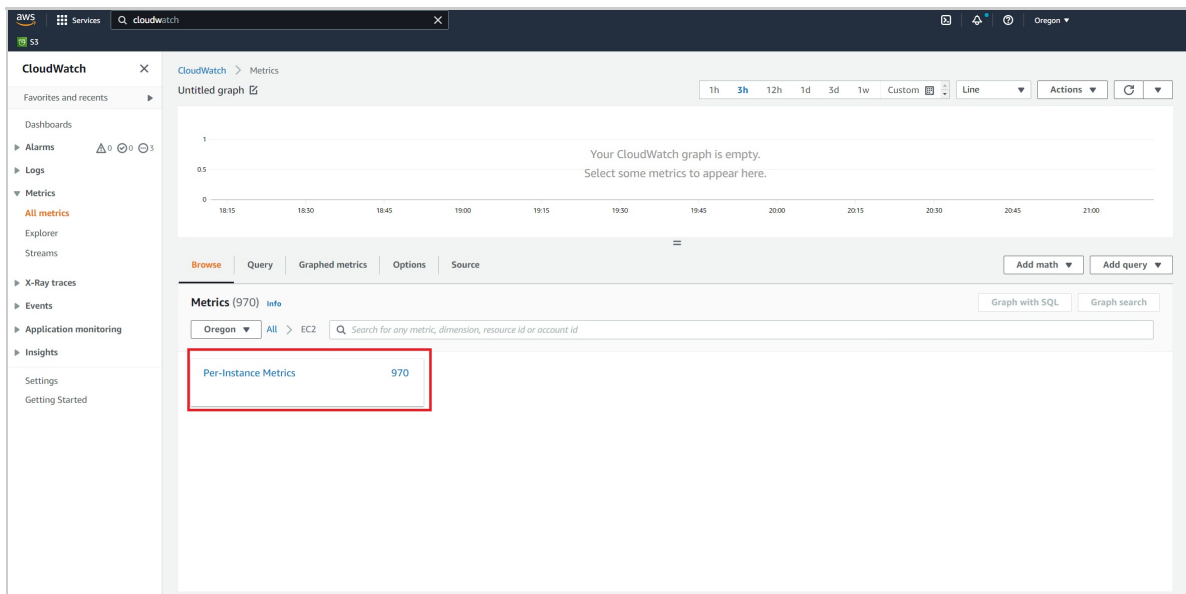
1. Search for "CloudWatch" in the AWS Search bar.



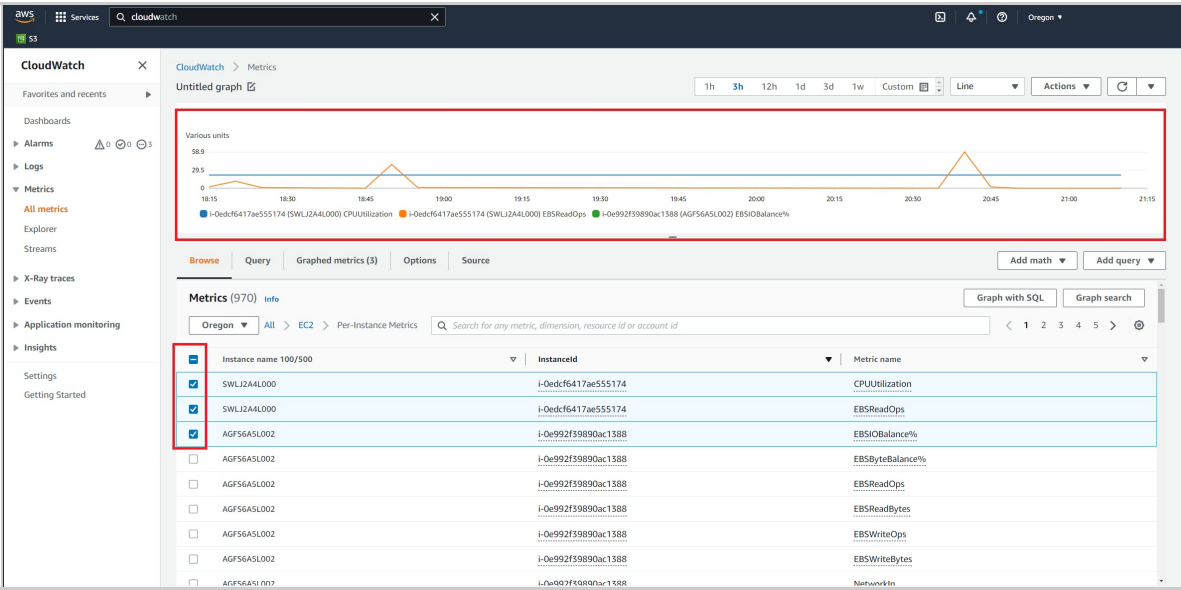
2. Select "All Metrics" on the left tree menu under "Metrics."
3. Select "EC2."



4. Select "Per-Instance Metrics."



5. Sort as desired. Instance ID works well.
6. View data in graph.



WARNING

A cloudSwXtch instance will consume CPU even when the connected agents are not producing/consuming data. This is because there are several vCPUs configured to constantly watch the interfaces.

Delete cloudSwXtch on AWS

WHAT TO EXPECT

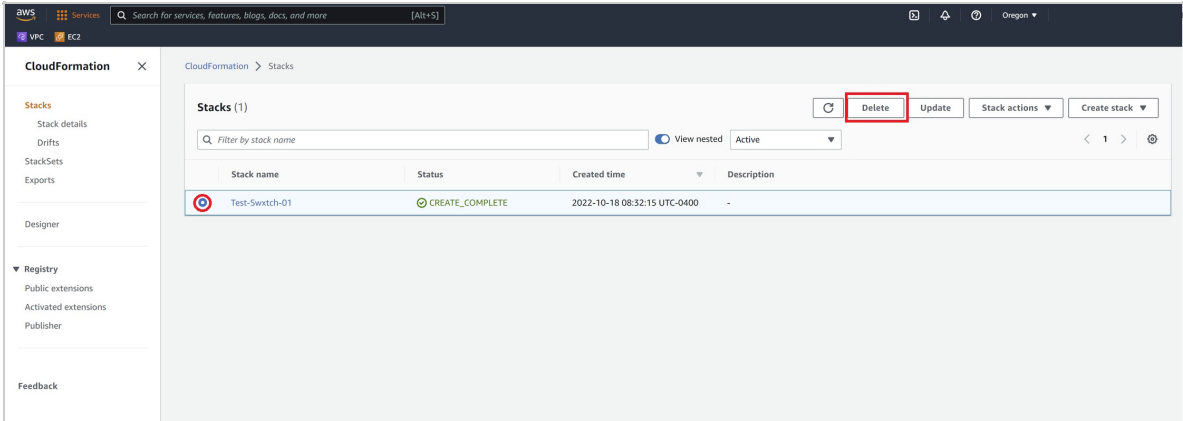
In this section, users will learn how to delete cloudSwXtch from their AWS environment.

Prior to deleting a cloudSwXtch, it is advised to uninstall any xNICs using it. See [xNIC Installation](#).

It is important to note that since your cloudSwXtch was created using a Stack, you do not want to just delete the EC2 instance by itself. Rather, you will want to delete the Stack as whole, which will also delete all associated resources as well.

To delete a cloudSwXtch:

- 1. **Navigate** to your cloud stack: "Cloud Formation → Stacks"



- 2. **Select** the stack you want to delete.
- 3. **Click "Delete"** and then confirm on the popup window.
- 4. **Refresh the page after a minute or so** to confirm the stack has been deleted.

cloudSwXtch on Azure

Pre-installation Steps

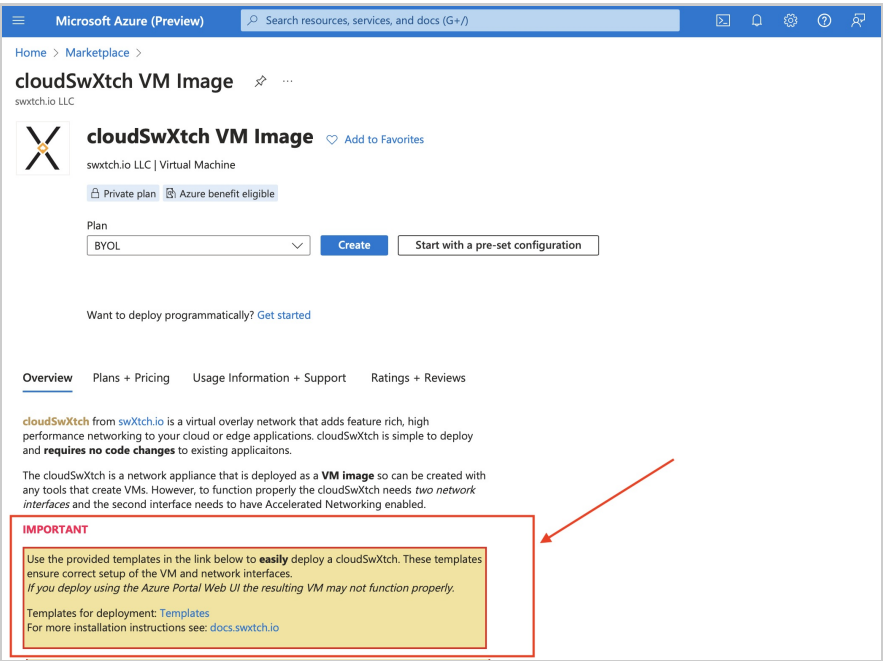
There are three methods that a cloudSwXtch instance can be deployed using the Azure Portal: via template, via Terraform, and via the Market Place.

Out of those three options, the preferred method is via template as it will create the two subnets needed for a cloudSwXtch to operate. In addition, the Network Interface will have "Accelerated Networking" enabled.

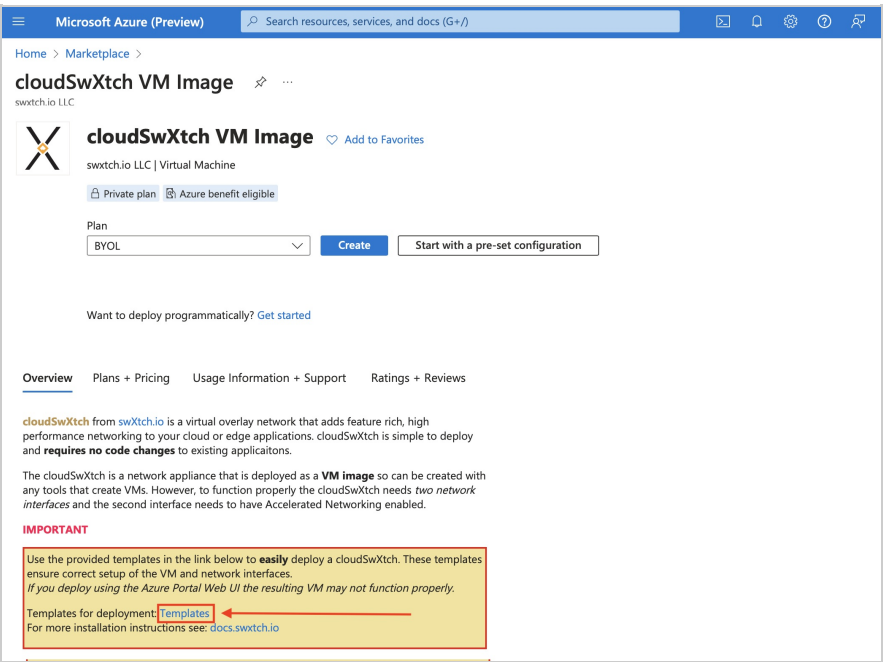
Template Method (PREFERRED):

- 1. [Review system requirements](#)
- 2. [Validate subnets on Azure](#)
- 3. [Create Azure cloudSwXtch Template](#)
- 4. [Install cloudSwXtch on Azure](#)

The template method is also mentioned in the Market Place cloudSwXtch installation as highlighted below:



Clicking on the "Templates" hyperlink shows more information about the template creation method.



Selecting this link will open the page below.

[Product](#)
[Solutions](#)
[Open Source](#)
[Pricing](#)

[Sign in](#)
[Sign up](#)

[swxtch.io / swx-cloudSwXtch-AzureTemplates](#)
Public

[Notifications](#)
[Fork 1](#)
[Star 0](#)

[Code](#)
[Issues](#)
[Pull requests](#)
[Actions](#)
[Security](#)
[Insights](#)

main

1 Branch

0 Tags

[Code](#)

brentyates-ix
Updated plans to match latest SKUs: byol & payg
5039a67 · last week
10 Commits

LICENSE	Initial commit	2 years ago
MPTemplateUI.json	Updated plans to match latest SKUs: byol & payg	last week
MPTemplateVM.json	Renamed for clarity against non-marketplace templates	2 years ago
README.md	Update README.md (#2)	2 years ago

README

MIT license

cloudSwXtch-templates

Templates to create a cloudSwXtch from the marketplace. The cloudSwXtch is deployed as a VM image so can be created with any tools that create VMs. However, to function properly the cloudSwXtch **needs two network interfaces** and the second interface needs to have Accelerated Networking enabled. Use these templates to simplify deploying a cloudSwXtch since these templates ensure correct setup of the network interfaces.

Install template into the Azure Protal

About

Templates to create a cloudSwXtch from the marketplace

- Readme
- MIT license
- Activity
- Custom properties
- 0 stars
- 8 watching
- 1 fork
- Report repository

Releases

No releases published

Packages

No packages published

Contributors 2

brentyates-ix
Brent Yates (swXtch.io)

Alternative Install Methods

Market Place

While a user can create a cloudSwXtch via the Market Place, it will require additional work in terms of maintenance. For example, the cloudSwXtch would have to be updated to add a second NIC and then have accelerated networking manually enabled. With the template method, users can bypass all this.

If you still wish to use the Market Place method, you can find more information [here](#).

Terraform

If you wish to deploy cloudSwXtch via Terraform, you can find more information [here](#).

Air-Gapped

For closed environments, users can follow the Azure Air-Gapped installation instructions [here](#).

Validate Subnets on Azure

WHAT TO EXPECT

A virtual network must be created before deploying a cloudSwXtch EC2 instance.

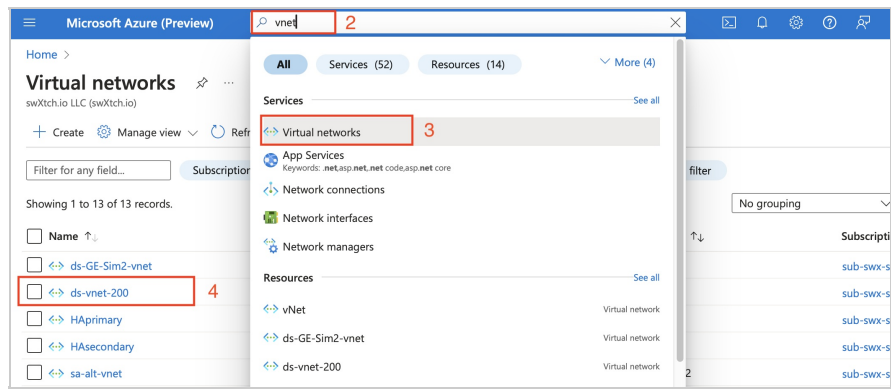
- It must contain at least one subnet that's used for both the control and data plane communication.
- It is recommended that it is **private facing** and **does not auto-assign public IPs**.
- This single subnet will be used for xNIC installation.

The subnets will also be used xNIC installations.

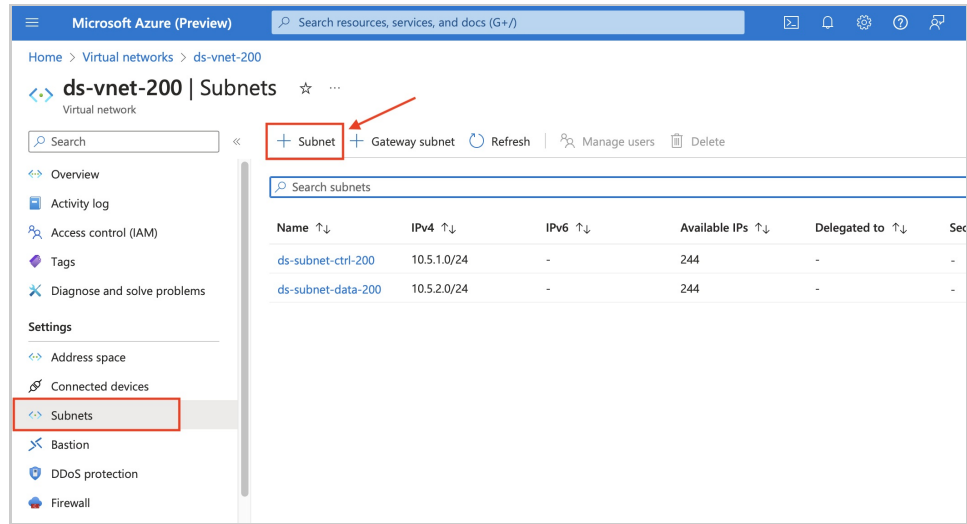
In this section, users will learn how to validate whether a subnet exists to be used as both the control and the data plane for their virtual network. This is in preparation for cloudSwXtch installation on Azure. We will also walk through an alternative method of using 2 subnets, separating the control and data plane.

To validate:

- Go to the Azure Console.
- Search for "vnet".
- Select Virtual Networks.
- Select the vnet to be used for cloudSwXtch.



- If using a single-subnet, name the subnet as "ctrl." If using two subnets, name the second one "data" to distinguish between them when creating an VM instance. Using two subnets will require them to be in the same Region (for example, East US). This enables a single VM instance to have two NICs connected to both subnets at the same time.
 - In the event that the second subnet does not exist, create it by selecting "+ Subnet."



- Enter data as shown below making sure the subnet in the same VNET and Availability zone as shown below.

sa-subnet-data

sa-vnet

Name

sa-subnet-data

Copy to clipboard

Subnet address range *

.2192.0/22

2.192.0 - 2.195.255 (1019 + 5 Azure reserved addresses)

☐ Add IPv6 address space

NAT gateway

None

Network security group

None

Route table

None

Endpoint Connections Limit

Please be mindful of the number of endpoints (virtual machines) you are allowed to connect to your cloudSwXtch after creation. For more information about sizing, please see [cloudSwXtch System Requirements](#).

NEXT STEP: Creating an Azure cloudSwXtch Template

After validating the subnets on Azure, continue on to the [Create an Azure cloudSwXtch Template](#) guide. This is in preparation for [installing cloudSwXtch on Azure](#).

Create an Azure cloudSwXtch Template

WHAT TO EXPECT

The easiest way to deploy a cloudSwXtch instance in your Azure environment is through the template method. **The following process is a one-time task per subscription.**

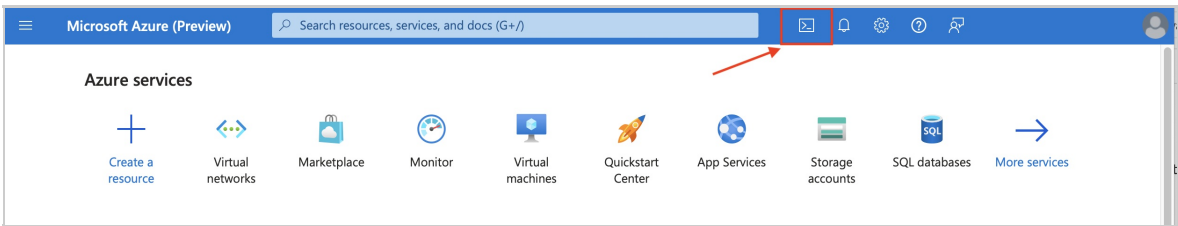
This section will walk you through the template creation process in preparation for Azure cloudSwXtch installation.

Template Creation

A cloudSwXtch template can be created by using the Azure Portal. This template will be used to create a cloudSwXtch "Creating cloudSwXtch via Template method". The template is not used during creation of a cloudSwXtch via the Market Place. The creation of the Template is a one-time task per subscription

1. Log into the Azure Portal. You will need permissions to create and manage virtual machines.
- virtual-machine-contributor

2. Open Cloud Shell.



If you need help setting up your Azure cloud-shell, use the link below for setup instructions.
[azure cloud-shell quick start](#)

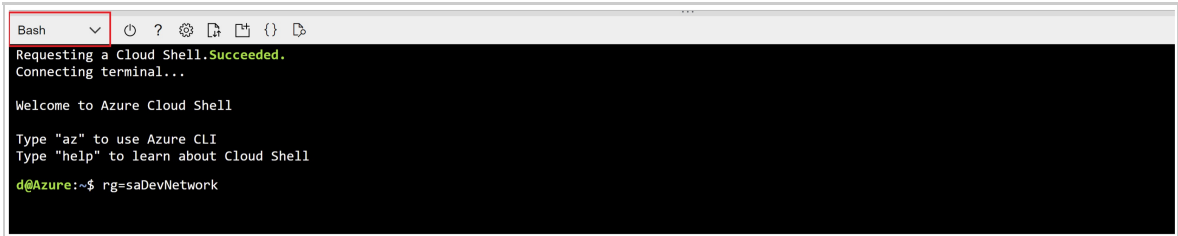
3. **Make sure** you are running your cloud shell terminal in Bash mode.
4. Enter in the following command to get to the proper resource group:

Bash

Copy

```
rg="<your-rg-here>"
```

Example below:



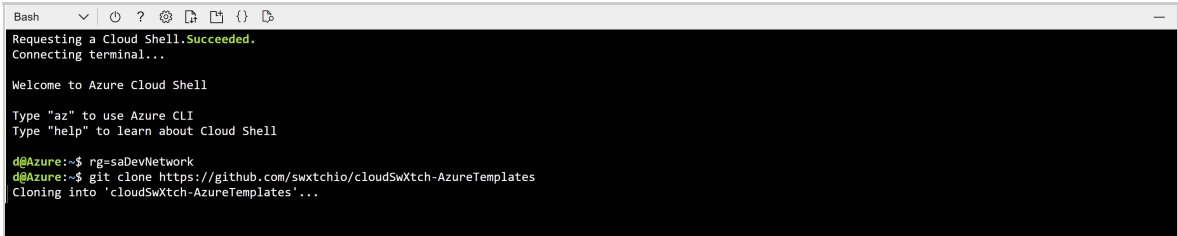
5. Enter in the following command to clone the "cloudSwXtch-AzureTemplates":

Bash

Copy

```
git clone https://github.com/swxtchio/cloudSwXtch-AzureTemplates
```

Example below:



6. **Change** directory (cd) to "cloudSwXtch-AzureTemplates".

Bash

Copy

```
cd cloudSwXtch-AzureTemplates
```

If desired, use the "ls" command to see what is in the directory. Example below:

```
Bash
Requesting a Cloud Shell.Succeeded.
Connecting terminal...

fernando [ ~ ]$ cd cloudSwXtch-AzureTemplates/
fernando [ ~/cloudSwXtch-AzureTemplates ]$ ls
LICENSE  MPTemplateUI.json  MPTemplateVM.json  README.md
fernando [ ~/cloudSwXtch-AzureTemplates ]$
```

7. Create "cloudSwXtch-from-mp-image" using the following command:

Bash	Copy
<pre>az ts create -n cloudSwXtch-from-mp-image -g \$rg -v 1 -f MPTemplateVM.json --ui-form-definition MPTemplateUI.json</pre>	

a. The output should look like the below screenshot:

```
Bash
{
  "validationMessage": "Enter full resource id. Must not be empty.",
},
"defaultValue": "",
"label": "Proximity Placement Group Resource ID",
"name": "ppgTextBox",
"toolTip": "WARNING: an invalid 'Proximity Placement Group' Resource ID will cause the deployment to fail",
"type": "Microsoft.Common.TextBox",
"visible": "[steps('network').AdvancedSection.ppgEnable]"
}
],
"label": "Advanced Networking (optional)",
"name": "AdvancedSection",
"type": "Microsoft.Common.Section"
}
],
"label": "Network",
"name": "network"
}
],
"title": "cloudSwXtch instantiation template"
}
}
}
}
eme@Azure:~/cloudSwXtch-AzureTemplates$
```

NEXT STEP: Azure cloudSwXtch Installation

After completing template creation and [validating subnets](#), continue on to the main [Azure cloudSwXtch installation guide](#).

Install cloudSwXtch on Azure

WHAT TO EXPECT

Installation of a cloudSwXtch instance consists of two parts: the cloudSwXtch and the xNIC software. The cloudSwXtch is instantiated once while the xNIC is installed on each VM that is part of the cloudSwXtch network.

In this section, users will learn how to install cloudSwXtch for their Azure environment through the template method.

Please note: This is the preferred method of installation. However, alternatively, you can do a manual install via the Marketplace. For more information on this method, please see the [Install cloudSwXtch via Market Place](#) guide.

NOTE:

Access to <https://services.swtch.io> should be enabled for marketplace installation of the cloudSwXtch. For closed environments, swXtch.io offers a BYOL model to allow installation and operation for highly secure deployments. Please contact support@swtch.io for more details.

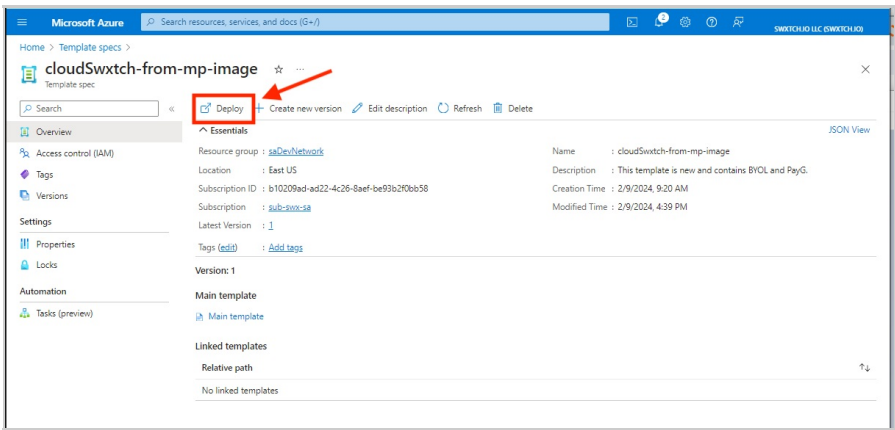
Deploying a cloudSwXtch instance

Prerequisites

Before starting, a user must do the following:

1. Review [cloudSwXtch System Requirements](#).
2. **Validate that there are two Subnets:** A virtual network must be created before creating a cloudSwXtch instance. This must contain two subnets, known as the ctrl- and data-subnet. In addition, the data subnet must have the "Network Acceleration" feature enabled.
3. **Create an Azure cloudSwXtch Template:** Creating a template will allow users to follow the easiest method for cloudSwXtch deployment detailed below.
4. **Make sure that your Azure subscription has the quota and access privileges to create the virtual machine instance used to run the cloudSwXtch.** Your instance will fail if you do not have the quota for the selected machine size.

1. Log into the **Azure Portal**.
2. Find the template by using the **Search resource, services, and docs (G+)** bar and enter **cloudSwxtch-from-mp-image** in the search. This will take to directly to the template.
3. Select the **template**.
4. Click **Deploy** to launch the template UI.



5. In the **cloudSwXtch commercial plan** area, click on the **Choose a cloudSwXtch plan** dropdown and **select a plan** (BYOL or PAYG). For more information on plans see: [cloudSwXtch System Requirements](#).

portal.azure.com/#create/Microsoft.Template

Home > Template specs > cloudswxtch-from-mp-image >

cloudSwXtch instantiation template

Template spec

New! Deployment Stacks let you manage the lifecycle of your deployments. Try it now →

Basics Network Review + create

Template

cloudSwXtch-from-mp-image (1)
3 resources

cloudSwXtch commercial plan

cloudSwXtch plans are billed at hourly rates in addition to any Azure infrastructure (VM) fees.
[Plan details and pricing information can be found here.](#)

Choose a cloudSwXtch plan: * byol

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * sub-sw-x-sa

Resource group * saDev/Network
[Create new](#)

Instance details

Region * (US) East US

Switch Name * cloudswxtch001

Switch Size * 1x Standard D8s v4
8 vcpus, 32 GB memory
[Change size](#)

Admin name * switchadmin

SSH public key source Generate new key pair

Key pair name * Name the SSH public key

Please enter the exact version. Version names are generally of the form '1.2.3'

Manual entry of software version * latest

Optional Resource Tags

Tags are name/value pairs that enable you to categorize resources and view consolidated billing by applying the same tag to multiple resources and resource groups. [Learn more about tags](#)

Note that if you create tags and then change resource settings on other tabs, your tags will be automatically updated.

Name	Value	Resource
CreatedBy	Jane Doe	2 selected

Previous Next Review + create

6. In the **Project Details** area, select a **Subscription**.

7. Pick (or create) an **Azure Resource Group**.

8. In the **Instance details** area, notice how the **region** is filled in from the **Azure Resource Group**.

9. Assign the **Virtual Machine** a name. This name must be unique in both the **resource group** and the **virtual network** in which the instance will exist. It also must meet the [requirements for a VM host name](#).

10. Select the **cloudSwXtch size**.

cloudSwXtch Size Explained

The default size is **1x Standard D8s V4**. A minimum of 8 cores is suggested for cloudSwXtch deployment.

NOTE:

Please be aware that the owner of the Azure Subscription in which the cloudSwXtch instance is created is responsible for all cloud resources used by the switch. These fees are to the cloud provider and do not include any fees to swtch.io for cloudSwXtch licensing.

11. Enter in an **Admin name**. This will default to **swtchadmin**, but can be modified.

12. Enter in a **SSH public key source**. The options are:

- **Generate new key pair.**
 - If selected, enter in **Key Pair Name**. This name must be unique among other key pairs in Azure.
- **Use existing key stored in Azure.**
 - If selected, choose a **stored key** from the drop-down menu.
- **Use existing public key.**
 - If selected, paste in a **SSH public Key** from Azure. Refer to <https://learn.microsoft.com/en-us/azure/virtual-machines/ssh-keys-portal> for how to get an existing public key.

13. Select the **software version**. The most common choice is **latest**, which will use the most recent software release for this instance. For more control, a specific release version can be entered, e.g. 3.0.0.

14. In the **Optional Resource Tags** area, optionally add **Tags**. Tags can be added to all Resources

15. Select **Next - Network**.

16. In the **Configure virtual networks** area, select a previously created virtual network.

WARNING

Due to an issue with Azure templates, **do not select the Create new option** for the network because the created network will not be accessible to you. **Always** select a previously created virtual network.

Network

The cloudSwXtch must be associated with a virtual network and the virtual network must have at least two subnets: one for control plane and one for data plane traffic. See "System Requirements" above for details.

17. In the **Configure virtual networks** area, select a **Control Subnet Name**.

18. Select a **Data Subnet Name**.

19. **OPTIONAL:** In the **Advanced Networking (optional)** section:

- Add a static IP Address.
- Specify a Proximity Placement Group.

The screenshot shows the 'cloudSwXtch instantiation template' form in the Azure portal. The form is titled 'cloudSwXtch instantiation template' and is a 'Template spec'. It has a breadcrumb path: 'Home > cloudSwXtch-from-mp-image >'. The form is divided into two main sections: 'Configure virtual networks' and 'Advanced Networking (optional)'. In the 'Configure virtual networks' section, there are three dropdown menus: 'Virtual network *' (set to '(new) pick-an-existing-vnet'), 'Control Subnet Name *' (set to '(new) ControlSubnet (10.0.0.0/29)'), and 'Data Subnet Name *' (set to '(new) DataSubnet (10.0.1.0/29)'). There is a 'Create new' link next to the 'Virtual network' dropdown. In the 'Advanced Networking (optional)' section, there is a checkbox labeled 'IP addresses are dynamic by default.' which is checked. Below this, there is a warning message: 'Static IPs are not validated here. An invalid static IP address will cause the deployment to fail.' There are two input fields for static IP addresses: 'Enter static IP for network interface on ControlSubnet *' and 'Enter static IP for network interface on DataSubnet *'. There is also a checkbox for 'Specify a Proximity Placement Group'. At the bottom of the form, there are three buttons: 'Review + create', '< Previous', and 'Next: Review + create >'.

20. Select **Review and Create**.

21. Review the **plan pricing**.

22. Read the **Terms & Conditions**.

23. Select **I agree** when ready.

The creation will take 1-3 minutes depending on Azure vagaries. When done, a cloudSwXtch instance shall exist within the selected Azure Resource Group. Your cloudSwXtch is now ready for use.

Post-Installation

- **IMPORTANT:** If this is a new install then each client that is expected to get traffic from the cloudSwXtch will need a xNIC installed. If this is a existing install then each client with an xNIC already installed will need to be upgraded. Please see [xNIC Installation](#).
- For Windows-related OS/servers, It's important to reboot the machine, once the installation is complete, in order to be able to execute cloudSwXtch tools properly from any client's user home directory.

24/7 Operations

If the services need to be up and running 24/7 swXtch.io suggests that redundant systems exist for which will be referred to as "Main" and "Backup". During an upgrade the Backup system should be upgraded, then the traffic should be routed to the Backup while the Main is upgraded.

Uninstalling cloudSwXtch

Delete the cloudSwXtch instance as you would any other virtual machine.

Install cloudSwXtch via Azure Marketplace

Installing cloudSwXtch via Template

The best method for deploying a cloudSwXtch on Azure is via a template. For more information on this method, please review the [Install cloudSwXtch on Azure](#) guide.

Prerequisites

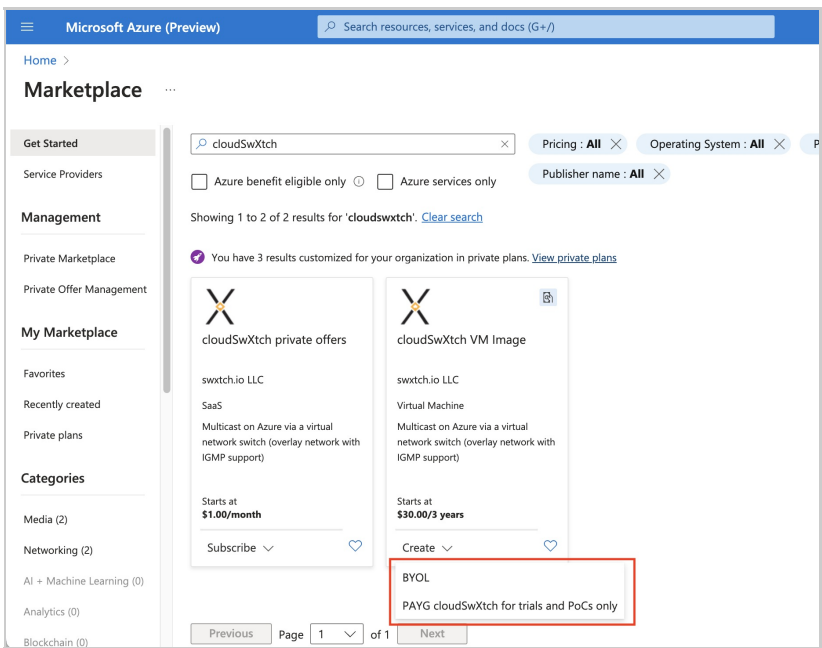
Before starting, ensure that you have [validated your subnets](#) on Azure. Return to this page after completing that preliminary step.

Creating a Virtual Machine

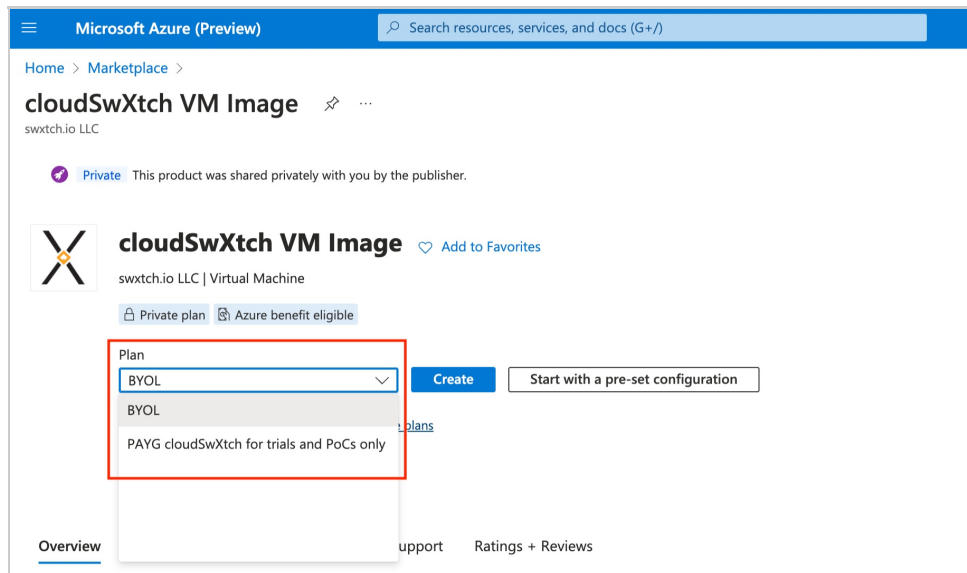
1. **Log in to the Azure Portal.** You will need the following permissions to create and manage virtual machines and to create Managed Applications.
 - **virtual-machine-contributor:** To create and manage virtual machines.
 - **managed-application-contributor-role:** To create Managed Applications.
2. **Select Marketplace.**
3. Search for **cloudswxtch**.
4. **Select a plan.** For more information, see: [cloudSwXtch System Requirements](#).
5. Click on the **cloudSwXtch VM Image** drop down menu to select a plan. **Please note:** A BYOL instance will require users to obtain a license from swXtch.io. For more information, see [here](#).

cloudSwXtch BYOL

It is recommended that users select **cloudSwXtch BYOL**, which allows for more customizability including expanded bandwidth, increased endpoint limit and additional licensable features. Users will need to request a license from support@swxtch.io. For more information, please see [cloudSwXtch System Requirements](#).



The **Create a virtual machine** will open with the selected plan. If the plan was not selected in the previous screen, then the following screen will open to choose a plan.



6. Select either **Create** or **Start with a pre-set configuration**.

NOTE

swtch.io is just using the standard Azure Marketplace VM from Image method, this document will not go over all the tabs and fields in the tabs as they are not cloudSwXtch specific. Some things of note in the Azure Marketplace VM image creation are as follows:

- The **Start with a pre-set configuration** vs **Create** will eventually lead to the same UI where there are many tabs to enter data. However, the **Start with a pre-set configuration** will fill in certain fields based on the user's selections. For example, in the **Basics** tab it will fill in **Boot diagnostics**, **Availability options**, and **Size**. In addition, the **Disks** tab will fill in the OS disk type.
- **REMINDER:** This Market Place method will only create one NIC. The second required NIC will need to be added after creation.

7. Follow the tabs and make appropriate selections – there are a number of fields that have to be filled in to create a cloudSwXtch instance.
8. In the **Basics** tab, select a **Subscription**.
9. Choose (or create) a **Resource Group**.
10. Assign the **Virtual Machine Name**. This name must be unique.
11. Select a **Region**.
12. Confirm the **Image** is the correct one you selected when choosing a plan: **cloudSwXtch BYOL** or **cloudSwXtch PAYG**.

13. Select the **Software Version**. The most common choice is **latest**, which will use the most recent software release for this instance. For more control, a specific release version can be entered.

14. Continue on the **Networking** Tab.

Networking Tab

The cloudSwXtch instance must be associated with a virtual network and the virtual network must have at least two subnets: one for control plane and one for data plane traffic. This user interface only allows attachment of one subnet. Below steps will describe how to add a second subnet after creation. See "System Requirements" above for details.

15. Check **Delete public IP and NIC when VM is deleted**.

16. **OPTIONAL**: Change values on other tabs.

17. Select **Review and Create**.

18. Carefully review the **plan pricing**.

19. Read the **Terms & Conditions**.

20. Select **I agree** when ready. **Note**: The creation will take 2-3 minutes depending on Azure varieties.

Creating the Second Subnet *REQUIRED*

1. Navigate to the newly created VM by selecting the **Go to Resource** button.
2. Click **Stop** at the top of the toolbar.
3. Select **Yes** when prompted.
4. Click **Networking** on the left hand side under settings. Alternatively, you can select **Networking** in the main Properties page.

Home > CreateVm-swxtchiolc1614108926893.cloudswtch-vm--20220930112431 | Overview >

vm-swxtch300

Virtual machine

Search

Connect Start Restart **Stop** Capture Delete Refresh Open in mobile CU / PS

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Settings

Networking

Connect

Disks

Size

Microsoft Defender for Cloud

Advisor recommendations

Extensions + applications

Continuous delivery

Availability + scaling

Configuration

Identity

Properties

Locks

Operations

Beacon

Auto-shutdown

Backup

Disaster recovery

Updates

Inventory

Essentials

Resource group (move) : [test-donna-300-rg](#)

Status : Stopped (deallocated)

Location : East US

Subscription (move) : [sub-swxt.sa](#)

Subscription ID : b10209ad-ad22-4c26-8aef-be93b270bb58

Tags (edit) : CreatedBy: dsilver@swxtch.io

Operating system : Linux

Size : Standard D4 v4 (4 vcpus, 16 GiB memory)

Public IP address : [40.112.52.133](#)

Virtual network/subnet : [test-donna-300-vnet/ds-subnet-ctrl-300](#)

DNS name : [Not configured](#)

Properties Monitoring Capabilities (7) Recommendations Tutorials

Virtual machine

Computer name : dsd-vm-swxtch300

Health state : -

Operating system : Linux

Publisher : swxtchiolc1614108926893

Offer : cloudswtch-vm-001-preview

Plan : swtch-small-002

VM generation : V1

VM architecture : x64

Host group : None

Host : -

Proximity placement group : -

Colocation status : N/A

Capacity reservation group : -

Availability + scaling

Availability zone : -

Availability set : -

Scale Set : -

Networking

Public IP address : 40.112.52.133

Public address (IPv6) : -

Private IP address : 10.1.1.6

Private address (IPv6) : -

Virtual network/subnet : [test-donna-300-vnet/ds-subnet-ctrl-300](#)

DNS name : [Configure](#)

Size

Size : Standard D4 v4

vCPUs : 4

RAM : 16 GiB

Disk

OS disk : dsd-vm-swxtch300_OsDisk_1_ba053f9f18e54a10

Encryption at host : Disabled

Azure disk encryption : Not enabled

Ephemeral OS disk : N/A

Data disks : 0

Auto-shutdown

5. Select **Attach network interface**.

Microsoft Azure

Search resources, services, and documentation

Home > CreateVm-swxtchiolc1614108926893.cloudswtch-vm--20220930112431 | Overview > vm-swxtch300

vm-swxtch300 | Networking

Virtual machine

Search

Attach network interface Detach network interface

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Settings

Networking

vm-swxtch300727

IP configuration ⓘ

ipconfig1 (Primary)

Network Interface: vm-swxtch300727 Effective security rules Troubleshoot

Virtual network/subnet: [test-donna-300-vnet/ds-subnet-ctrl-300](#) NIC Public IP: [40.112.52.133](#)

Inbound port rules Outbound port rules Application security groups Load balancers

Network security group dsd-vm-swxtch300-nsg (attached to network interface)

6. Select a **Resource Group** under **Project Details**.

7. Enter in a **Name** under **Network Interface**.

8. Select a **Subnet**. **Please note:** You can optionally change other data.

9. Select **Create**.

Project details

Subscription sub-sw-x-sa

Resource group * test-300-rg [Create new](#)

Location (US) East US

Network interface

Name * vm-swxtch300-data

Virtual network test-300-vnet

Subnet * subnet-data-300 (10.1.2.0/24)

NIC network security group

☐ None

☒ Basic

☐ Advanced

Public inbound ports *

☒ None

☐ Allow selected ports

Select inbound ports

Select one or more ports

All traffic from the internet will be blocked by default. You will be able to change inbound port rules in the VM > Networking page.

Private IP address assignment

☒ Dynamic ☐ Static

[Create](#)

10. Refresh the screen after completing the form and the second subnet should be added in a second tab.

Enabling "Accelerated Networking" *REQUIRED*

The newly created Network Interface needs to be updated to enable **Accelerated Networking** to do this follow the steps below:

1. Select the **Network Interface**. In the example below, it is named **vm-swxtch-300-data**.
2. Click the blue link to the **Network Interface**.

Network interface **vm-swxtch300-data** [Effective security rules](#) [Troubleshoot VM connection issues](#) [Topology](#)

Virtual network/subnet: test-donna-300-vnet/ds-subnet-data-300 NIC Public IP: - NIC Private IP: **10.1.2.6** Accelerated networking: **Disabled**

[Inbound port rules](#) [Outbound port rules](#) [Application security groups](#) [Load balancing](#)

Network security group basicNsgrdsd-vm-swxtch300-data (attached to network interface: dsd-vm-swxtch300-data) [Add inbound port rule](#)

Impacts 0 subnets, 1 network interfaces

Priority	Name	Port	Protocol	Source	Destination	Action
65000	AllowVnetInBound	Any	Any	VirtualNetwork	VirtualNetwork	Allow
65001	AllowAzureLoadBalancerInBound	Any	Any	AzureLoadBalancer	Any	Allow
65500	DenyAllInBound	Any	Any	Any	Any	Deny

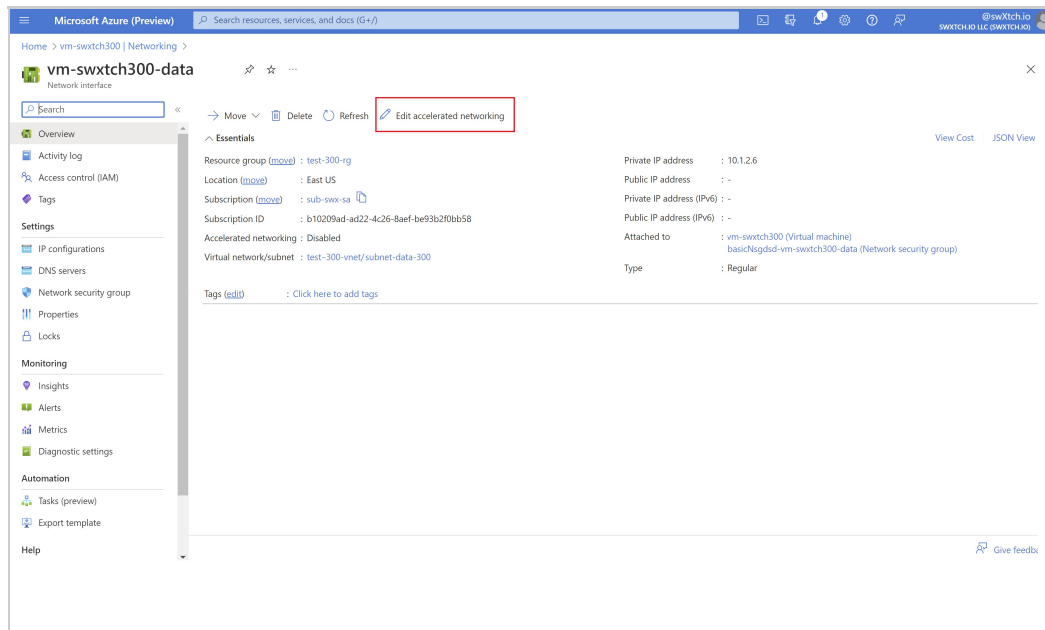
Need help?

[Understand Azure load balancing](#)

[Quickstart: Create a public load balancer to load balance Virtual Machines](#)

[Quickstart: Direct web traffic with Azure Application Gateway](#)

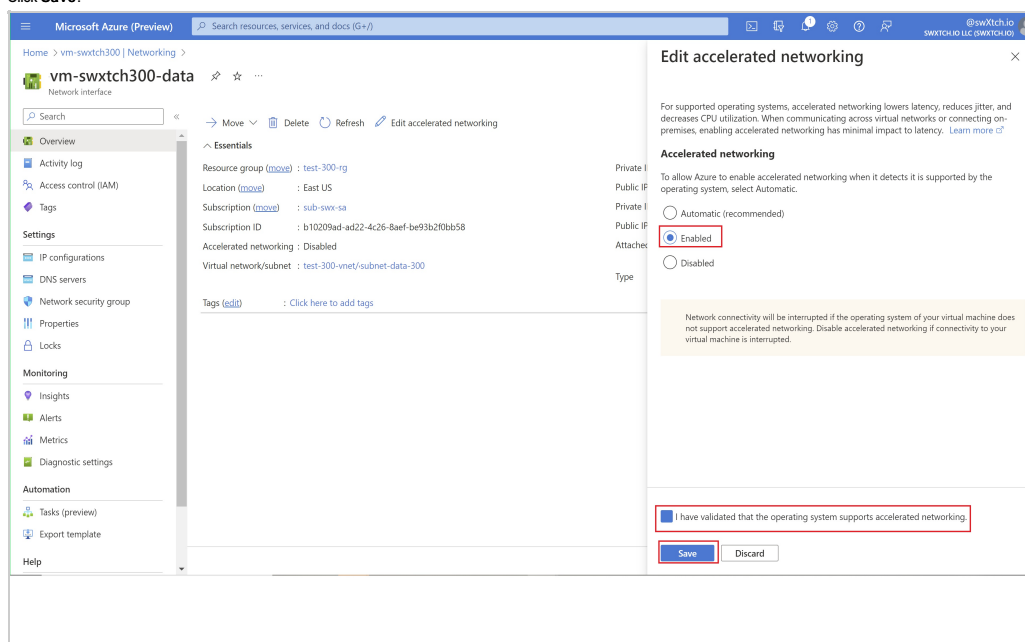
3. Click **Edit accelerated networking**.



4. Select **Enable**.

5. Select **I have validated that the operating system supports accelerated networking**.

6. Click **Save**.



7. Start the VM for use.

Upgrading your cloudSwXtch Instance

After deployment, it is recommended to update your cloudSwXtch instance to latest. Please refer to the [Upgrading cloudSwXtch](#) article for more information.

Required Step for BYOL: Request a License from swXtch.io

Users deploying a BYOL instance of cloudSwXtch will need to contact swXtch.io for a license file. For more information, see [How to License a cloudSwXtch](#).

Important

If this is a new install then each client that is expected to get traffic from the cloudSwXtch or send to the cloudSwXtch will need a xNIC installed. If this is a existing install then each client with an xNIC already installed will need to be upgraded. Please see [xNIC Installation](#).

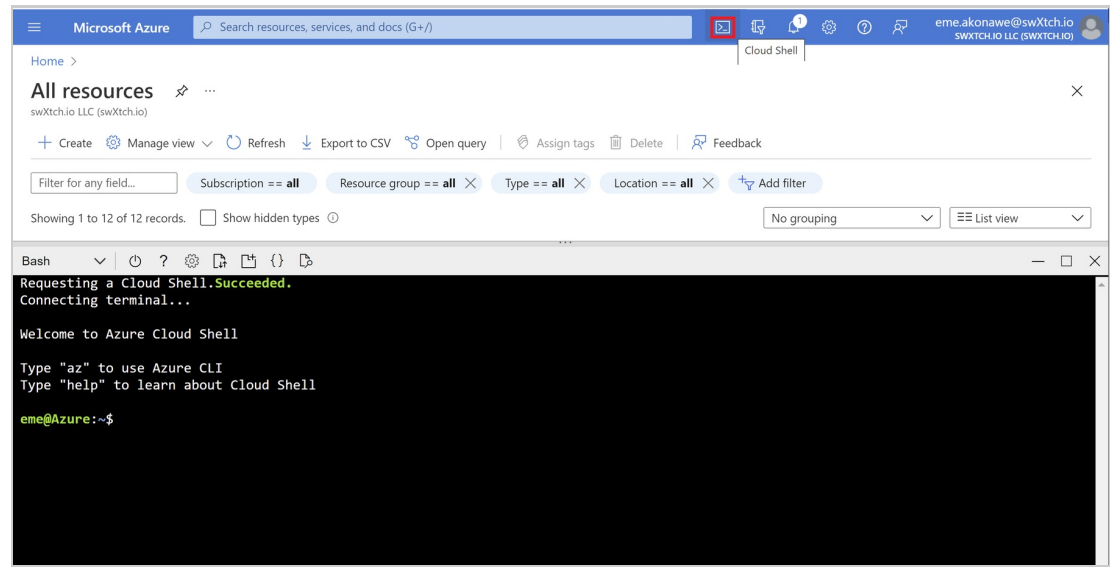
Deploy cloudSwXtch with Terraform on Azure

WHAT TO EXPECT

By default, the terraform script will spin up a "small" cloudSwXtch. You can make edits to the [Azure/terraform/terraform.tfvars](#) file to declare a different cloudSwXtch size. There is also an option to delegate static ip addresses on your cloudSwXtch. Further details on how to do this can be found at the end of this article.

Deploying cloudSwXtch with Terraform on Azure

1. Sign-in to your Azure portal under the subscription where you want to deploy the cloudSwXtch.
2. Open the Azure Cloud Shell interface and select the Bash environment as shown.



3. Clone the example repository from [GitHub](#).

You can do this either **via SSH** (requires setting up your SSH authentication with GitHub):

Bash	Copy
<pre>\$ git clone git@github.com:swxtchio/cloudSwXtch-support.git</pre>	

or **via HTTPS**:

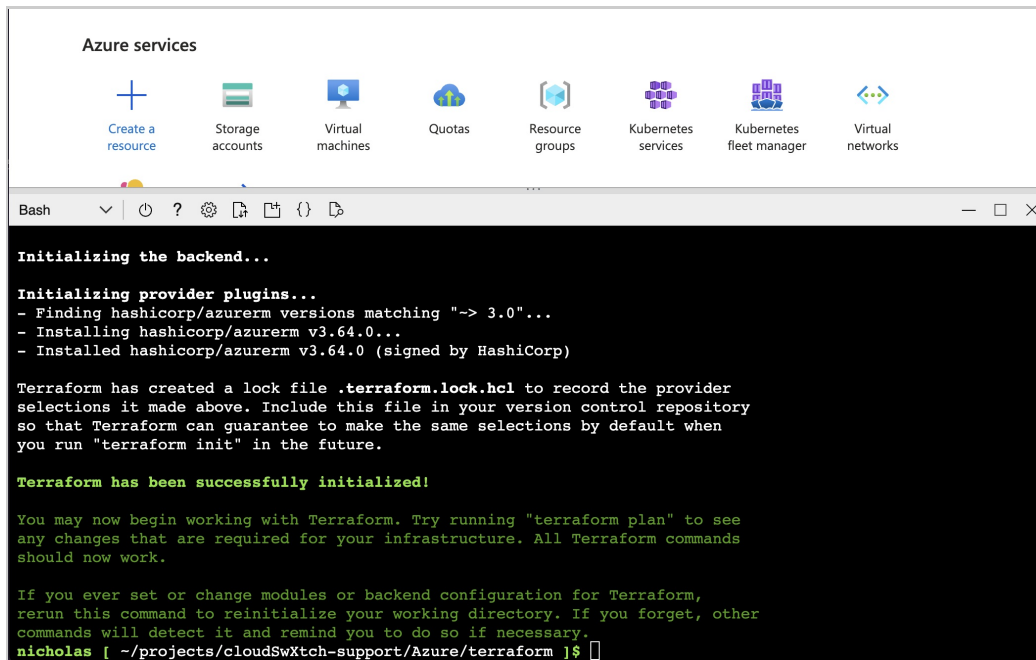
Bash	Copy
<pre>\$ git clone https://github.com/swxtchio/cloudSwXtch-support.git</pre>	

4. Update the values in the [Azure/terraform/terraform.tfvars](#) file to match your existing azure resources such as: resource group, virtual network, subnets, etc.

The format of the key file that the scripts can process is the ssh-rsa type. The content of the file should start with "ssh-rsa AAAAB3..."

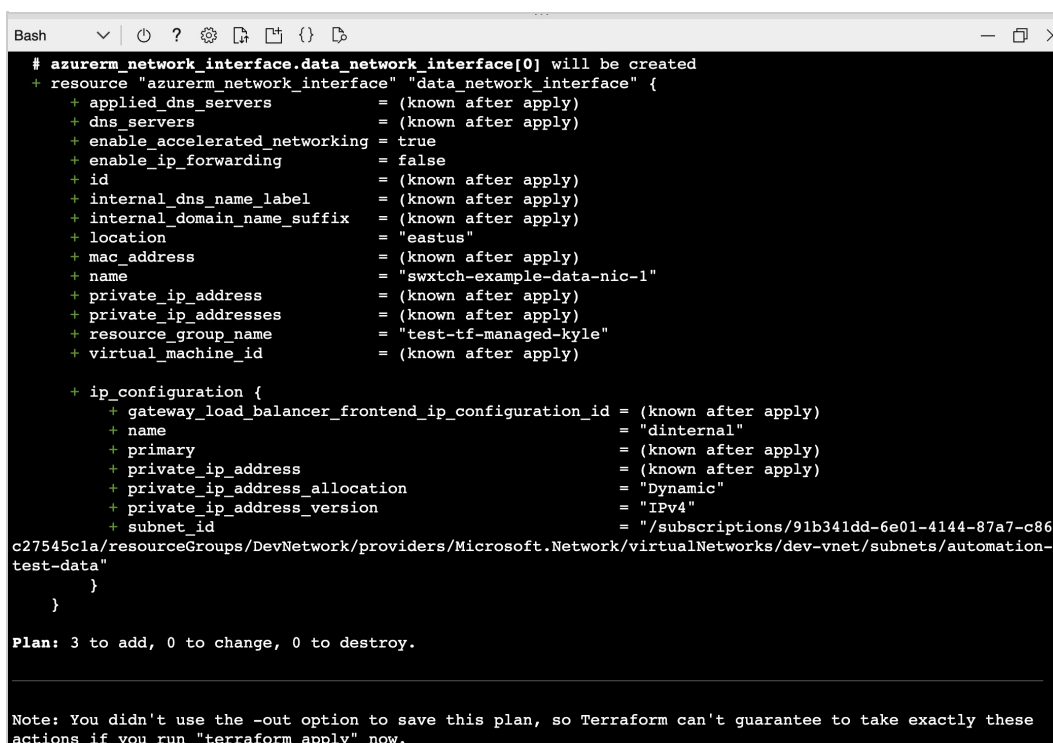
5. In the Cloud Shell terminal, cd into the [Azure/terraform](#) directory and initialize the terraform environment:

Bash	Copy
<pre>\$ cd Azure/terraform \$ terraform init</pre>	



6. Now that Terraform has been initialized, run this command to evaluate the config and confirm the desired output which will be shown:

Bash	Copy
\$ terraform plan	



Since you are using all pre-existing resources to deploy your cloudSwXtch, there should only be 3 resources added - 1 cloudSwXtch, and 2 NICs - as can be seen at the bottom of the screenshot, "Plan: 3 to add, 0 to change, 0 to destroy."

7. Run the Terraform apply command (followed by "yes" when prompted) to approve the action.

Bash	Copy
Terraform apply yes	

8. Once the resources have applied successfully you can view the resources created from your Azure portal as confirmation of a successful deployment.

STATIC IPs

If you'd like to deploy a cloudSwXtch using Static IPs, then you just need to make some small changes to the `azure_deployswxtch.tf` & `terraform.tfvars` files.

1. Un-comment the Parameter `private_ip_address` in the `azure_deployswxtch.tf` code file for both your `data_network_interface` & `control_network_interface` resources.

```

source "azurerm_network_interface" "data_network_interface" {
  count          = var.counter
  name           = "${var.data_nic}-${count.index +1}"
  location       = data.azurerm_resource_group.resource_group.location
  resource_group_name = data.azurerm_resource_group.resource_group.name
  enable_accelerated_networking = true

  ip_configuration {
    name                = "dinternal"
    subnet_id           = data.azurerm_subnet.datasubnet.id
    private_ip_address_allocation = "Static"
    private_ip_address   = var.datanic_staticip
  }
}

```

```

39 resource "azurerm_network_interface" "control_network_interface" {
40   count          = var.counter
41   name           = "${var.control_nic}-${count.index +1}"
42   location       = data.azurerm_resource_group.resource_group.location
43   resource_group_name = data.azurerm_resource_group.resource_group.name
44
45   ip_configuration {
46     name                = "cinternal"
47     subnet_id           = data.azurerm_subnet.ctrlsubnet.id
48     private_ip_address_allocation = "Static"
49     private_ip_address   = var.controlnic_staticip
50   }
51 }

```

2. Set the parameter `private_ip_address_allocation` to "Static".

Your 2 lines of code should look like below for both network interface resources:

Bash

Copy

```

private_ip_address_allocation = "Static"
private_ip_address = var.datanic_staticip

```

Your `terraform.tfvars` file will have variables defined for your control and data NIC StaticIP definitions. You can update those values based on your subnet setup.

Note: This static IP address allocation will only work for `swxtch_count` of 1.

Install cloudSwXtch for an Air-Gapped Environment

WHAT TO EXPECT

In this article, you will learn how to install a cloudSwXtch in an Air-Gapped (Closed Network) environment for Azure. For standard Azure installation instructions, please see the [cloudSwXtch on Azure](#) article.

Before You Start

Review VM Requirements for a cloudSwXtch Instance in [cloudSwXtch System Requirements](#).

VM Image Creation

The cloudSwXtch software is delivered as a **Virtual Machine Disk Image**. This Image file can be added to an Azure **Image Gallery**. Images in an Image Gallery can be used to create Virtual Machines.

To assist with creation of VMs from images in a gallery, swXtch.io provides instructions on how to accomplish the following:

- 1. [Get the VM Disk Image](#)
- 2. [Upload the VM Image into an Azure Storage Account](#)
- 3. [Create a VM Image from the Disk Image](#)
- 4. [Create cloudSwXtch from VM Image](#)
- 5. [License the cloudSwXtch](#)

Complete all steps to successfully install cloudSwXtch for an Air-Gapped environment.

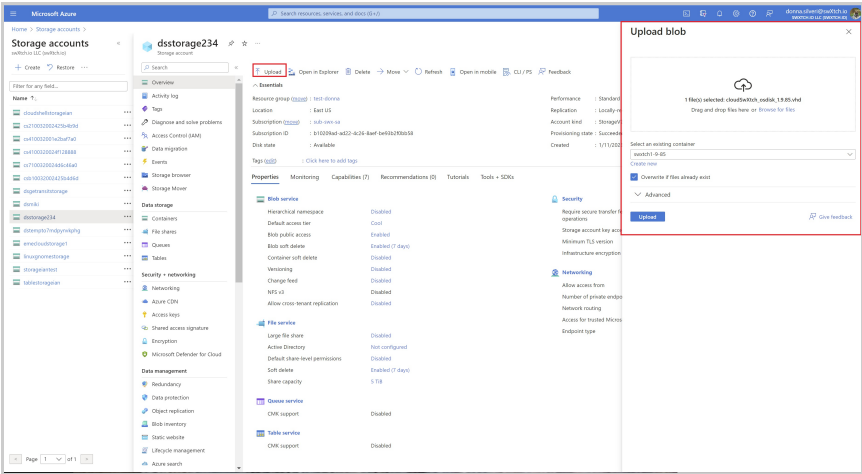
STEP ONE: Get the VM Disk Image

Log onto an environment that has access to the internet and download the following file (~30GB):

Plaintext	Copy
https://swxtchpublic.blob.core.windows.net/3hwgfe98hfglsrdfh4/cloudSwXtch_osdisk_1.9.85.vhd	

STEP TWO: Upload the VM Disk Image into an Azure Storage Account

- 1. Place the file onto a machine with access to the **Azure Air Gapped Environment**.
- 2. Upload the files into an Azure storage account in the **secure Azure Environment**.
 - a. Log into the **Azure Portal**
 - b. Navigate to **Storage Accounts**.
 - c. Select the desired storage account.
 - d. Select the desired Container or **create a new one**.
 - e. Select **Upload** and select the VM Disk Image file you copied to the local PC.

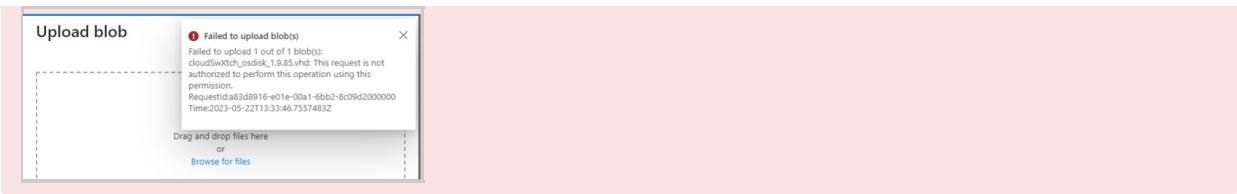


- f. Start the upload and wait for it to complete.

This may take some time to upload the file (up to an hour). When completed, the file should show with a green checkbox.

Failed to Upload Blob(s) Message

If you receive a "Failed to Upload Blob(s)" message when uploading the file in the Storage Account, select **Configuration** and validate the **Allow storage key access** is enabled.



STEP THREE: Create a VM Image from the Disk Image

Once we have a disk image in storage, we can use it to create a VM image. A VM image is a *description* of a VM. The real VM will be created later. The VM Image only needs to be created once. Any number of VMs can be instantiated from a single VM image.

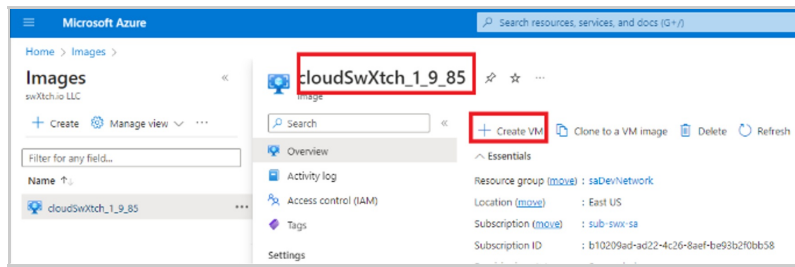
1. In the Azure Portal, **Search** for and **select Images**.
2. **Select Create**.
3. Select the appropriate **Resource Group**.
4. Give the VM Image a name. The cloudSwXtch instance will be created later with a different name. Pick a name with the cloudSwXtch software version in it as you may end up with multiple images after some time.
5. Ensure that the region is the same for the storage account holding the disk image.
6. Select **Linux** as the OS type
7. Select **Gen 1**.
8. Click **Browse** on the **Storage blob**.
 - a. In the new panel, navigate to the storage account and container holding the disk image.
 - b. Select the file that was previously uploaded.
9. For **Account Type**, select **Standard SSD**. See the example of the screen filled out completely.

10. If tags are desired, then select **Tags** and enter the required tags.
11. The other fields can be left as default.
12. Select **Review and create**.
13. When validation passes, select **Create**. When it is complete, click **Go to Resource** to see the image.

STEP FOUR: Create cloudSwXtch from VM Image

Now that we have a cloudSwXtch VM Image, we can use it to instantiate a cloudSwXtch.

1. Navigate to **Images**.
2. Select the image with the cloudSwXtch version you require.
3. Select **Create VM**.



4. Fill out the Create Virtual machine form like below:

Microsoft Azure

Home > Microsoft Image-20230603122553 | Overview > cloudSwXtch_1_9_85_image >

Create a virtual machine

Basics Disks Networking Management Monitoring Advanced Tags Review & create

Create a virtual machine that runs Linux or Windows. Select an image from Azure marketplace or use your own customized image. Complete the Basics tab then Review & create to provision a virtual machine with default parameters or review each tab for full customization. [Learn more](#) >

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription *

Resource group * [Create new](#)

Instance details

Virtual machine name *

Region

Availability options

Security type

Image * [See all images](#) | [Configure VM generation](#)

VM architecture ☐ Arm64 ☒ x64

Arm64 is not supported with the selected image.

Run with Azure Spot discount ☐

Size * [See all sizes](#)

Administrator account

Authentication type ☒ SSH public key ☐ Password

Azure now automatically generates an SSH key pair for you and allows you to store it for future use. It is a fast, simple, and secure way to connect to your virtual machine.

Username *

SSH public key source

Stored keys

Inbound port rules

Select which virtual machine network ports are accessible from the public internet. You can specify more limited or granular network access on the Networking tab.

Public inbound ports * ☒ None ☐ Allow selected ports

Select inbound ports

All traffic from the internet will be blocked by default. You will be able to change inbound port rules in the VM > Networking page.

Licensing

License type *

If you are using a Red Hat or SLES image, you may be eligible for the Azure Hybrid Benefit and can save money on the license costs. [Learn more](#) > about this benefit and how to enable it using Azure CLI for custom images from snapshots and Azure compute gallery.

[Review & create](#) < Previous Next > [Disks](#)

- Set the **subscription** and **Resource Group** for where you want the cloudSwXtch instance to be located.
- Name the Virtual Machine with a valid host name.
- Select appropriate machine size. For recommendations based on features, endpoints, and bandwidth needs, read the [Quotas](#) article.
- Use SSH for the authentication type. Enter your **SSH public key source**. Refer to [ssh-keys-portal](#) for details.
- Set the **Licensing Type** to **Other**.
- Navigate to the **Networking** tab and fill out the form like below:

i. Select the appropriate **Virtual Network**.

ii. Select the appropriate control subnet.

g. Navigate to other tabs as desired and enter in information as preferred. For example, some installations expect **Tags** to be entered.

h. Select **Review + Create**.

i. When validation passes, select **Create**.

5. When the deployment is complete, select **Go to Resource**.

a. Select **Stop** to stop the VM.

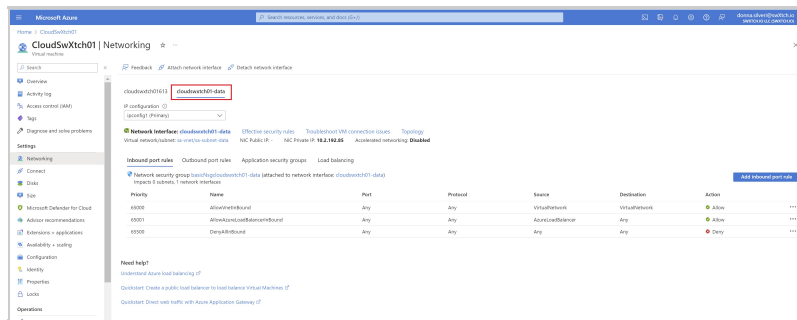
6. Navigate to **Networking**.

7. Select **Attach network Interface**.

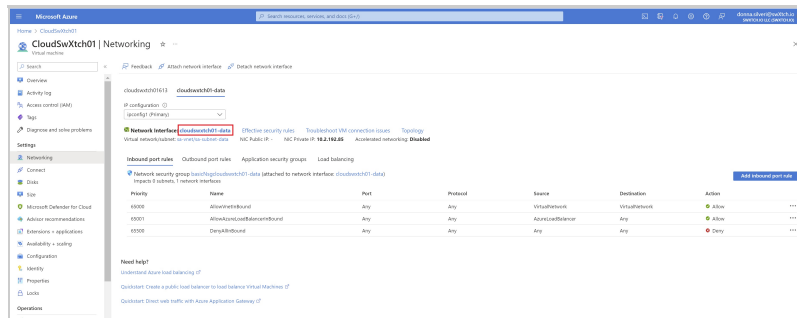
8. Select **Create and attach Network** and enter in data into the form to add a new NIC like shown.

9. Select **Create**.

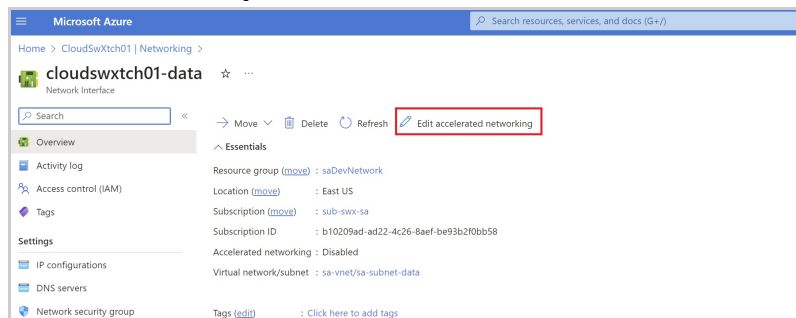
10. When it is done, refresh the screen. **There should now be a control and data interface.**



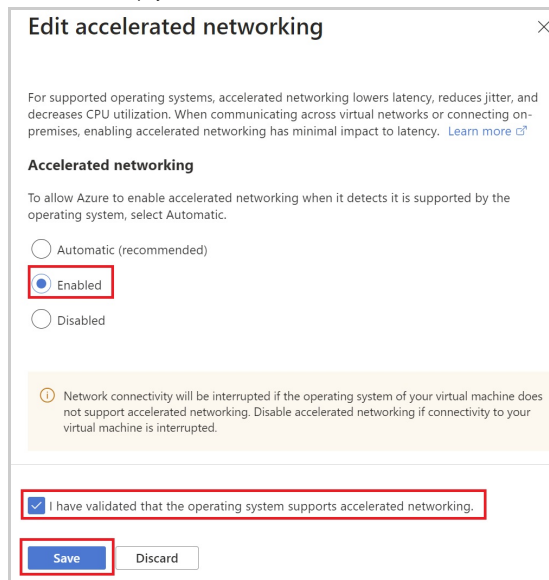
11. Select the data Network Interface.



a. Select Edit accelerated Networking.



b. A new window will display.

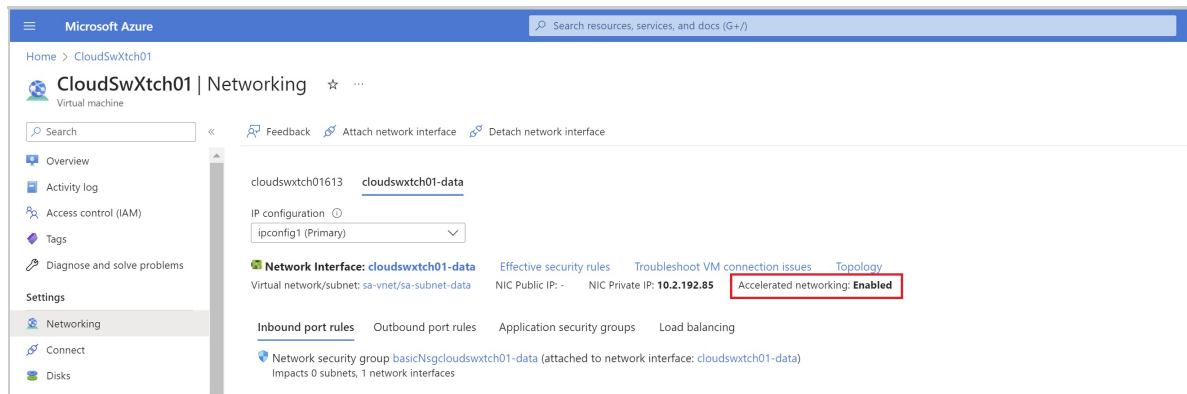


c. Select Enabled.

d. Check the agreement.

e. Select Save.

12. Refresh page and navigate back to Networking data tab to validate that Accelerated networking is Enabled.



13. Start the newly created cloudSwXtch VM.

STEP FIVE: License the cloudSwXtch

1. Log onto the newly created VM.

2. Run this command:

Text

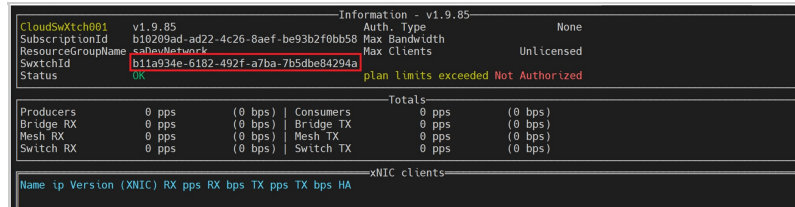


Bash

Copy

```
sudo /swxtch/swxtch-top dashboard --swxtch localhost
```

3. The swXtch-top dashboard will display.



4. Copy the "SwxtchId" and email it to support@swxtch.io requesting a license.

5. When you receive the license file, upload it onto the cloudSwXtch VM.

6. Move the license.json file to the /swxtch directory using the following command replacing user with the appropriate value:

Text



Bash

Copy

```
sudo mv /home/<user>/license.json /swxtch
```

7. Reboot the cloudSwXtch and run swxtch-top again or journal to check the license took place:

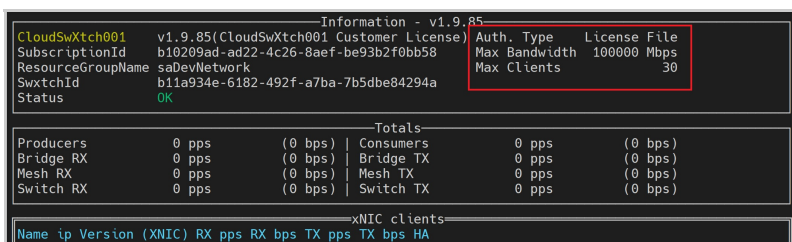
Text



Bash

Copy

```
sudo journalctl -u swxtch-ctrl.service -f -n 500
```



The cloudSwXtch is ready for use. IMPORTANT: Each client that is expected to get traffic from the cloudSwXtch will need an xNIC installed. See [Installing xNIC](#) for next steps in preparing clients (producers and consumers of Multicast).

Prerequisite: Installing Dependencies for Air-Gapped Clients

Before installing the xNIC, users will need to install the necessary packages on their air-gapped clients. Failure to complete this prerequisite will result in an unsuccessful xNIC deployment. To learn more, see [How to Install xNIC Dependencies in an Air-Gapped Environment](#).

cloudSwXtch on GCP

WHAT TO EXPECT

In this article, users will find links to articles on deploying a cloudSwXtch on Google Cloud Platform (GCP).

Currently, there is two ways of deploying a cloudSwXtch on GCP:

- [From the Google Cloud Marketplace](#)
 - **Note:** This method will require users to contact [swXtch.io](#) for a license.
- [Cloud agnostic cloudSwXtch VM Install](#)
 - **Note:** This method will require the user to already have a Virtual Machine installed with Ubuntu 20.04 that adheres to all the [cloudSwXtch System Requirements](#).

Install cloudSwXtch via GCP Marketplace

WHAT TO EXPECT

In this article, users will learn how to deploy a cloudSwXtch instance via the Google Cloud Platform (GCP) Marketplace.

- [Prerequisites](#)
- [Step One: Navigate to cloudSwXtch in the GCP Marketplace](#)
- [Step Two: Configure cloudSwXtch deployment](#)
- [Step Three: Add SSH Key\(s\)](#)
- [Required Step for BYOL: Contact swXtch.io for a License](#)

Prerequisites

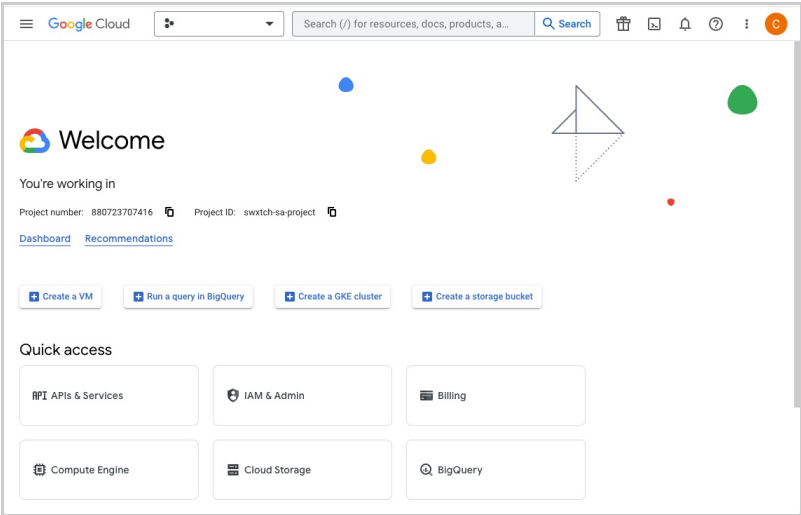
A user needs the following to deploy a cloudSwXtch via the GCP Marketplace:

- An existing **Deployment Service Account** established in their Google Cloud Console project. Creating a new account is further detailed in Step Two.
- **Two (2) VPCs available** -- one for the Control NIC and another for Data. All cloudSwXtch installations require 2 NICs. Please review GCP documentation on [how to create and modify VPCs](#).
 - **Note:** GCP does not allow you to have 2 NICs on the same VPC. It will return an error message. The Control and Data NIC must be assigned their own VPC.

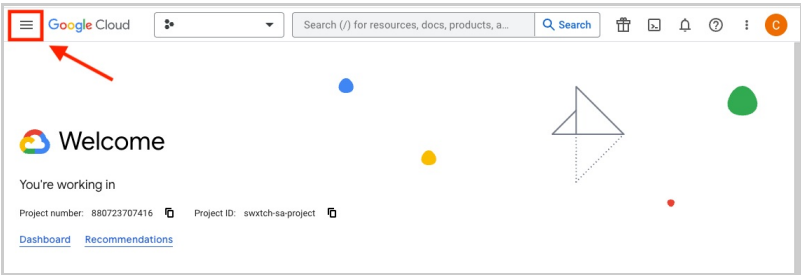
Please review [cloudSwXtch System Requirements](#) for additional prerequisites.

Step One: Navigate to cloudSwXtch in the GCP Marketplace

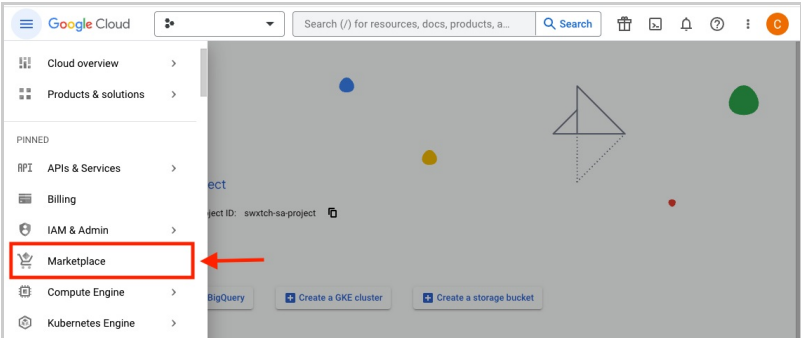
1. Log into the **Google Cloud Console**.



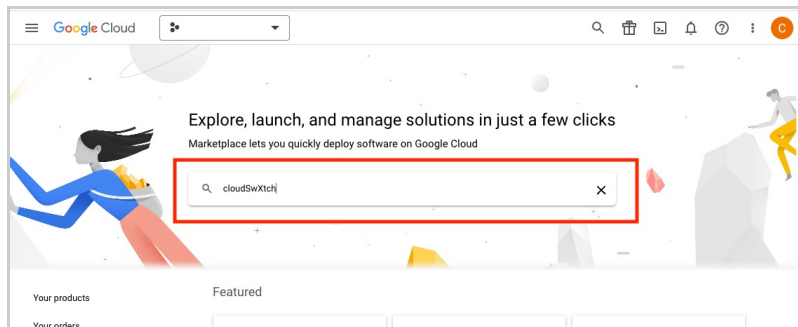
2. Navigate to the **Google Cloud Marketplace** using the **Navigation** menu at the top left corner.



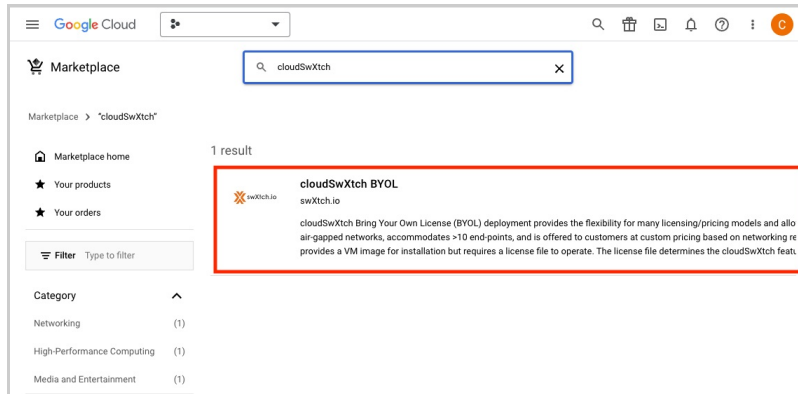
3. Select **Marketplace**.



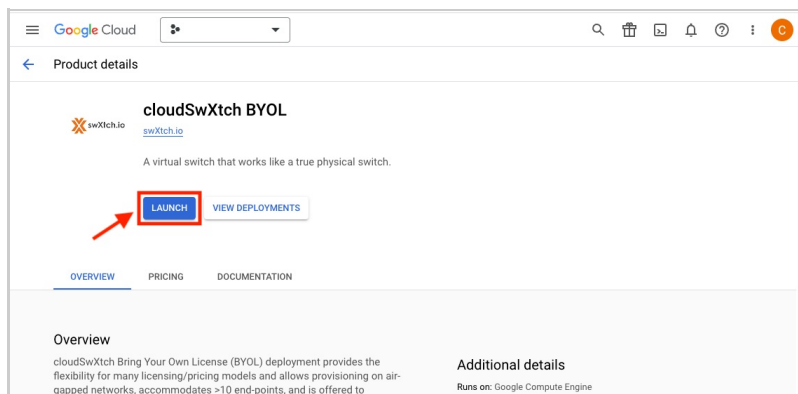
4. Search for **cloudSwXtch**.



5. Select the product, **cloudSwXtch BYOL**.



6. Click **Launch** to open the deployment configuration page.

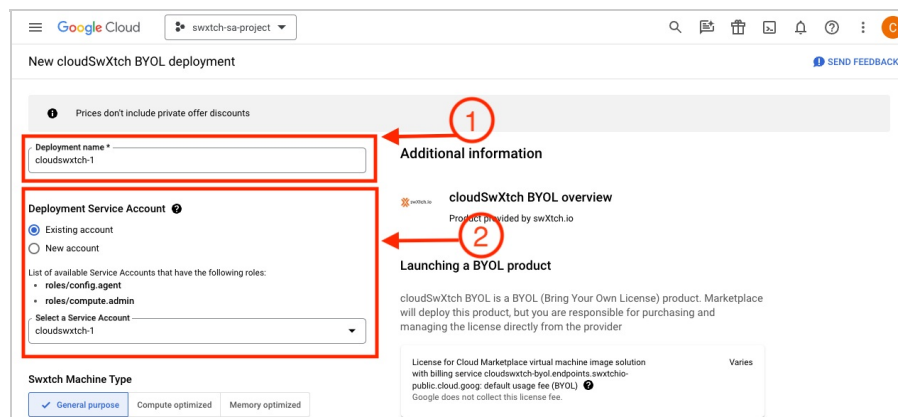


Enabling APIs

After hitting launch, a new window might open asking you to enable Google APIs. You must enable the suggested APIs to continue.

Step Two: Configure cloudSwXtch Deployment

1. Enter a **Deployment name** for your cloudSwXtch.
2. Select an **Existing account** under Deployment Service Account.



- a. If you do not have a **Deployment Service Account**, select **New Account**. You will need permissions from your Project IAM Admin (or someone with the `resourcemanager.projects.setIamPolicy` permissions) to allow you to create a new Deployment Service Account.
- b. Enter the same name used for your cloudSwXtch Deployment for your New Account Name and ID. The names must match and only use lowercase letters and numbers.
- c. **The account will be created after you deploy your cloudSwXtch.** Once an account is created, users without permissions can use it as an Existing account option.

Prices don't include private offer discounts

Deployment name *
cloudswtch-2

Deployment Service Account

☐ Existing account

☒ New account

Create a new Service Account

This will create a new Service Account with the following roles:

- roles/config.agent
- roles/compute.admin

Name *
cloudswtch-2

ID *
cloudswtch-2

Description

Swtxch Machine Type

☒ General purpose ☐ Compute optimized ☐ Memory optimized

Additional information

cloudSwXtch BYOL overview

Product provided by swXtch.io

Launching a BYOL product

cloudSwXtch BYOL is a BYOL (Bring Your Own License) product. Marketplace will deploy this product, but you are responsible for purchasing and managing the license directly from the provider

License for Cloud Marketplace virtual machine image solution with billing service cloudswtch-byol endpoints, swtxchio-public-cloud-goop: default usage fee (BYOL)

Google does not collect this license fee.

Price estimates based on 30-day, 24hrs per day usage of the listed resources in the selected region. The Monthly Infrastructure Fee is not included in the estimates and varies depending on all Google Cloud IaaS resources actually created or consumed by this product (or the fees charged for such consumption). The provider of this product may be able to provide a more accurate estimate of monthly GCP IaaS consumption.

- Under **Swtxch Machine Type**, confirm that **N2** is selected under **Series**.
- Confirm your sizing under **Machine Type**. The default is set to **n2-standard-16**, which is 16 core. A cloudSwXtch must have a minimum of 8 cores. For cloudSwXtch Sizing guidelines, see [cloudSwXtch System Requirements](#).
- Confirm your desired **Zone**.

Prices estimates based on 30-day, 24hrs per day usage of the listed resources in the selected region. The Monthly Infrastructure Fee is not included in the estimates and varies depending on all Google Cloud IaaS resources actually created or consumed by this product (or the fees charged for such consumption). The provider of this product may be able to provide a more accurate estimate of monthly GCP IaaS consumption.

Swtxch Machine Type

☒ General purpose ☐ Compute optimized ☐ Memory optimized

Machine types for common workloads, optimized for cost and flexibility

Series
N2

Powered by Intel Cascade Lake and Ice Lake CPU platforms

Machine type
n2-standard-16 (16 vCPU, 8 core, 64 GB memory)

vCPU
16

Memory
64 GB

Zone
us-central1-a

- Use the dropdown arrow under **Control network interface** to open the configuration panel. If your default subnet is already selected and you do not wish to set a public IP, continue you on Step 10.

Control network interface

Definitions for the control network interface.

Network interfaces

default default (10.128.0.0/20)

ADD A NETWORK INTERFACE

- Select a **Network** and **Subnetwork**. This subnet will be used for your control plane communications.
 - Optional:** Users can select **Ephemeral** under **External IP** if they wish for their Control NIC to be assigned a randomized public IP address.

Control network interface

Definitions for the control network interface.

Network interfaces

Edit network interface

Network
default

Subnetwork
default

External IP


None

Ephemeral

- Click **Done** when you are happy with your selections.
- Use the dropdown arrow under **Data network interface** to open the configuration panel.

Data network interface
Definitions for the data network interface.

Network interfaces



vpc02 dpdk-uscentral1 (10.28.0.0/20) 



ADD A NETWORK INTERFACE



10. Select a **Network** and **Subnetwork**. This 2nd subnet will be used for your data plane communications and should've been created before starting the deployment process.
- Please note:** The control and data NICs cannot share a subnet. They must have separate subnets.
 - Optional:** Users can select Ephemeral under External IP if they wish for their Data NIC to be assigned a randomized public IP address.



Data network interface
Definitions for the data network interface.

Network interfaces

Edit network interface  

Network vpc02  

Subnetwork dpdk-uscentral1  


External IP  

None

Ephemeral

DONE


11. Click **Done** when you are happy with your selections.
12. Click **Deploy**.

default default (10.128.0.0/20) 

ADD A NETWORK INTERFACE

Data network interface
Definitions for the data network interface.

Network interfaces

vpc02 dpdk-uscentral1 (10.28.0.0/20) 

ADD A NETWORK INTERFACE

DEPLOY

Your cloudSwXtch instance will now be deployed.

☰

Google Cloud

swtch-sa-project

Search (/) for resources, docs, products, a...

Search

☰

Products & solutions

All products

Jump Start Solutions

Solution deployments

Solution deployments

Solution deployments

This is your solution deployment history and may not reflect changes since the time of the initial deployment

Filter

Enter property name or value

Status	Deployment name	Location	Deployment date	Products	Deployed from	Labels
Deployed	cloudswtch-1	us-central1...	Oct 3, 2023, 5:05:1...		Marketplace	goog-ctk

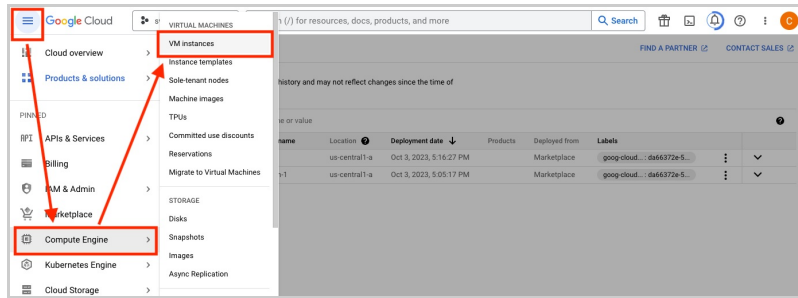
FIND A PARTNER

CONTACT SALES

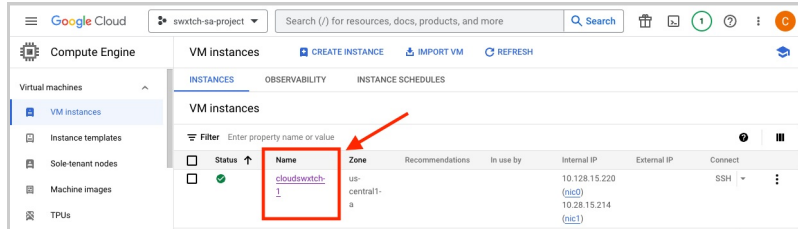
Step Three: Add SSH Key(s)

In order to access your Google Cloud VM instance, you will need to add an SSH key to your cloudSwXtch deployment.

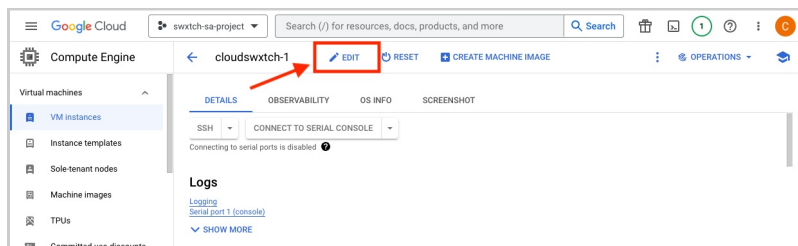
- Click on the **Navigation** menu on the left hand corner, highlight **Compute Engine** and select **VM instances**.



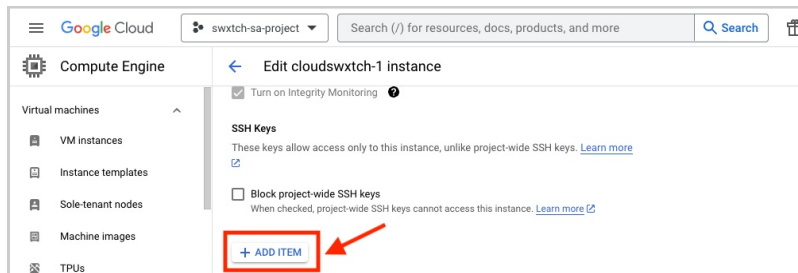
2. Select the name of your cloudSwXtch deployment to open its configuration page.



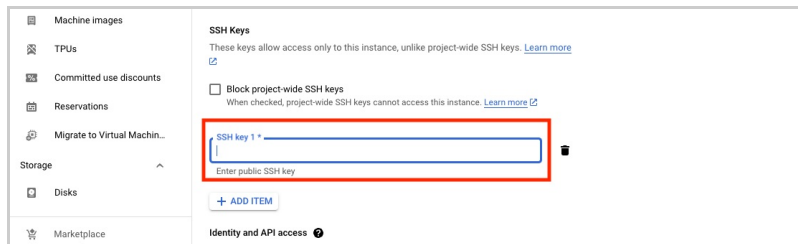
3. Choose **Edit** next to your cloudSwXtch deployment name.



4. Scroll down to **SSH Keys** under **Security and Access**.
5. Click **+Add Item**.



6. Enter your **SSH key**. You can add multiple.



7. Hit **Save** at the bottom of the page.

Upgrading your cloudSwXtch Instance

After deployment, it is recommended to update your cloudSwXtch instance to latest. Please refer to the [Upgrading cloudSwXtch](#) article for more information.

Required Step for BYOL: Contact swXtch.io for a license

Users deploying a BYOL instance of cloudSwXtch will need to contact swXtch.io for a license file. For more information, see [How to License a cloudSwXtch](#).

cloudSwXtch on OCI

WHAT TO EXPECT

In this article, users will find links to articles on deploying a cloudSwXtch in Oracle Cloud Infrastructure (OCI).

Currently, there are only two ways of deploying a cloudSwXtch on OCI:

- [From the Oracle Cloud Marketplace](#)
 - **Note:** This method will require users to contact [swXtch.io](#) for a license.
- [Cloud agnostic cloudSwXtch VM Install](#)
 - **Note:** This method will require the user to already have a Virtual Machine installed with Ubuntu 20.04 that adheres to all the [cloudSwXtch System Requirements](#).

Please stay tuned for more information about alternative methods of installation.

Install cloudSwXtch via OCI Marketplace

WHAT TO EXPECT

In this article, users will learn how to deploy a cloudSwXtch instance via the Oracle Cloud Marketplace.

- [Step One: Navigate to cloudSwXtch in the Oracle Marketplace](#)
- [Step Two: Create Compute Instance](#)
- [Step Three: Attach Secondary VNIC](#)
- [Optional Step for BYOL: Contact swXtch.io for License](#)

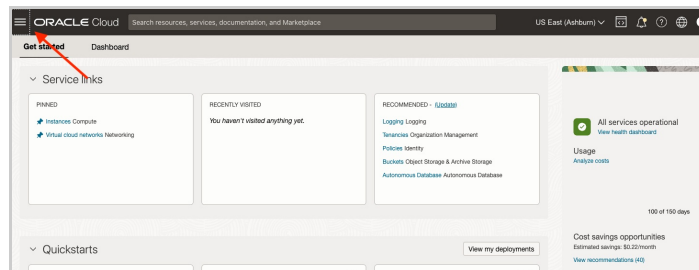
Please note: At this time, our only product offering in OCI is a BYOL instance of cloudSwXtch. This requires a user to contact swXtch.io for a license.

Prerequisites

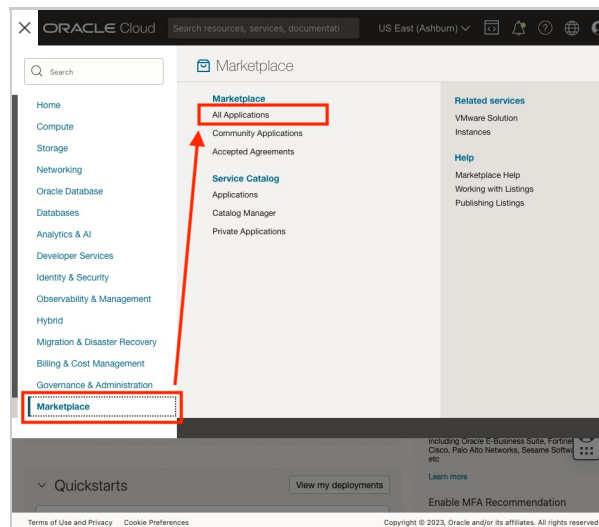
A user should have a **Compartment** established in their Oracle Cloud console before they start to deploy a cloudSwXtch. For more information about compartments, please see the [Managing Compartments](#) page under Oracle Cloud Infrastructure Documentation.

Step One: Navigate to cloudSwXtch in the Oracle Marketplace

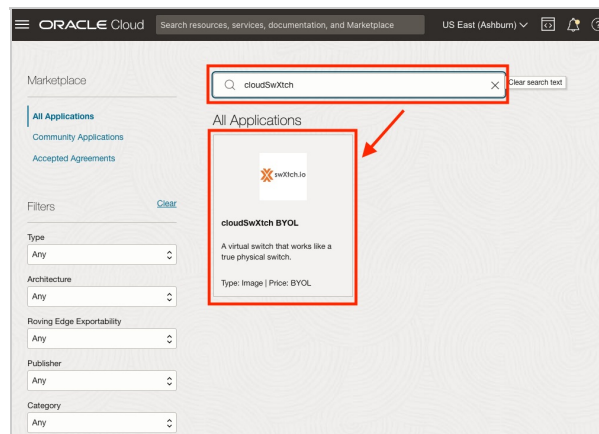
1. Log into Oracle Cloud.
2. Navigate to the Oracle Cloud Marketplace using the Navigation menu at the top left corner.



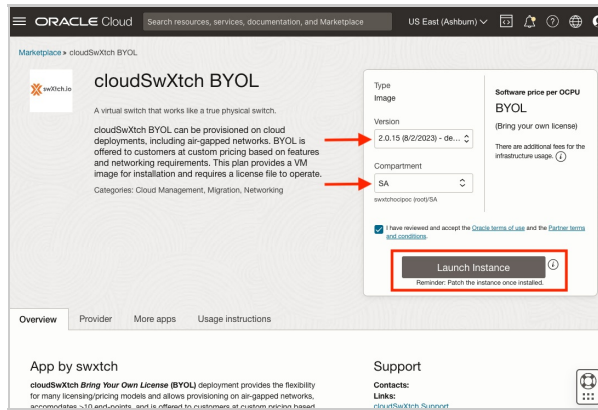
3. Select Marketplace and All Applications.



4. Search for cloudSwXtch and select the product, cloudSwXtch BYOL.



5. Select the Version and the Compartment that you will like to use. It is best to use the default since it will be the most recent version.
6. Click Launch Instance when you're happy with your selections.



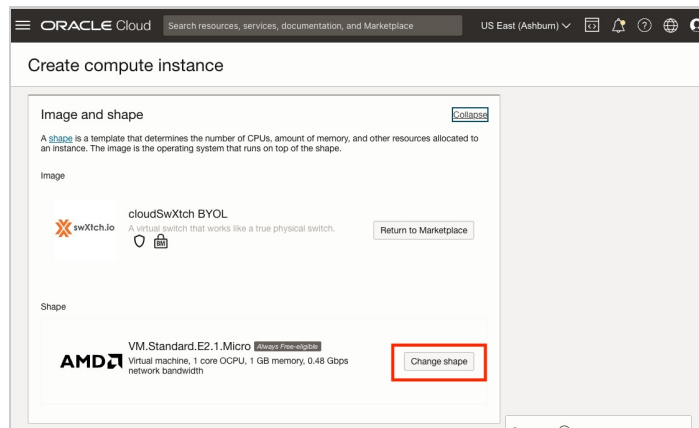
Step Two: Create Compute Instance

1. Give your **Compute Instance** a unique name.
2. Confirm that your desired **Compartment** is populated.
3. **Optional:** Edit selections for **Placement** and **Security**. This is dependent on a user's specific needs.

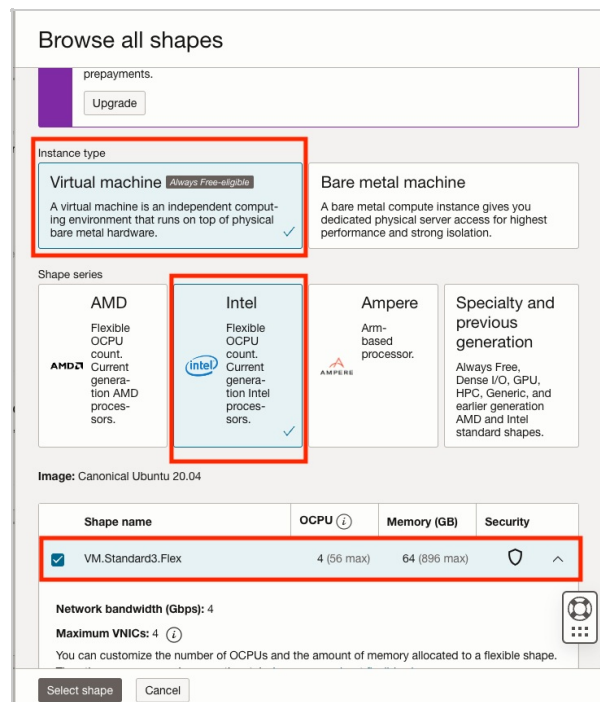
4. Select the **Edit** button for **Image and Shape**.

- a. Confirm that **cloudSwXtch BYOL** is selected for **Image**.

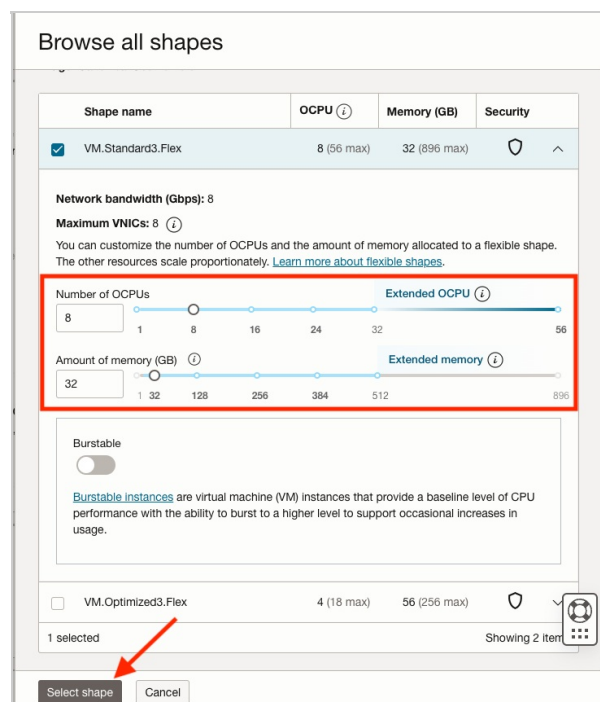
- b. Click **Change Shape**.



c. Choose Intel and VM.Standard3.Flex.



d. Configure the Number of OCPUs and Amount of memory (GB). Please note: It is recommended to have at least eight (8) cores for your cloudSwXtch instance. For more information on recommended sizing, please see [cloudSwXtch System Requirements](#).



e. Click **Select Shape** when you're happy with your selection.

5. Select the **Edit** button for **Primary VNIC information**.

Primary VNIC information

Virtual cloud network: vcn-sa

Subnet: subnet-sa-jh

Launch options: -

Use network security groups to control traffic: No

Assign a public IPv4 address: Yes

DNS record: Yes

Edit

- a. **Optional:** Add a name to your VNIC. If left blank, Oracle will assign it the name of your instance with a note that it is the Primary VNIC.
- b. Assign a VCN to your **Primary VNIC**.
- c. Select a **subnet**. **Please note:** This ctrl subnet will also be used for your secondary VNIC.

ORACLE Cloud

Search resources, services, documentation, and Marketplace

US East

Create compute instance

Primary VNIC information Collapse

A [virtual network interface card \(VNIC\)](#) connects your instance to a [virtual cloud network \(VCN\)](#) and endpoints in and outside the VCN. Having a public IP address is required to make this instance accessible from the internet.

VNIC name *Optional*

Primary network

☒ Select existing virtual cloud network ☐ Create new virtual cloud network ☐ Enter subnet OCID

VCN in SA [\(Change compartment\)](#)

vcn-sa

Subnet

An IP address from a public subnet and an [internet gateway](#) on the VCN are required to make this instance accessible from the internet.

☒ Select existing subnet ☐ Create new public subnet

Subnet in SA [\(Change compartment\)](#)

subnet-sa-ctrl (regional)

- d. Click on **Show advanced options**.

Primary VNIC IP addresses

Private IPv4 address

☒ Automatically assign private IPv4 address ☐ Manually assign private IPv4 address

Public IPv4 address

☒ Automatically assign public IPv4 address

If you're not sure whether you need a public IP address, you can always assign one later.

IPv6 addresses

☐ Assign IPv6 addresses from subnet prefixes

You can only assign one IPv6 address per subnet prefix at first instance creation. Subnets can have more than one IPv6 prefix.

The selected VCN and subnet combination does not support IPv6 addresses. You must enable IPv6 addressing on the VCN and subnet before you can assign IPv6 addresses to this instance.

Show advanced options

- e. Select **Hardware-assisted (SR-IOV) networking** under **Launch options**.

Advanced options Hide advanced options

VCN tags

Subnet tags

☐ Use network security groups to control traffic [?](#)

DNS record

☒ Assign a private DNS record ☐ Do not assign a private DNS record

Hostname *Optional*

No spaces. Only letters, numbers, and hyphens. 63 characters max.

Fully qualified domain name: <hostname>.sactrl.vcn07241005.oraclevcn.com

Launch options

☐ Let Oracle Cloud Infrastructure choose the best networking type

Allow Oracle Cloud Infrastructure to choose the [networking type](#), depending on the instance shape and operating system image.

☐ Paravirtualized networking

For general purpose workloads such as enterprise applications, microservices, and small databases.

☒ Hardware-assisted (SR-IOV) networking

For low-latency workloads such as video streaming, real-time applications, and large or clustered databases. Does not support live migration.

6. Add an **SSH key**.

Add SSH keys

Generate an [SSH key pair](#) to connect to the instance using a Secure Shell (SSH) connection, or upload a public key that you already have.

☒ Generate a key pair for me
 ☐ Upload public key files (.pub)
 ☐ Paste public keys
 ☐ No SSH keys

Download the private key so that you can connect to the instance using SSH. It will not be shown again.

[Save private key](#)
[Save public key](#)

7. Hit **Create** button when you're happy with all of your selections.

Step Three: Attach a Secondary VNIC

When deploying a cloudSwXtch, you will need two VNICs. Both can share a single subnet for control and data plane communications. In this step, we will walkthrough how to attach your secondary VNIC and how to manually add its IP to your cloudSwXtch instance.

1. Make sure that your **Instance with cloudSwXtch installed is running**. You **cannot** attach a secondary VNIC if the machine is off.
2. Select **Attached VNICs** under **Resources**.

Resources
Metrics
Quick actions
Attached block volumes
Attached VNICs
Boot volume
Console connection
Run command

Attached VNICs

A [virtual network interface card \(VNIC\)](#) attaches an instance to a subnet within a VCN and is required for connectivity with other endpoints.

Create VNIC

Name	Subnet or VLAN	State	FQDN	VLAN tag	MAC address
docs-test-instance (Primary VNIC)	Subnet - subnet-sa-lh	Attached	docs-test-...	2726	02:00:17:14:06:D6

3. Click **Create VNIC**.
- a. **Pro-Tip:** Assign your secondary VNIC a user-friendly name. Otherwise, Oracle will assign a randomized ID.
- b. Choose the same **Virtual cloud network** and **ctrl Subnet** as your Primary VNIC.
- c. Select **Save Changes**.

Create VNIC

VNIC name *Optional*

Virtual cloud network in SA [\(Change compartment\)](#)

vcn-sa

Normal setup: subnet

The typical choice when adding a VNIC to an instance.

Advanced setup: VLAN

Only for experienced users who have purchased the Oracle Cloud VMware Solution.

Subnet in SA [\(Change compartment\)](#)

subnet-sa-ctrl (regional)

☐ Use network security groups to control traffic (optional)
 ☐ Skip source/destination check

VNIC IP addresses

Private IPv4 address

Save changes

Cancel

4. Click on the freshly created VNIC's name after it finishes attaching.

Resources
Metrics
Quick actions
Attached block volumes
Attached VNICs
Boot volume
Console connection
Run command
Work requests
OS Management

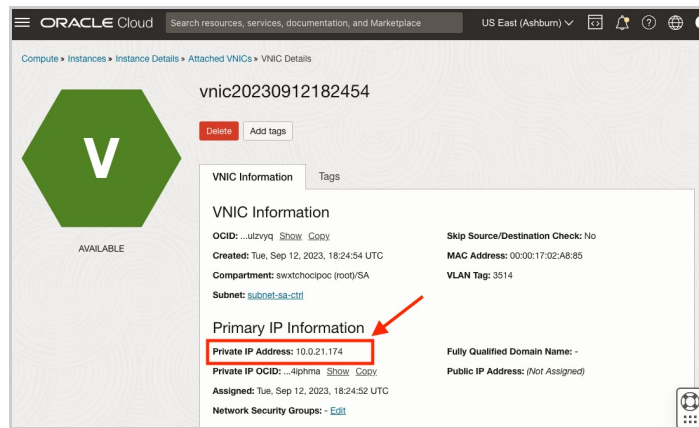
Attached VNICs

A [virtual network interface card \(VNIC\)](#) attaches an instance to a subnet within a VCN and is required for connectivity with other endpoints.

Create VNIC

Name	Subnet or VLAN	State	FQDN	VLAN tag	MAC address
docs-test-instance (Primary VNIC)	Subnet - subnet-sa-lh	Attached	docs-test-...	2726	02:00:17:14:06:D6
vnic20230912182454	Subnet - subnet-sa-ctrl	Attached	-	3514	00:00:17:02:A8:85

5. Record the **Private IP address**. **You will need it later.**



6. Log into your Instance with cloudSwXtch installed.

7. Create the following file in the `/etc/netplan` folder and name it `02-datanic-static-config.yaml`. Please note: You will need to add the Private IP Address of the secondary VNIC into the file below.

Bash	Copy
<pre> network: version: 2 ethernet: ens4: match: macaddress: 02:00:17:09:c2:cf dhcp4: false addresses: - <ADD IP ADDRESS OF 2ND VNIC>/<XX> </pre>	

Where the `<XX>` is the net mask (or network mask) of ctrl-plane CIDR (in single-subnet configuration).

8. Apply the new config (`sudo netplan apply`).

9. Find the file `/etc/iptables/rules.v4` and open it in your editor.

10. Search for the following lines:

Bash	Copy
<pre> -A INPUT -p all -s 10.0.128.0/24 -j ACCEPT -A INPUT -p all -s 10.0.192.0/24 -j ACCEPT </pre>	

11. Replace the `CIDRs` with your own `CIDRs`, corresponding to the ctrl and data subnets. These numbers can be the same if using a single-subnet configuration for both your VNICs.

12. Save file and reboot instance.

The secondary VNIC should now be successfully attached.

Upgrading your cloudSwXtch

After deployment, it is recommended to update your cloudSwXtch instance to latest. Please refer to the [Upgrading cloudSwXtch](#) article for more information.

Optional Step for BYOL: Contact swXtch.io for a license

Users deploying a BYOL instance of cloudSwXtch will need to contact swXtch.io for a license file. For more information, see [How to License a cloudSwXtch](#).

NEXT STEPS

The cloudSwXtch is ready to use. The next step is to install the xNIC on each client expected to get traffic from the cloudSwXtch. See [Installing xNIC](#) for more information on preparing clients. [How to License a cloudSwXtch](#)

Cloud agnostic cloudSwXtch VM Install

WHAT TO EXPECT

In this article, you will learn how to install a cloudSwXtch instance on an existing Linux Ubuntu 20.04 virtual machine. This install process can be used on any cloud but requires a license file from swXtch.io. For more information about VM prerequisites, please see the [cloudSwXtch System Requirements](#).

Pre-Installation Step: Create VM

Before installing cloudSwXtch, you will need to create an **Ubuntu 20.04 virtual machine** on your desired cloud **with connection to the internet**. In addition, it should encompass all the prerequisites outlined in the [cloudSwXtch System Requirements](#)

Step One: Install cloudSwXtch

In this step, users will execute commands in their VMs to manually install a cloudSwXtch instance.

1. Run your freshly created virtual machine using your desired tool.
2. Enter the following command to download the **cloudSwXtch installer script**:

Shell

Bash

Copy

```
token="si=RDONLY&spr=https&sv=2021-06-08&sr=c&sig=xyPF7SyIicagUAEIZViqCHz7RroFTy2Fk1tn2wwvMzc%3D"
curl -X GET -H "x-ms-date: $(date -u)" "https://sdmdevstorage.blob.core.windows.net/imagebuilder/image_install.sh?${token}" -o image_install.sh
chmod +x image_install.sh
```

3. Use the following command to get the latest version of cloudSwXtch. **The latest release is 3.1.0.**

Shell

Bash

Copy

```
ver="v3.1.0"
```

4. Enter the following to download that version.

Shell

Bash

Copy

```
curl -X GET -H "x-ms-date: $(date -u)" "https://sdmdevstorage.blob.core.windows.net/imagebuilder/install-${ver}.tar.gz?${token}" -o install-${ver}.tar.gz
```

5. Execute the installer.

Shell

Bash

Copy

```
sudo ./image_install.sh ${ver}
```

- a. **Precision Time Protocol (PTP)** is not installed at default. Users can install with the argument `--ptp true` with the installer script. This should only be done if they plan to deploy a PTP configuration with their endpoints. For guidance on how to enable PTP for the xNIC, see [Windows](#) and [Linux](#) installation guides.

With both files on the VM (and in the same directory), run the installer as follows:

Bash

Copy

```
sudo ./image_install.sh ${ver} --ptp true
sudo reboot now
```

This will automatically reboot the machine.

Step Two: Contact swXtch.io for a license

Users will need to contact swXtch.io for a license file. For more information, see [How to License a cloudSwXtch](#).

NEXT STEPS

The **cloudSwXtch** is **ready to use**. The next step is to install the xNIC on each client expected to get traffic from the cloudSwXtch. See [Installing xNIC](#) for more information on preparing clients.

Upgrading cloudSwXtch

WHAT TO EXPECT

In this article, users will learn how to update their cloudSwXtch when new versions are available. The following commands are cloud agnostic so they should work regardless of what cloud they're using.

Prerequisites

For major upgrades, it is important to remove your high availability and protocol fanout configurations before upgrading your cloudSwXtch. This will ensure that you have the latest and greatest version of both features.

Users have two ways of doing this: via the UI and via the cloudSwXtch.

- **via the wXcked Eye UI**
 1. Go to the Settings page in wXcked Eye and navigate to either the high availability or protocol fanout tabs.
 2. Delete all your configurations.
- **via the cloudSwXtch**
 1. `cd` into the `/swXtch/` folder on the cloudSwXtch VM.
 2. `ls` all of the files.
 3. Copy the `config.json` as a backup.
 4. Delete the original `config.json`.
 5. Delete any old "install-x.x.x" files. *This is optional for users lacking space.*

You may now proceed with the upgrade.

There are two ways of ensuring your cloudSwXtch is up-to-date: via the cloudSwXtch or via the xNIC.

Upgrading cloudSwXtch via the cloudSwXtch

1. Sign onto the VM where the cloudSwXtch is running.
2. Run the following command:

Shell



Bash	Copy
<pre>sudo /swxtch/swx update -i localhost -v v<desired version></pre>	

Example:

Shell



Bash	Copy
<pre>sudo /swxtch/swx update -i localhost -v v3.0.0</pre>	

Upgrading cloudSwXtch via the xNIC

1. Connect to any VM where an xNIC is running.
2. Run the following command:

Shell



Bash	Copy
<pre>swx update -v <desired version> --ip <ip of cloudSwXtch></pre>	

Example:

Bash	Copy
<pre>swx update -v v3.0.0 --ip 10.5.1.6</pre>	

Note: The `<desired version>` includes a "v" before the version number (e.g. v3.0.0).

Upgrading cloudSwXtch and xNICs

Make sure you upgrade all cloudSwXtches and xNICs in the environment to have the best functionality.

Installing cloudSwXtch Bridge

WHAT TO EXPECT

There are currently 3 types of cloudSwXtch Bridges: Type 3, Type 2 and Type 1. Depending on their needs, it is suggested for users to use either cloudSwXtch Bridge Type 3 or Type 2 for most cases. Bridge Type 1 should really only be used for testing purposes.

In this article, users will learn about the difference between each cloudSwXtch Bridge Types and links on installation instructions for both.

Bridge Type 3

Type 3 of the cloudSwXtch Bridge is DPDK-based, performing similarly to Bridge Type 2. It supports the following:

- **Higher throughput**
- **swXtch Lossless UDP for Ground to Cloud traffic***
- IGMPv3 for SSM
- Protocol Fanout available for all protocols
- Bi-directional traffic between on-prem and the cloud
- Dynamic IGMP joins and leaves

*Only if you're not using high availability.

See [Install cloudSwXtch Bridge Type 3](#) for installation instructions.

Bridge Type 2

Type 2 of the cloudSwXtch Bridge is the more performant version of Bridge Type 1. It supports the following:

- IGMPv3 for SSM
- Protocol Fanout available for all protocols
- Bi-directional traffic between on-prem and the cloud
- **Dynamic IGMP joins and leaves.** When an application in the cloud sends an IGMP join, then the cloudSwXtch in the cloud sends the information to the ground cloudSwXtch as a bridge, allowing the traffic to go through. Dynamic bridge is only supported from ground to cloud, not from cloud to ground.

See [Install cloudSwXtch Bridge Type 2](#) for installation instructions.

Bridge Type 1

Type 1 of the cloudSwXtch Bridge should **only be used for testing purposes** where there is no VPN or Express Route, only access via the Internet. Unlike Bridge Type 2, it **does not support bi-directional traffic between the on-prem and the cloud**. It only supports one direction: on-prem to the cloud. Additionally, it **does not** support dynamic bridge.

See [Install cloudSwXtch Bridge Type 1](#) for installation instructions.

cloudSwXtch Bridge System Requirements

WHAT TO EXPECT

In this article, users will learn about the prerequisites before deploying a cloudSwXtch Bridge. It is recommended for a users to review this page before installing the cloudSwXtch Bridge.

Note that cloudSwXtch Bridge 3 has additional requirements, such as docker engine and third-party package installations.

Network and cloudSwXtch Connectivity

- Before deploying a cloudSwXtch Bridge, a user must already have a cloudSwXtch instance running in any cloud.
- A user should have network connectivity from on-premises to the Virtual Network hosting the cloudSwXtch instance. A user should be able to ping the cloudSwXtch instance from the on-premises network.
- The bridge host must be able to receive and/or send multicast traffic from the local network and send UDP packets to the cloud's Virtual Network using a VPN or Express Route. Internet only access is NOT viable for V3. (See Install cloudSwXtch Bridge V1 if this is your only option.)

Testing On-Premises and Cloud Link

Use one of the following UDP testing tools to measure point from on-prem and into the cloud and to profile your link. This will ensure the link between on-premises and the cloud used by both the cloudSwXtch Bridge and the cloudSwXtch is optimal.

- iPerf: <https://iperf.fr/iperf-doc.php>
- Sockperf: <https://docs.nvidia.com/networking/display/vmav9851/appendix+sockperf+%E2%80%93+udp/tcp+latency+and+throughput+benchmarking+tool>

NIC Requirements

The cloudSwXtch Bridge requires 2 NICs with the non-primary NIC as a Mellanox or Intel card.

- **Mellanox:** ConnectX-5 cards are preferred for the non-primary NIC. For a list of recommendations, please visit the following link: https://www.cisco.com/c/en/us/products/servers-unified-computing/third-party-adapters-listing.html?ft2_general-table0=Nvidia%2FMellanox
- **Intel:** For a list of recommendations (drivers and NIC names), please visit the following link: <https://core.dpkg.org/supported/nics/intel/>

Operating System

A VM or BareMetal bridge host machine running one of the following operating systems with a **minimum 8 cores, 16GB RAM**, a recommended **hard drive of 40GB (20GB min)** and **Kernel 5.11 or greater**:

Bridge Type 3	Bridge Type 2	Bridge Type 1*
Ubuntu 20.04	Ubuntu 20.04	RHEL 7+
RHEL9	RHEL8/CentOS8	CentOS 7+
	RHEL9	Ubuntu 18.04+

*Type 1 only requires a minimum 2 cores, 4GB RAM. It should only be used for testing purposes.

CPUs

CPUs must be at least 2nd generation Intel Core processors, supporting AVX.

Firewall Exceptions

The cloudSwXtch Bridge Type 2 and 3 installer script will automatically open the following ports:

Bash	Copy
<pre>firewall-cmd --add-port=80/tcp --permanent firewall-cmd --add-port=9999/udp --permanent firewall-cmd --add-protocol=igmp --permanent</pre>	

Docker Engine (Type 3 Only)

For cloudSwXtch Bridge Type 3 only, users should install the latest version of Docker Engine for your VM.

- **Ubuntu:** <https://docs.docker.com/engine/install/ubuntu/>
- **RHEL:** <https://docs.docker.com/engine/install/rhel/>

Required Third-Party Packages (Type 3 Only)

This a list of required 3rd party packages for the cloudSwXtch Bridge Type 3 VM. Installation with internet will automatically install the required packages.

Non-internet installation, or air-gapped, will require the packages to be downloaded and installed manually. After installing the packages, it is good practice to reboot the virtual before continuing with cloudSwXtch Bridge Installation.

RHEL	Ubuntu
dialog	dialog
kernel-modules-extra	iproute2
libatomic	libdw-dev
librdmacm	libssl-dev
wget	libstdc++6
	build-essential
	librdmacm-dev
	libnuma-dev
	libmnl-dev
	meson

For DPDK driver install script, the following packages are also required for both RHEL and Ubuntu:

- dkms
- gcc
- git
- make
- pciutils
- net-tools
- python3

How to Install Third Party Packages onto your cloudSwXtch Bridge Type 3 VM

1. Run these commands to enable the following repositories to download the above packages.

a. RHEL9

Bash	Copy
dnf config-manager --enable crb	
dnf -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-9.noarch.rpm	

- i. Depending on the image, the repo name could be different. A command that could help you verify the name of the CodeReady Builder repository is `dnf repolist all`. Once you have the correct name, you can use the first command as follows, replacing {reponame}: `dnf config-manager --enable {reponame}`

b. Ubuntu

Bash	Copy
add-apt-repository ppa:ubuntu-toolchain-r/test	
add-apt-repository ppa:canonical-server/server-backports	

2. Depending on the virtual machine's operating system, run the following command to download the necessary packages:

a. RHEL

Bash	Copy
sudo dnf install dialog kernel-modules-extra libatomic librdmacm wget	

b. Ubuntu

Bash	Copy
sudo apt install dialog iproute2 libdw-dev libssl-dev libstdc++6 build-essential librdmacm-dev libnuma-dev libmnl-dev meson dkms gcc git make pciutils net-tools python3	

Install cloudSwXtch Bridge Type 3

WHAT TO EXPECT

In this article, users will learn how to install cloudSwXtch Bridge Type 3.

Before You Start

Review the [cloudSwXtch Bridge System Requirements](#) to understand the prerequisites to installing cloudSwXtch Bridge Type 3 on your virtual machine.

Pre-Installation

STEP ONE: Update Currently Installed OS Packages

1. Please update all of your currently installed OS packages

a. RHEL

Bash	Copy
<pre>dnf upgrade</pre>	

b. Ubuntu

Bash	Copy
<pre>apt update && apt upgrade</pre>	

2. Reboot VM.

STEP TWO: Update Ubuntu 20.04 to Kernel 5.15 (Ubuntu ONLY)

If the cloudSwXtch Bridge VM is on Ubuntu 20.04, please update to Kernel 5.15. To do this:

1. Use the following commands:

PowerShell	Copy
<pre>sudo apt-get install linux-generic-hwe-20.04 sudo apt update && sudo apt full-upgrade</pre>	

2. Reboot your machine. If a user is running an Air-Gapped install, they will need to download and Install the package manually: <https://vitux.com/how-to-install-latest-linux-kernel-5-15-on-ubuntu-20-04>

3. Use the following to verify the kernel version is at 5:15:

PowerShell	Copy
<pre>uname -r</pre>	

The output will read:

PowerShell	Copy
<pre>5.15.0-1023</pre>	

STEP THREE: Gather Information from the cloudSwXtch Bridge Data NIC (RHEL ONLY)

This step is only required for users running on a RHEL machine.

1. Run the following command on the VM or the on-premises machine of your Data NIC:

Bash	Copy
<pre>ip a</pre>	

2. Make note of the PCIAddress, IP Address, subnetMask, and mac for your cloudSwXtch Bridge data NIC. This information is required for configuration after installation.

STEP FOUR: Review Firewall Exceptions for RHEL [Optional]

The cloudSwXtch Bridge installer script will automatically open the following ports:

Bash	Copy
<pre>firewall-cmd --add-port=80/tcp --permanent firewall-cmd --add-port=9999/udp --permanent firewall-cmd --add-protocol=igmp --permanent</pre>	

To open up additional ports for producing/consuming multicast traffic, use the following command:

Bash	Copy
<pre>sudo firewall-cmd --add-port=<port>/udp --permanent sudo systemctl restart firewallld</pre>	

Note: In some rare cases, it might be helpful to disable the firewall, if cloudSwXtch Bridge installation fails, for troubleshooting purposes.

Installation

This method can be used to install the bridge application onto the bridge host machine. It will only work if the cloudSwXtch instance is up and running and the bridge host has network connectivity to the cloudSwXtch instance.

1. **Open** a bash console on any VM that is on the same control plane network as the cloudSwXtch that you intend to use as the bridge host.
2. **Ping** the cloudSwXtch using your instance name or IP.

Bash	Copy
<pre>ping <cloudSwXtch-instance-IP or instance-name></pre>	

- a. **If the ping fails to find the cloudSwXtch instance by name**, try pinging the IP address of the cloudSwXtch instance. If the IP works, they use the IP address in place of the name in all future commands. This can happen if the default DNS settings are changed for the Virtual Network.

3. Run the cloudSwXtch Bridge installer script:

Bash	Copy
<pre>sudo sh -c "curl -s http://<swxtch-ip>/services/install/swxtch-bridge-install.sh bash -s -- -t 3"</pre>	

- a. If running an air-gapped, or no internet, installation, add `--airgap` or `--ag` parameter after the installer script:

Bash	Copy
<pre>sudo sh -c "curl -s http://<swxtch-ip>/services/install/swxtch-bridge-install.sh bash -s -- -t 3 --airgap"</pre>	

Note: Please ensure that you have completed all prerequisites for an air-gapped installation, including the installation of 3rd party packages as detailed in the cloudSwXtch Bridge System Requirements page.

- b. When prompted, select the network interface that will be used as the data interface (this is the interface for data from the cloudSwXtch Bridge to the cloud):



- c. When prompted, select the network interface that will be used to receive and send multicast traffic (i.e. interface to/from on prem):



Note: The cloudSwXtch Bridge Type 3 config JSON file will use the PCIAddress for the data and user interface names.

The service will automatically initialize. Continue to cloudSwXtch Bridge Type 3 installation Validation if you're not on a RHEL machine.

For Installations Using an Intel Card for Data NIC

After completing the initial install, users on an Intel Data NIC will need to complete the following steps:

1. Run the following command to get a list of PCI addresses.

Bash	Copy
<pre>lspci</pre>	

2. Copy and save the one that will be use in Step 4.
3. Edit the cloudSwXtch Bridge Type 3 configuration file:

Bash	Copy
<pre>nano /var/opt/swxtch/swxtch-bridge3-cfg.json</pre>	

4. Insert the following **nicsConfig** section into the configuration file, replacing the PCIAddress, IP Address, subnetMask, and mac to match your cloudSwXtch Bridge data NIC. **Reminder:** Your Data NIC information (IP Address, subnetMask and mac address) was gathered during [pre-installation](#). The PCIAddress running the lspci was gathered during Step 1.

Bash	Copy
<pre>"nicsConfig": { "0000:65:00.3": { "ip": "10.85.4.130", "subnetMask": "255.255.255.0", "mac": "6c:fe:54:3d:cc:fb" } },</pre>	

- a. **Before:**

```

{
  "bridgeConfig": {
    "ctrlInterfaceName": "eno1",
    "dataInterfaceName": "0000:65:00.3",
    "userInterfaceName": "0000:65:00.3",
    "swxtchCtrlIp": "10.64.20.179",
    "swxtchCtrlPort": 80,
    "swxtchDataIp": "10.64.46.245",
    "swxtchDataPort": 9999,
    "overwriteSenderIp": null,
    "dataGatewayIp": "10.85.4.10",
    "groundToCloudSubscriptions": null,
    "cloudToGroundSubscriptions": [],
    "pollingIntervalMilliseconds": 1000,
    "subscriptionsPollingIntervalMilliseconds": 100,
    "mtuSize": 1500,
    "adaptorsConfig": {},
    "overrideSrcIp": false,
    "slp": {
      "InFlightPacketCount": null,
      "PacketDeliveryRetries": null,
      "ConnectionTimeoutRetries": null,
      "DisableAutoUpdateEstimateFirstNack": null,
      "MinTimeToFirstRetransmissionRequestNs": null,
      "IncomingRetransmissionDelayNs": null,
      "AcknowledgementIntervalNs": null,
      "ChannelBondingStartPort": null,
      "NumberOfChannelBondingPorts": null,
      "ChannelRebondingStartPort": null,
      "NumberOfChannelRebondingPorts": null,
      "MinValueAvgWindow": null,
      "EnableAutoIncoming": null,
      "LatencyFactor": 2
    },
    "xdpModeData": null,
    "xdpModeUser": null,
    "cloudTunIp": null,
    "rpcPort": 10002
  },
  "ha": {
    "producer": false,
    "consumer": false,
    "bufferSizeInPackets": 131072,
    "maxTimeToBufferPacketsMs": 50,
    "protocol": "swxtch"
  },
  "streamSpecs": null
}

```

b. After:

```

"bridgeConfig": {
  "ctrlInterfaceName": "eno1",
  "dataInterfaceName": "0000:65:00:3",
  "userInterfaceName": "0000:65:00:3",
  "nicsConfig": {
    "0000:65:00:3": {
      "ip": "10.85.4.130",
      "subnetMask": "255.255.255.0",
      "mac": "6c:fe:54:3d:cc:fb"
    }
  },
  "swtchCtrlIp": "10.64.20.179",
  "swtchCtrlPort": 80,
  "swtchDataIp": "10.64.46.245",
  "swtchDataPort": 9999,
  "overwriteSenderId": null,
  "dataGatewayIp": "10.85.4.10",
  "groundToCloudSubscriptions": null,
  "cloudToGroundSubscriptions": [],
  "pollingIntervalMilliseconds": 1000,
  "subscriptionsPollingIntervalMilliseconds": 100,
  "mtuSize": 1500,
  "adaptorsConfig": {},
  "overrideSrcIp": false,
  "slp": {
    "InFlightPacketCount": null,
    "PacketDeliveryRetries": null,
    "ConnectionTimeoutRetries": null,
    "DisableAutoUpdateEstimateFirstNack": null,
    "MinTimeToFirstRetransmissionRequestNs": null,
    "IncomingRetransmissionDelayNs": null,
    "AcknowledgementIntervalNs": null,
    "ChannelBondingStartPort": null,
    "NumberOfChannelBondingPorts": null,
    "ChannelRebondingStartPort": null,
    "NumberOfChannelRebondingPorts": null,
    "MinValueAvgWindow": null,
    "EnableAutoIncoming": null,
    "LatencyFactor": 2
  },
  "xdpModeData": null,
  "xdpModeUser": null,
  "cloudTunIp": null,
  "rpcPort": 10002
},
"ha": {
  "producer": false,
  "consumer": false,
  "bufferSizeInPackets": 131072,
  "maxTimeToBufferPacketsMs": 50,
  "protocol": "swtch"
},
"streamSpecs": null
}

```

5. If the **dataGatewayIp** was not populated or does not exist, add it with the appropriate gateway information.
6. Save and close the configuration file.
7. Run the following command:

Bash	Copy
<code>sudo systemctl restart swtch-bridge3-ctrl swtch-bridge3-data</code>	

cloudSwXtch Bridge Type 3 Installation is now complete for RHEL.

Verify cloudSwXtch Bridge Type 3 installation in swXtch-top

Verify cloudSwXtch Bridge Type 3 installation was a success by navigating to swXtch-top on your cloudSwXtch instance. There, you should see the new Bridge Type 3 virtual machine listed under the Components view.

Administrator: Windows PowerShell

Information - dev

ip-10-64-20-179 dev.20240924E(donna-samc-east-cloudswxtch-02-swxtch-all) Licensing License File NEW NOTIFICATIONS

Cloud AWS Bandwidth 0.0 bps/50.06 bps

AccountId 639720666639 Clients 2/30 Bridges 1/1

Region us-east-1 License expiration 2033-Dec-15

SwXtchId i-0621fe002d73761f0

Status OK

Components

Name	Class	Type	Rx (bps)	Rx (pps)	Tx (bps)	Tx (pps)	Primary IP	OS	Version	Hostname
ip-10-64-20-179	swXtch	X1	0.0	0.0	0.0	0.0	10.64.20.179	Ubuntu 20.04	dev.20240924E	ip-10-64-20-179
metal-r13	Bridge	3	0.0	0.0	0.0	0.0	10.85.0.13	AlmaLinux 9.4	dev.20240924E	metal-r13
EC2AMAZ-7EB22UP	XNIC	2	0.0	0.0	0.0	0.0	10.64.29.51	Windows Serv...	dev.20240924E	EC2AMAZ-7EB22UP
samc-aws-agent-002	XNIC	2	0.0	0.0	0.0	0.0	10.64.23.154	Ubuntu 20.04	dev.20240924E	samc-aws-agent-002

Interfaces metal-r13

Name	Index	Ip	SubnetPrefix	Public Ip	MTU	Driver	PciAddress
dpdk-0	0	10.85.4.130	10.85.4.0/24	--	--	igb_uio	0000:65:00.3
eno1	2	10.85.0.13	10.85.0.0/24	--	1500	igb	0000:18:00.0
swxtch-tun-igmp	25	172.10.1.10	172.10.1.0/24	--	4096		

Components

Streams

StreamLinks

Adaptors

HA

Lossless

Settings

PTP

Subscriptions

Notifications

Configurations

Setup

Accessing control and data logs for cloudSwXtch Bridge Type 3

The logs for control or data service can be seen with the following commands:

control

Bash	Copy
sudo journalctl -u swxtch-bridge3-ctrl -f -n 100	

data

Bash	Copy
sudo journalctl -u swxtch-bridge3-data -f -n 100	

Accessing the bridge-install log

If there are any problems with installation, we recommend users export the **bridge-install.log** from your virtual machine's home directory and email it to support@swxtch.io for additional troubleshooting.

Troubleshoot: Configuring Bridge Type 3 Interfaces of Gateway

In the event that your cloudSwXtch Bridge is not sending data to the cloudSwXtch, then it is recommended to review the Bridge Type 3 configuration json file to verify the correct interfaces have been selected. For more information on how to do this, please see the [Configuring cloudSwXtch Bridge Type 2 Instances](#) section.

Alternatively, there can be an issue with the gateway address. For more information, see [Using a Specific Gateway Address for Bridge Type 2](#).

(Both Bridge Type 2 and Type 3 are similar in their JSON file formatting.)

Configuring cloudSwXtch Bridge Type 3

There may be some scenarios that require special configuration for the cloudSwXtch Bridge Type 3, similarly to Bridge Type 2. For more information, see [Bridge Type 2 under Configuring cloudSwXtch](#).

cloudSwXtch Bridge Type 3 Commands

After deploying your cloudSwXtch Bridge Type 3, a user can execute commands to stop, start, and restart their instance. They can execute these commands in the command window of their cloudSwXtch Bridge.

STOP

Bash	Copy
sudo systemctl stop swxtch-bridge3-ctrl	

START

Bash	Copy
<pre>sudo systemctl start swxtch-bridge3-ctrl</pre>	

RESTART

Bash	Copy
<pre>sudo systemctl restart swxtch-bridge3-data swxtch-bridge3-ctrl</pre>	

Note: This example shows how to restart multiple services. Typically, for configuration changes, only ctrl needs to be restarted. However, if the interfaces are altered in the configuration file, the data service should also be restarted.

Uninstalling cloudSwXtch Bridge Type 3

To uninstall your cloudSwXtch Bridge Type 3 application from your bridge host machine:

1. Execute the following command on the Bridge VM on-prem:

Bash	Copy
<pre>sudo sh -c "curl -s http://<swxtch-ip>/services/install/swxtch-bridge-install.sh bash -s -- -u"</pre>	

Your cloudSwXtch Bridge Type 3 instance should now be uninstalled.

Install cloudSwXtch Bridge Type 2

WHAT TO EXPECT

In this article , users will learn how to install cloudSwXtch Bridge Type 2.

Before Your Start

Review the [cloudSwXtch Bridge System Requirements](#) to understand the prerequisites to installing cloudSwXtch Bridge Type 2 on your virtual machine.

Pre-Installation: Update Ubuntu 20.04 to Kernel 5.15

1. Use the following commands:

Shell



Bash	Copy
<pre>sudo apt-get install linux-generic-hwe-20.04</pre>	

Bash	Copy
<pre>sudo apt update && sudo apt full-upgrade</pre>	

3. Reboot your machine. If a user is running an Air-Gapped install, they will need to download and Install the package manually: <https://vitux.com/how-to-install-latest-linux-kernel-5-15-on-ubuntu-20-04/>
4. Use the following to verify the kernel version is at 5.15:

Shell



Bash	Copy
<pre>uname -r</pre>	

The output will read:

Bash	Copy
<pre>5.15.0-1023</pre>	

Updating RHEL8/CentOS8 with Kernel 5.11 or Greater

For more information on how to update your kernel for RHEL8/CentOS8, please read the following articles:

- <https://www.ubuntumint.com/install-linux-kernel-rhel-8/>
- <https://www.2daygeek.com/changing-default-kernel-rhel-8-rhel-9/>

WARNING: Update your kernel at your own risk for RHEL/CentOS8.

Installation

This method can be used to install the bridge application onto the bridge host machine. It will only work if the cloudSwXtch instance is up and running and the bridge host has network connectivity to the cloudSwXtch instance.

1. **Open** a bash console on any VM that is on the same control plane network as the cloudSwXtch that you intend to use as the bridge host.
2. **Ping** the cloudSwXtch using your instance name or IP.

Text



None	Copy
<pre>ping <swxtch-instance-ip or instance-name></pre>	

- a. **If the ping fails to find the cloudSwXtch instance by name**, try pinging the IP address of the cloudSwXtch instance. If the IP works, they use the IP address in place of the name in all future commands. This can happen if the default DNS settings are changed for the Virtual Network.

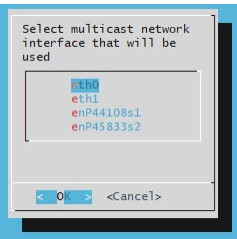
3. **Run** the cloudSwXtch bridge installer script:

Text



None	Copy
<pre>sudo sh -c "curl -s http://<swxtch-ip>/services/install/swxtch-bridge-install.sh bash -s -- -t 2"</pre>	

- a. When prompted, **select** the network interface that will be used to receive and send multicast traffic (i.e. interface to/from on prem).



The service will be automatically initialized and the logs can be seen with:

None	Copy
<pre>sudo journalctl -u swxtch-bridge2 -f -n 100</pre>	

TROUBLESHOOT: Configuring Bridge Type 2 Interfaces or Gateway

In the event that your cloudSwXtch Bridge is not sending data to the cloudSwXtch, then it is recommended to review the Bridge Type 2 configuration json file to verify the correct interfaces have been selected. For more information on how to do this, please see the [Configuring cloudSwXtch Bridge Type 2 Instances](#) section.

Alternatively, there can be an issue with the gateway address. For more information, see [Using a Specific Gateway Address for Bridge Type 2](#).

Configuring cloudSwXtch Bridge Type 2

There may be some scenarios that require special configuration for the cloudSwXtch Bridge. For more information, see [Bridge Type 2 under Configuring cloudSwXtch](#).

cloudSwXtch Bridge Type 2 Commands

After deploying your cloudSwXtch Bridge Type 2, a user can execute commands to stop, start, and restart their instance. They can execute these commands in the command window of their cloudSwXtch Bridge.

STOP

Bash	Copy
<pre>sudo systemctl stop swxtch-bridge2.service</pre>	

START

Bash	Copy
<pre>sudo systemctl start swxtch-bridge2.service</pre>	

RESTART

Bash	Copy
<pre>sudo systemctl restart swxtch-bridge2.service</pre>	

Uninstalling cloudSwXtch Bridge Type 2

To uninstall your cloudSwXtch Bridge Type 2 application from your bridge host machine:

- 1. Execute the following command on the Bridge VM on-prem:

Bash	Copy
<pre>sudo sh -c "curl -s http://<swxtch-ip>/services/install/swxtch-bridge-install.sh bash -s -- -u"</pre>	

Your cloudSwXtch Bridge Type 2 instance should now be uninstalled.

Install cloudSwXtch Bridge Type 1

PREREQUISITES

You will need:

- A cloudSwXtch instance running in a cloud.
- Network connectivity from on-premises to the Virtual Network hosting the cloudSwXtch instance. You should be able to ping the cloudSwXtch instance from the on-premises network.
- A VM or Bare Metal bridge host machine running RHEL 7+, CentOS 7+, or Ubuntu 18.04+ with a minimum of 2 cores, 4GB RAM.
- A bridge host that must be able to receive multicast traffic from the local network and send UDP packets to the cloud Virtual Network.

Direct installation to bridge host -T1

This method can be used to install the bridge application onto the **bridge host** machine. It will only work if the cloudSwXtch instance is up and running and the **bridge host** has network connectivity to the cloudSwXtch instance.

1. **Open** a shell script on any VM that is on the same control plane network as the cloudSwXtch that you intend to use as the bridge host.
2. **Ping** the cloudSwXtch using your instance name.

Text



Bash	Copy
<pre>sudo ping <swxtch-instance-name></pre>	

- a. **If the ping fails to find the switch instance by name**, try pinging the IP address of the cloudSwXtch instance. If the IP works, then use the IP address in place of the **<swxtch-instance-name>** in all future commands. This can happen if the default DNS settings are changed for the virtual network.

3. **Run** the bridge installer script:

Text



Bash	Copy
<pre>curl http://<swxtch-instance-name>/services/install/swxtch-bridge-install.sh bash -s -- -k -t 1</pre>	

Installing xNIC

SUMMARY

- The following article will explain how to install the xNIC component on your Windows and Linux system.
- xNIC is the software that runs on your VM to create a virtual NIC. The xNIC connects your VM to a cloudSwXtch instance.

xNIC System Requirements

There are some major feature considerations to make when deciding what xNIC version to use. These prerequisites are further detailed in the [xNIC System Requirements](#) article.

Linux Installation Guide

[xNIC Linux Installation](#)

The installer script will install the xNIC as a service as well as the utility applications used to verify the operation of the xNIC and cloudSwXtch instance network for a Linux system. See [Testing](#).

Windows Installation Guide

[xNIC Windows Installation](#)

The installer script will install the xNIC as a service as well as the utility applications used to verify the operation of the xNIC and the cloudSwXtch instance network for a Windows system.

xNIC System Requirements

A cloudSwxtch must exist to create a xNIC. See [cloudSwxtch System Requirements](#) for more information.

xNIC software

The xNIC software must be run on each virtual machine that is to be part of the IP multicast network and not a cloudSwxtch or a cloudSwxtch Bridge. This software can be installed on hosts which meet the following requirements:

Available Operating Systems

Linux	Windows
<ul style="list-style-type: none">AlmaLinux 8.8Amazon Linux 2023Centos 8 MinimumOracle Linux 8RHEL 8.8RHEL 9.2Rocky Linux 8Rocky Linux 9Ubuntu 20.04Ubuntu 22.04 <p>Minimum Kernel Version 4.18</p>	<ul style="list-style-type: none">Windows Server 2022Windows Server 2019Windows 11 Pro/EnterpriseWindows 10 Pro/Enterprise

CPU Architecture

x86_x64

Network Connectivity

1 NIC or 2 NICs for higher performance (one for each sub-net: ctrl-subnet and data-subnet)

1 NIC vs. 2 NICs

An xNIC instance may have 1 or 2 NICs depending on the subnet configuration of the cloudSwxtch.

- If a cloudSwxtch has 2 NICs sharing a single subnet, an xNIC needs only 1 NIC (control). This NIC will share the same single subnet for control and data plane communications as the cloudSwxtch.
- For high performance, a cloudSwxtch should have 2 NICs using 2 different subnets, an xNIC will need 2 NICs connected to separate subnets:
 - A subnet for control plane traffic (referred to as the **ctrl-subnet** from here on).
 - A subnet for data plane traffic (referred to as the **data-subnet** from here on).

Subnet Selection

The subnets must be the same subnets used for the cloudSwxtch.

The install requires a simple command that installs the xNIC from the cloudSwxtch. The install typically takes less than one minute per host. See the installation sections for more details.

Tunnel network for xNIC Type 1

The xNIC software must be installed on each virtual machine that is to send or receive multicast traffic. For xNIC Type 1, the software will create a tunnel network interface (called **swxtch-tun0** for Linux and **swxtch-tun** for Windows) that presents to the application a network subnet of 172.30.X.Y. Each virtual machine running the xNIC software will be assigned an IP address in this range.

Please note: In almost all scenarios, it is recommended that users install xNIC Type 2 as this will bypass this requirement.

NOTE:

The swxtch tunnel interface should only be used for multicast traffic. Any other network traffic should target other network interfaces.

Install xNIC on Linux

WHAT TO EXPECT

The xNIC is a lightweight service that must be installed on every VM sending or receiving multicast traffic to and from a cloudSwXtch. An xNIC should not be installed on a cloudSwXtch or cloudSwXtch Bridge VM.

In this article, users will learn how to install the xNIC software in the Linux systems.

Installing xNIC for Linux

BEFORE YOU START

Review [xNIC System requirements](#).

Network Acceleration

If using Azure, the data-subnet must have the "Network Acceleration" feature enabled.

Running the Install Script

To make installation easy, the xNIC is installed from the cloudSwXtch instance via a one-time shell command. The xNIC is matched to the attached cloudSwXtch instance and should be reinstalled if the cloudSwXtch version changes.

To run the install:

1. Open a terminal on the VM you wish to install the xNIC software on.
2. Verify network connectivity to the cloudSwXtch instance by "pinging" the switch.

Bash

```
ping <cloudSwXtch-instance-name>
```

Copy

- a. If the ping fails to find the cloudSwXtch instance by name, try pinging the IP address of the cloudSwXtch instance. If the IP works, then use the IP address in place of the <switch-instance-name> in all further commands.This can happen if the DNS settings are not configured for the virtual network.

Review Firewall Exceptions

The installer script will automatically open ports 10800 and 9999.

To open up additional ports for producing/consuming multicast traffic, use the following command:

Bash

```
sudo firewall-cmd --add-port=<port>/udp --permanent
sudo systemctl restart firewalld
```

Copy

3. Run the following installer script:

Bash

```
curl http://<cloudSwXtch-instance-name>/services/install/swxtch-xnic-install.sh | bash
```

Copy

Alternatively, you can run the install script after downloading it using the **wget** command:

Bash

```
wget http://<cloudSwXtch-ip>/services/install/swxtch-xnic-install.sh
chmod +x swxtch-xnic-install.sh
./swxtch-xnic-install.sh
```

Copy

The installer script will install the xNIC as a service and a set of utility applications that can be used to verify the operation of your cloudSwXtch network. Refer to [Testing](#) for more details.

A successful install is shown below:

```
testadmin@agent-101:~$ curl http://10.2.128.10/services/install/swxtch-xnic-install.sh | bash
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
Dload  Upload   Total   Spent    Left   Speed
100 21432    0 21432    0    0   5232k    0 --:--:-- --:--:-- --:--:-- 5232k
xNIC installer detected ubuntu 20.04. Installing xNIC version 1.
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
Dload  Upload   Total   Spent    Left   Speed
100 7308k   100 7308k    0    0   113M    0 --:--:-- --:--:-- --:--:-- 113M
Selecting previously unselected package swxtch-xnic.
(Reading database ... 141654 files and directories currently installed.)
Preparing to unpack swxtch-xnic_1.0.0_ubuntu20.04_amd64.deb ...
Unpacking swxtch-xnic (1.0.0) ...
Setting up swxtch-xnic (1.0.0) ...
/var/opt/swxtch/swxtch-xnic.conf has been updated.
Service swxtch-xnic.service started
testadmin@DSd-agent-101:~$
```

IF THE INSTALL FAILS:

Validate that the VM has at least two NICs and the NICs are on the same subnets for control and data as the cloudSwXtch. The ctrl-subnet should be assigned to the primary NIC.

If you are using Azure, validate that the data-subnet has "Network Acceleration" feature enabled.

Setting the rp_filter on Linux

During xNIC installation, the Linux rp_filter is set to loose mode by default at runtime. This allows for the xNIC to work on asymmetric networks, meaning that it can receive packets from machines outside of its subnet if the source is routable.

To opt out of this configuration, navigate to the **xnic.json** file after completing the xNIC installation process. This file can be found in **/var/opt/swxtch/xnic.json**. To edit the file, one option is to use nano as shown below:

Bash

Copy

```
sudo nano /var/opt/swxtch/xnic.json
```

Next to "overrideSourceIP" in the json file, change the parameter to **true**. Save the file and restart the xNIC VM. This will set the rp_filter back to the original mode and will remain like that for future reboots. Note that this means our software will do source network address translation on incoming packets.

Additional Arguments

There are additional arguments when installing the xNIC.

Note that the **ctrl-** and **data-** interfaces are from the VM the xNIC is installed. These will be set automatically by the installer. There may be some instances where you will need to specify them. For example, if you have three network interfaces and you want to specify what you want to use for ctrl or data, you can manually select them using the - **ctrl_interface <interface index>** or **-data_interface <interface index>** arguments. Also, these argument help in complex contexts where the agent is in a different vNet/VPC from the cloudSwXtch.

A full list of arguments is detailed below:

Bash

Copy

```
$ ./swxtch-xnic-install.sh -h
Usage: ./swxtch-xnic-install.sh [OPTIONS]
-t <1|2>                xNIC type to install (default: 2 if supported in this OS, 1 otherwise)
-u                    uninstall xNIC instances
--ctrl_interface <interface name>  manual selection of the Control interface
--data_interface <interface name>  manual selection of the Data interface
--ptp <false|true>          installing of Precision Time Protocol (default: false)
--verify              Verify install package with respective signature file (sig_public.pem if public_key is not
setted).
--public_key <public key file>    Public key file to use for signature verification (used with verify parameter).
-h | --help              shows this help
```

Note: There is an option for users to switch between xNIC Type 1 and Type 2, latter being the default. All installation instructions and system requirements are solely for Type 2. It is not recommended to use Type 1 unless otherwise suggested by swXtch.io Support.

Precision Time Protocol Installation

By default, precision time protocol is not enabled during xNIC installation. To install the Precision Time Protocol feature, please run the following command:

PowerShell

Copy

```
wget http://<swxtch-ip>/services/install/swxtch-xnic-install.sh
chmod +x swxtch-xnic-install.sh
./swxtch-xnic-install.sh -t 1 --ptp true
```

Verifying Installer Files [Optional]

Prior to installing the xNIC, the user may need to verify the authenticity of swXtch.io's installer files. While it is not a necessary step in the installation process, it is still an available option to users with security protocols that require files to be validated. **Please note:** This option is only available in cloudSwXtch versions 2.2 or greater.

After downloading the **swxtch-xnic-install.sh** file on your VM, run the following steps:

1. Download the public key from swXtch.io using the following command on your VM:

Shell



Bash

Copy

```
curl https://services/swxtch.io/assets/sig_public.pem
```

2. Move the public key into the same directory as the swxtch-xnic-install.sh file.
3. Run the install command with the **--verify** argument:

Shell



Bash	Copy
<pre>./swxtch-xnic-install.sh --verify</pre>	

The `--verify` argument will download the `.sig` file associated with the xNIC installer file based on the VM's operating system, verify with the public key, and proceed to install the xNIC onto the VM as a service.

Testing

xNIC installation includes a set of utility applications that you can use to verify the operation of your cloudSwXtch network. Refer to [Testing](#) for details.

- **swxtch-top**: An application to display real-time statistics of the cloudSwXtch instance.
- **swxtch-perf**: An application to produce and consume unicast and multicast traffic for testing purposes.

Running **swxtch-top** on Linux

<swxtch-hostname>: name of your existing swxtch or "host" swxtch

Bash	Copy
<pre>swxtch-top dashboard --swxtch <swxtch-hostname></pre>	

Uninstalling xNIC on Linux

To uninstall xNIC on Linux, users can follow the steps in the [xNIC Linux Uninstall Guide](#).

Upgrading xNIC on Linux

To upgrade xNIC on Linux, users can follow the steps in the [xNIC Linux Upgrade Guide](#).

xNIC Linux Uninstall

WHAT TO EXPECT

In this article, users will learn how to remove the xNIC from their Linux system for both Ubuntu and Redhat.

Uninstalling xNIC on Linux

- 1. **Open** a shell on the host VM. The host VM is the VM where you wish to uninstall the xNIC software.
- 2. **Run** the following command depending the xNIC version:

None	Copy
<pre>curl http://<swxtch-instance-name>/services/install/swxtch-xnic-install.sh bash -s -- -u</pre>	

- 3. **The uninstall script will remove Linux xNIC.**

xNIC Linux Upgrade

BEFORE YOU START

When a cloudSwXtch has been updated, it's recommended to update connected xNICs as well.

In this article, users will be able to use the appropriate script to upgrade their xNIC.

Upgrading Linux xNIC

24/7 Operations

If the services need to be up and running 24/7, swXtch.io suggests that redundant systems exist for which will be referred to as "Main" and "Backup". During an upgrade the Backup system should be upgraded, then the traffic should be routed to the Backup while the Main is upgraded.

You can use the following command to uninstall the existing xNIC and upgrade it.

1. Run the installer script:

Shell



Bash	Copy
<pre>curl http://<swxtch-instance-name>/services/install/swxtch-xnic-install.sh bash</pre>	

Additional Arguments

There are additional options when installing the xNIC.

Note that the **ctrl-** and **data-** interfaces are from the VM the xNIC is installed. These will be set automatically by the installer. There may be some instances where you will need to specify them. For example, if you have three network interfaces and you want to specify what you want to use for ctrl or data, you can manually select them using the **-ctrl_interface** or **-data_interface** arguments. Also, these argument help in complex contexts where the agent is in a different vNet/VPC from the cloudSwXtch.

For xNIC 1 on Linux, multiple xNICs can be installed on one VM by using the **-i** and the **--tun-subnet** arguments. In this case, the control interface will be the same while the data interface will differ for each xNIC on the Linux VM.

A full list of arguments is detailed below:

Bash	Copy
<pre>\$./swxtch-xnic-install.sh -h Usage: ./swxtch-xnic-install.sh [OPTIONS] -t <1 2> xNIC type to install (default: 2 if supported in this OS, 1 otherwise) -u uninstall xNIC instances --ctrl_interface <interface name> manual selection of the Control interface --data_interface <interface name> manual selection of the Data interface --ptp <true false> installing of Precision Time Protocol (default: true) -h --help shows this help</pre>	

Install xNIC on Windows

WHAT TO EXPECT

The xNIC is a lightweight service that must be installed on every VM sending or receiving multicast traffic to and from a cloudSwXtch. An xNIC should not be installed on a cloudSwXtch or a cloudSwXtch Bridge VM.

In this article, users will learn how to install the xNIC software on Windows systems.

Installing xNIC for Windows

BEFORE YOU START

Review [xNIC System Requirements](#).

Firewall Restrictions
The Windows installation process adds rules to Windows Defender Firewall, which allow for traffic through the UDP ports 10800 and 9999. The rule names are SwXtchControl, SwXtchData, and SwXtchTun.

Network Acceleration
If using Azure, the data-subnet must have the "Network Acceleration" feature enabled.

Running the Install script

To make installation easy, the xNIC is installed from the cloudSwXtch instance via a one-line shell command. The xNIC is matched to the attached cloudSwXtch instance and should be reinstalled if the cloudSwXtch version changes.

The xNIC takes less than a minute to install on an existing VM.

To run the install:

1. Open a PowerShell terminal on the Windows VM that you aspire to install the xNIC software on. If you are working on Windows 11, please use Windows Terminal instead for installation.
2. Verify network connectivity to the cloudSwXtch instance by "pinging" the switch.

Bash

Copy

```
ping <cloudSwXtch-instance-name>
```

Ping Fails

If the ping fails to find the cloudSwXtch instance by name, try pinging the IP address of the cloudSwXtch instance. If the IP works, then use the IP address in place of the `<switch-instance-name>` in all further commands.

This can happen if the default DNS settings are changed for the virtual network.

3. The installer script will automatically remove any firewall restrictions to UDP ports 10800 and 9999. The cloudSwXtch sends UDP packets to these ports as part of normal operation.

Special Rules for Windows Defender Firewall

It is recommended to simply turn off the firewall. Additionally, users can open up additional ports for producing/consuming multicast traffic by using the following command in PowerShell:

Bash

Copy

```
New-NetFirewallRule -Name 'rule_name' -DisplayName 'rule_name' -Enabled True -Direction Inbound -Protocol UDP -Action Allow -LocalPort 1234
```

4. Ensure Windows SecureBoot UEFI is disabled. This can be enabled or disabled from the VM or EC2 configuration in your preferred cloud.

5. Download and run the installer script in Powershell as an administrator:

Bash

Copy

```
Invoke-WebRequest -Uri 'http://<cloudSwXtch-instance-name>/services/install/swxtch-xnic-win-install.ps1' -Outfile swxtch-xnic-win-install.ps1
```

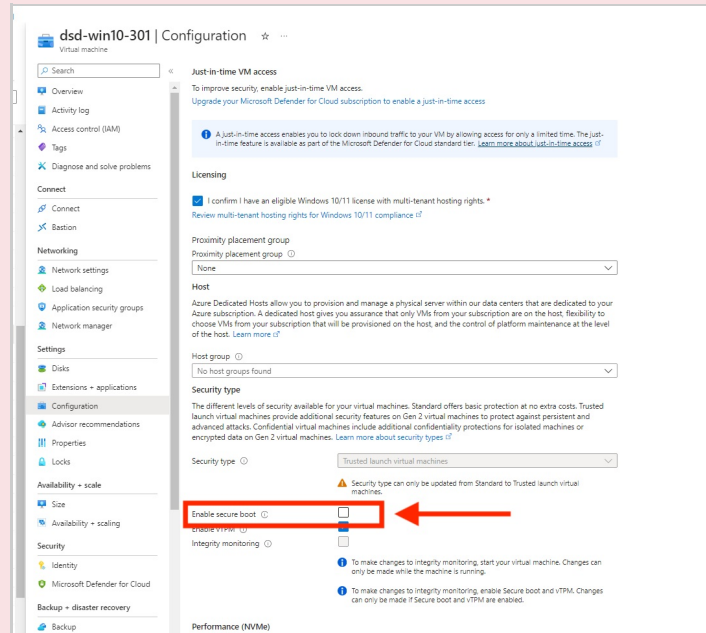
Secure Boot Error

The following error will appear if your Secure Boot is enabled or if you're not running as an administrator in Powershell. The below example shows running as an administrator, therefore it is likely that secure boot is incorrectly enabled on the VM.

```
Administrator: Windows PowerShell

PS C:\Users\testadmin> Invoke-WebRequest -Uri 'http://10.1.1.6/services/install/swtch-xnic-win-install.ps1' -Outfile swtch-xnic-win-install.ps1
PS C:\Users\testadmin> .\swtch-xnic-win-install.ps1
***** xNIC installation script *****
OS detected is Microsoft Windows 10 Pro
Getting Tag/Release version.
Success: downloaded 142 bytes.
Tag/Release is Version dev.cloudswtch.2.1.1.07
xNIC Install is requested.
Checking if XDP driver is installed
SecureBoot UEFI is enabled. Unable to proceed
Please disable SecureBoot UEFI to be able to proceed
Installation cancelled
PS C:\Users\testadmin>
```

To disable Secure Boot on your Azure VM, go to the Microsoft Azure Portal and navigate to your VM. Under **Settings**, select **Configuration** and scroll down to **Security Type**. De-select **Enable secure boot** and then select **Apply** to confirm your changes.



To disable Secure Boot on your GCP VM, go to the Compute portal and navigate to your VM Instances and select your desired VM. Under Details, scroll down to Security and Access and validate that Secure Boot is off under Shielded VM. If it is enabled, stop your VM and select edit to disable the option. Save and restart VM.

Additional Arguments

There are additional options when installing the xNIC. To see these options, use the **-h** argument.

The **ctrl-** and **data-** interfaces are from the VM the xNIC is installed. These will be set automatically by the installer. There may be some instances where you will need to specify them. For example, if you have three network interfaces and you want to specify what you want to use for ctrl or data, you can manually select them using the **-ctrl_interface <interface index>** or **-data_interface <interface index>** arguments detailed below.

Note: There is an option **(-t)** for users to switch between xNIC Type 1 and xNIC Type 2 (default). All installation instructions and system requirements are solely for Type 2. It is not recommended to use Type 1 unless otherwise suggested by swTch.io Support.

Bash	Copy
<pre>PS C:\Users\testadmin> .\swtch-xnic-win-install.ps1 -h Running on Microsoft Windows Server 2022 Datacenter Usage: C:\Users\testadmin\swtch-xnic-win-install.ps1 [OPTIONS] -t <1 2> xNIC type to install (default: 2 if supported in this OS, 1 otherwise) -u uninstall xcd xNIC only (no other options allowed) -unattended unattended installation (in case of reboot, the user will not be prompted) -ctrl_interface <interface index> manual selection of the Control interface -data_interface <interface index> manual selection of the Data interface -ptp <false true> installing of Precision Time Protocol (default: false) -verify verify install package with respective signature file (sig_public.pem if public_key is not setted). -public_key <public key file> public key file to use for signature verification (used with verify parameter). -h shows this help</pre>	

Please note: The **ctrl_interface** and **data_interface** commands should only be used in complex configurations where the installer cannot locate them. Contact support@swTch.io for more information.

Precision Time Protocol Installation

By default, precision time protocol is not enabled during xNIC installation. To install the Precision Time Protocol feature, please run the following command:

PowerShell

Copy

```
.\swxtch-xnic-win-install.ps1 --ptp true
```

Getting the interface index for ctrl_interface and data_interface

To get the interface index, you can run the following command:

Bash

Copy

```
get-netipconfiguration
```

A sample of the response will be returned with the InterfaceIndex (***) listed for both:

Bash

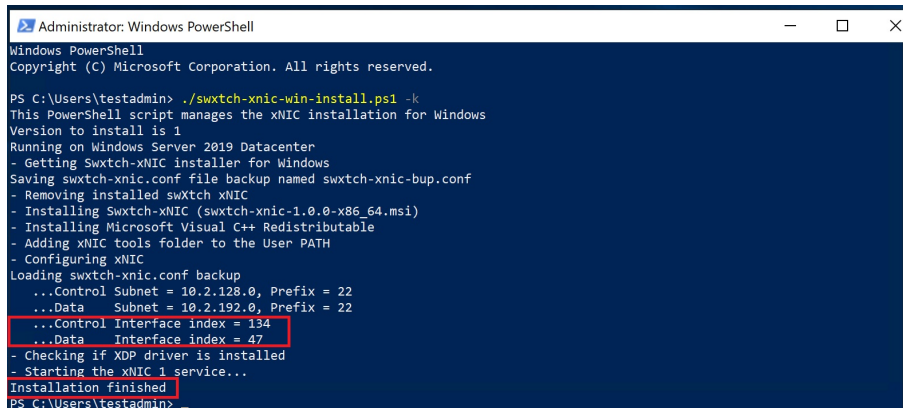
Copy

```
PS C:\Users\testadmin> get-netipconfiguration

InterfaceAlias      : Ethernet
InterfaceIndex      : 36 ***
InterfaceDescription : Microsoft Hyper-V Network Adapter
NetProfile.Name     : Network
IPv4Address         : 10.2.128.75
IPv6DefaultGateway  :
IPv4DefaultGateway  : 10.2.128.1
DNSServer           : 168.63.129.16

InterfaceAlias      : Ethernet 2
InterfaceIndex      : 60 ***
InterfaceDescription : Microsoft Hyper-V Network Adapter #2
NetProfile.Name     : Unidentified network
IPv4Address         : 10.2.192.82
IPv6DefaultGateway  :
IPv4DefaultGateway  :
DNSServer           : 168.63.129.16
```

7. The installer script will install a **Windows service** called **swXtchNIC**:



```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\testadmin> .\swxtch-xnic-win-install.ps1 -k
This PowerShell script manages the xNIC installation for Windows
Version to install is 1
Running on Windows Server 2019 Datacenter
- Getting Swxtch-xNIC installer for Windows
Saving swxtch-xnic.conf file backup named swxtch-xnic-bup.conf
- Removing installed swXtch xNIC
- Installing Swxtch-xNIC (swxtch-xnic-1.0.0-x86_64.msi)
- Installing Microsoft Visual C++ Redistributable
- Adding xNIC tools folder to the User PATH
- Configuring xNIC
Loading swxtch-xnic.conf backup
...Control Subnet = 10.2.128.0, Prefix = 22
...Data Subnet = 10.2.192.0, Prefix = 22
...Control Interface index = 134
...Data Interface index = 47
- Checking if XDP driver is installed
- Starting the xNIC 1 service...
Installation finished
PS C:\Users\testadmin>
```

8. **Reboot** your machine once the installation is complete. This will enable you to execute cloudSwXtch tools properly from your user home directory such as **swxtch-top**.

Errors

The control and data interfaces should have proper numbers. A 0, or negative number, indicates an error in the configuration of the control or data subnets for the xNIC. The control and data subnets of the cloudSwXtch and the xNICs should be the same.

If you are using Azure, validate that the data-subnet has "Network Acceleration" featured enabled.

Testing

The installation includes a set of utility applications that you can use to verify the operation of your cloudSwXtch network. Refer to [Testing](#) for details.

- **swxtch-top.exe** : An application to display real-time statistics of the cloudSwXtch instance.
- **swxtch-perf.exe** : An application to produce and consume multicast traffic for testing purposes.

Running swtch-top on Windows

Bash

Copy

```
swtch-top --swtch <cloudSwXtch-hostname>
```

- **<cloudSwXtch-hostname>**: the name of your existing or "host" cloudSwXtch

Uninstalling xNIC on Windows

To uninstall xNIC on Windows, users can follow the steps in the [Uninstall xNIC on Windows](#) guide.

Upgrading xNIC on Windows

To upgrade xNIC on Windows, users can follow the steps in the [Upgrade xNIC on Windows](#) guide.

Uninstall xNIC on Windows

WHAT TO EXPECT

In this article, users will learn how to remove the xNIC from their Windows system.

Uninstalling xNIC on Windows

When uninstalling xNIC on Windows, please **do not** uninstall using the Add/Remove Programs feature. It is important to use the command below instead for uninstall.

- 1. **Open** Powershell on your Windows system (command window if Windows 11).
- 2. Run the following command:

Text



None	Copy
<pre>.\swxtch-xnic-win-install.ps1 -u</pre>	

Upgrade xNIC on Windows

WHAT TO EXPECT

When a cloudSwXtch has been updated, their xNIC should be upgraded as well. This is very simple since you will only need to reinstall the script. The installer will automatically remove the older version of xNIC.

In this article, users will learn to use the appropriate script to upgrade their xNIC.

Make sure that you have the latest version of cloudSwXtch installed. You can find information about how to upgrade your cloudSwXtch by clicking here: [Upgrading cloudSwXtch](#). You can also upgrade your cloudSwXtch by deleting and recreating the instance.

Upgrading xNIC on Windows

- 1. Open PowerShell. If you are using Windows 11, please use Windows Terminal.
- 2. Download the installer script:

Bash	Copy
<pre>Invoke-WebRequest -Uri 'http://<swxtch-instance-name>/services/install/swxtch-xnic-win-install.ps1' -Outfile swxtch-xnic-win-install.ps1</pre>	

- 3. Run the script. Please use the appropriate command for your version. Note: xNIC Type 2 is the default.

None	Copy
<pre>./swxtch-xnic-win-install.ps1</pre>	

The latest version of the Windows xNIC will be installed.

Remember to Reboot

Reboot the machine after the upgrade is complete. You must do this to be able to execute the cloudSwXtch tools properly from your user home directory.

Additional Arguments

To see all the options available for the xNIC installation/update script, use the -h argument.

Note that the ctrl- and data- interfaces are from the VM the xNIC is installed. These will be set automatically by the installer. There may be some instances where you will need to specify them. For example, if you have three network interfaces and you want to specify ctrl or data interfaces, you can manually select them using the -ctrl_interface <interface index> or -data_interface <interface index> arguments detailed below.

Bash	Copy
<pre>PS C:\Users\testadmin> .\swxtch-xnic-win-install.ps1 -h Usage: C:\Users\testadmin\swxtch-xnic-win-install.ps1 [OPTIONS] -t <1 2> xNIC type to install (default: 2 if supported in this OS, 1 otherwise) -u uninstall xcd xNIC only (no other options allowed) -unattended unattended installation (in case of reboot, the user will not be prompted) -ctrl_interface <interface index> manual selection of the Control interface -data_interface <interface index> manual selection of the Data interface -ptp <true false> installing of Precision Time Protocol (default: true) -h shows this help</pre>	

Install xNIC on Kubernetes

WHAT TO EXPECT

For Kubernetes, the xNIC is a lightweight daemonset that must be installed on every node with pods sending or receiving cloudSwXtch traffic. This creates a virtual network interface within the node in a Kubernetes Cluster. Applications that use IP multicast should target this virtual network interface.

In this article, you will learn how to install xNIC on K8s.

Supported Types

Below is a list of supported K8s Clusters. The * denotes default.

- AKS: Kubenet CNI*, Cilium CNI, or Calico CNI
- EKS: Amazon VPC CNI* and Cilium CNI
- GKE: Legacy (Calico)* and Dataplane V2 (Cilium)

The following operating systems in a pod are supported for the xNIC on K8s: **RHEL 8, RHEL 9, CentOS 8, Ubuntu 20.04, or Ubuntu 22.04.**

Installation

The installation process can be split into three steps:

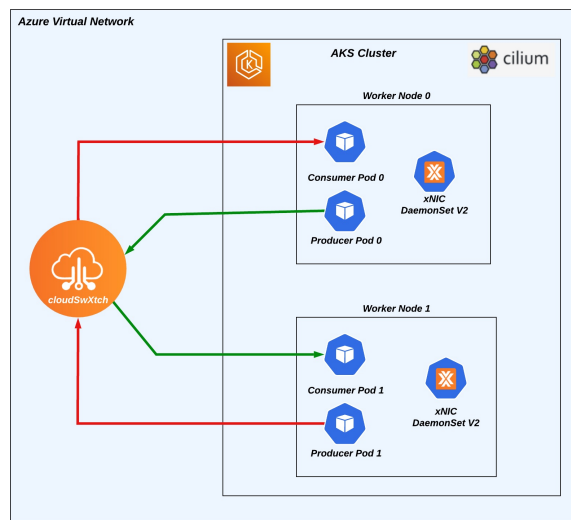
- Create a Kubernetes Cluster (AKS, EKS, or GKE)
- [Install xNIC on K8s Cluster](#)
- [Test xNIC on K8s](#)

Post-Installation

You can learn how to upgrade your xNIC nodes on K8s, [here](#).

xNIC Architecture Diagram

Below is an example architecture for an xNIC installed on AKS with Cilium with communication to and from a cloudSwXtch. Other Virtual Machines (not AKS) with xNICs installed could also communicate with the AKS worker nodes via cloudSwXtch and xNIC v2.



Please note: The producer pod and consumer pod of the same stream must be in different nodes. For example, Consumer Pod 0 is consuming a stream from Producer Pod 1 from a different node. Consumer Pod 0 cannot consume what Producer Pod 0 is creating since they are in the same node.

Install xNIC on K8s Cluster

WHAT TO EXPECT

For Kubernetes, the xNIC is a lightweight daemonset that must be installed on every node with pods sending or receiving cloudSwXtch traffic. This creates a virtual network interface within the node in a Kubernetes Cluster. Applications that use IP multicast should target this virtual network interface.

In this article, users will learn how to install xNIC Daemonset for Kubernetes on one of the supported clouds (AKS, EKS, or GKE).

Overview

Unicast traffic will not be affected by this feature since it will work as it did before. The xNIC will only be used for Multicast traffic. The default interface xNIC will use is `eth0`. It can be installed via your preferred cloud's CloudShell or you can assign a VM as a manager to control your cluster. Either way, it is required to have access to the cloudSwXtch and the cluster.

In this document, we will discuss how to do it via the CloudShell. However, the commands below will work in either the CloudShell or on the VM managing the K8s cluster.

Running the xNIC Daemonset Install Script

BEFORE YOU START

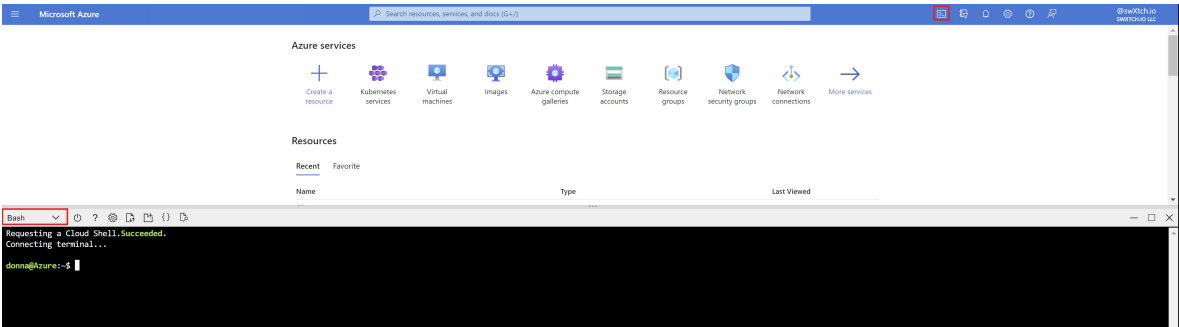
If you haven't already, please create a Kubernetes Cluster. This is a prerequisite before installing the xNIC.

To make installation easy, the xNIC is installed from the cloudSwXtch instance via a one-line shell command. The xNIC is matched to the attached cloudSwXtch and should be upgraded if the cloudSwXtch version changes.

This process takes less than a minute to install on an existing K8s cluster.

To run the install:

1. Ensure your cloudSwXtch is version **2.0.89 or greater**. If it is not upgraded, see [Upgrading cloudSwXtch](#).
2. Sign into your **desired cloud provider**.
3. Open **cloudShell** as Bash.



4. Paste in the following commands, replacing the `<cloudSwXtch_IP>` with your cloudSwXtch's Ctrl IP address.

Bash	Copy
<pre>kubectl run installer --image=busybox -- sh -c "wget http://<cloudSwXtch_IP>/services/install/xnic_ds_installer.sh; sleep 3650" kubectl cp default/installer:/xnic_ds_installer.sh xnic_ds_installer.sh kubectl delete po/installer --grace-period 1 chmod +x xnic_ds_installer.sh</pre>	

5. Run one of the following scripts:

cloudSwXtch with Internet Access:

Bash	Copy
<pre>./xnic_ds_installer.sh</pre>	

cloudSwXtch without Internet Access (Air-Gapped):

Bash	Copy
<pre>./xnic_ds_installer.sh -ag</pre>	

An example of a successful install without INTERNET access is shown below:

Bash	Copy
<pre>\$./xnic_ds_installer.sh -ag [i] Detected Cloud: AZURE [i] Cilium Installation detected [i] Setting CNI to CILIUM... ##### This script modifies the underlying configuration of Cilium CNI to make it compatible with Multicast Networks. It also installs xNIC DaemonSet on the existing cluster. ##### - RUNNING INSTALLER: Airgap - IMAGE: 10.144.0.115:443/xnicv2:airgap - CNI PLUGIN: CILIUM - SWXTCH IP ADDRESS: 10.144.0.115 - AGENT TYPE: XNIC XCD ===== Adjusting BPF filter priority on Cilium ===== Setting flag "bpf-filter-priority" to "50000" configmap/cilium-config patched Done! ===== Restarting Cilium Agents ===== daemonset.apps/cilium restarted daemonset.apps/cilium-node-init restarted Waiting for Cilium Agents to be fully UP and Running.....OK Done! Proceeding with xNIC Installation ===== Creating xNIC ConfigMap ===== configmap/xnic-config created ===== Installing xNIC ===== daemonset.apps/swxtch-xnic created Done! ===== Completed! ===== Please allow a minute for the xNIC DaemonSet to fully spin up before starting to use it. Feel free to follow up on the xNIC Agents installation by running kubectl logs -n kube-system daemonsets/swxtch-xnic -f</pre>	

6. Run the following command to view the xNIC DaemonSet logs in the Bash window:

Bash	Copy
<pre>kubectl logs -n kube-system daemonsets/swxtch-xnic -f</pre>	

7. Use the command below to follow the xNIC DaemonSet status in the Bash window and check if they have started (i.e "Running"):

Plaintext	Copy
<pre>kubectl get pods -l app=swxtch-xnic -n kube-system</pre>	

Example:

Bash	Copy
<pre>user@Azure:~\$ kubectl get pods -l app=swxtch-xnic -n kube-system NAME READY STATUS RESTARTS AGE swxtch-xnic-fc58t 1/1 Running 0 11d swxtch-xnic-kn9hg 1/1 Running 0 11d</pre>	

8. Sign into your cloudSwXtch and enter in the following command to see the new instances in swXtch-top.

Bash	Copy
<pre>swxtch-top</pre>	

Restarting xNIC DaemonSet

To restart xNIC DaemonSet for K8s, run the following command:

Bash	Copy
<pre>kubectl rollout restart ds/swxtch-xnic -n kube-system</pre>	

***Managing Multicast Traffic**

Following are some `tc` commands that can be useful when it comes to allowing/denying either incoming or outgoing multicast traffic on producer and consumer pods. You **must** run these commands **inside** the target producer/consumer pods so that the correct interface name (eth0 in the examples) is picked up.

By default, **ALL** multicast traffic is **allowed** on every pod.

For Outgoing (Traffic leaving the Pod)

Deny ALL outgoing multicast

To deny all outgoing multicast, use the following commands:

Specific syntax:

Bash	Copy
<pre># DENY ALL OUTGOING tc qdisc add dev eth0 root handle 1: prio tc filter add dev eth0 parent 1: protocol ip u32 match ip dst 224.0.0.0/4 action drop</pre>	

Alternatively, users can deny outgoing multicast to specific groups:

General Syntax:

Bash	Copy
<pre># DENY OUTGOING TO SPECIFIC GROUP(S) tc qdisc add dev eth0 root handle 1: prio tc filter add dev eth0 parent 1: protocol ip u32 match ip dst <multicast_group_0> action drop ... tc filter add dev eth0 parent 1: protocol ip u32 match ip dst <multicast_group_n> action drop</pre>	

Example: denying outgoing traffic to multicast group `239.0.0.1` :

Bash	Copy
<pre>tc qdisc add dev eth0 root handle 1: prio tc filter add dev eth0 parent 1: protocol ip u32 match ip dst 239.0.0.1/32 action drop</pre>	

Allow outgoing multicast to a specific group(s) - Deny any other

Bash	Copy
<pre># DENY ALL OUTGOING tc qdisc add dev eth0 root handle 1: prio tc filter add dev eth0 parent 1: protocol ip u32 match ip dst 224.0.0.0/4 action drop # ALLOW SPECIFIC GROUP(S) tc filter add dev eth0 parent 1: protocol ip u32 match ip dst <multicast_group_0> action ok ... tc filter add dev eth0 parent 1: protocol ip u32 match ip dst <multicast_group_n> action ok</pre>	

Example: allowing outgoing traffic **ONLY** to multicast group `239.0.0.1` :

Bash	Copy
<pre>tc qdisc add dev eth0 root handle 1: prio tc filter add dev eth0 parent 1: protocol ip u32 match ip dst 224.0.0.0/4 action drop tc filter add dev eth0 parent 1: protocol ip u32 match ip dst 239.0.0.1/32 action ok</pre>	

Incoming (Traffic entering the Pod)

To **deny ALL incoming multicast**, use the following command:

Specific syntax:

Bash	Copy
<pre># DENY ALL INCOMING tc qdisc add dev eth0 ingress tc qdisc add dev eth0 parent ffff: protocol ip u32 match ip dst 224.0.0.0/4 action drop</pre>	

Alternatively, users can deny incoming multicast for a specific group(s)

General syntax:

Bash	Copy
<pre># DENY INCOMING TO SPECIFIC GROUP(S) tc qdisc add dev eth0 ingress tc filter add dev eth0 parent ffff: protocol ip u32 match ip dst <multicast_group_0> action drop ... tc filter add dev eth0 parent ffff: protocol ip u32 match ip dst <multicast_group_n> action drop</pre>	

Example: denying incoming multicast traffic to multicast group `239.0.0.1` :

Bash	Copy
<pre>tc qdisc add dev eth0 ingress tc filter add dev eth0 parent ffff: protocol ip u32 match ip dst 239.0.0.1/32 action drop</pre>	

In addition, users can specify allowing incoming multicast by a specific group(s) while denying any other:

General syntax:

Bash	Copy
<pre># DENY ALL INCOMING tc qdisc add dev eth0 ingress tc qdisc add dev eth0 parent ffff: protocol ip u32 match ip dst 224.0.0.0/4 action drop # ALLOW SPECIFIC GROUP(S) tc filter add dev eth0 parent ffff: protocol ip u32 match ip dst <multicast_group_0> action ok ... tc filter add dev eth0 parent ffff: protocol ip u32 match ip dst <multicast_group_n> action ok</pre>	

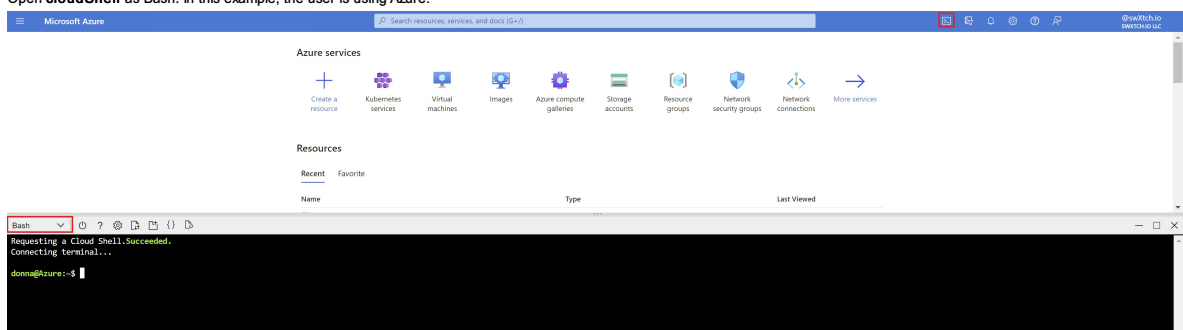
Example: allowing incoming traffic **ONLY** to multicast group `239.0.0.1` :

Bash	Copy
<pre>tc qdisc add dev eth0 ingress tc qdisc add dev eth0 parent ffff: protocol ip u32 match ip dst 224.0.0.0/4 action drop tc filter add dev eth0 parent ffff: protocol ip u32 match ip dst 239.0.0.1/32 action ok</pre>	

Getting a shell to an xNIC DaemonSet pod

At times, it is nice to be able to get into the pod and be able to run commands such as `swtch-tcpdump`. To accomplish this, follow these steps:

1. Sign into your desired cloud.
2. Open cloudShell as Bash. In this example, the user is using Azure.



3. Enter in the following command to get the pod name:

Bash	Copy
<pre>kubectl get pods -l app=swtch-xnic -n kube-system</pre>	

Example:

Bash	Copy
<pre>user@Azure:~\$ kubectl get pods -l app=swtch-xnic -n kube-system NAME READY STATUS RESTARTS AGE swtch-xnic-fc58t 1/1 Running 0 11d swtch-xnic-kn9hg 1/1 Running 0 11d</pre>	

4. Enter in the following command, replacing **Pod** with the pod name:

Bash	Copy
<pre>kubectl exec -it pod/swtch-xnic-name -n kube-system -- bash</pre>	

Example:

Bash	Copy
<pre>user@Azure:~\$ kubectl exec -it pod/swxtch-xnic-kn9hg -n kube-system -- bash root@aks-nodepool11-23164585-vmss00000A:/</pre>	

You can now enter in commands similar to any VM, such as `ip a` or `sudo swxtch-tcpdump -i eth0`. Note that the pods created in this example do not have tools such as the standard `tcpdump`. However, `swxtch-tcpdump` will work. For testing, see `swxtch-perf` under [Testing cloudSwXtch](#).

Switching Contexts

If you have more than one AKS Kubernetes cluster, then you may need to change the context to work on the desired instance. For more information, please review the [Changing K8s Context in Your Preferred Cloud](#) section.

Accessing xNIC Logs

You can get xNIC logs once signed in to the pod. See [How to Find xNIC Logs](#) and follow directions for xNIC.

Using xNIC config

Getting to the xNIC config is available once you're signed into the Pod. To get to the xNIC config, use the command below:

Bash	Copy
<pre>cat /var/opt/swxtch/swxtch-xnic.conf</pre>	

Exiting the Pod

To exit the pod, enter in the following command:

Bash	Copy
<pre>exit</pre>	

To Change K8s Context in Your Preferred Cloud

If there are more than one K8s clusters in your preferred cloud, then you may need to switch between them to run commands in the CloudShell Bash. Below are steps to switch between K8s clusters.

1. Get a list of all K8s Contexts by using the following command:

Bash	Copy
<pre>kubectl config get-contexts</pre>	

Example in Azure:

Bash

Copy

user@Azure:~\$ kubectl config get-contexts

CURRENT	NAME	CLUSTER	AUTHINFO	NAMESPACE
	cilium-sample	cilium-sample	clusterUser_saDevNetwork_cilium-sample	
	cilium-sample-200	cilium-sample-200	clusterUser_test-donna-200-rg_cilium-sample-200	
	cilium-sample2	cilium-sample2	clusterUser_saDevNetwork_cilium-sample2	
*	cilium-sample300	cilium-sample300	clusterUser_test-donna-300-rg_cilium-sample300	
	dsd-k8-cluster-100	dsd-k8-cluster-100	clusterUser_saDevNetwork_dsd-k8-cluster-100	

Notice in the above list there are multiple context but only one has the asterisks (*). The asterisks marks what is the default context.

2. To change context, run the following command. The example is changing to cilium-sample2.

Bash	Copy
<pre>kubectl config use-context cilium-sample2</pre>	

3. Re-run the `get-context` command:

Bash	Copy
<pre>kubectl config get-contexts</pre>	

Example in Azure:

Bash

Copy

```
user@Azure:~$ kubectl config get-contexts
```

CURRENT	NAME	CLUSTER	AUTHINFO	NAMESPACE
	cilium-sample	cilium-sample	clusterUser_saDevNetwork_cilium-sample	
	cilium-sample-200	cilium-sample-200	clusterUser_test-donna-200-rg_cilium-sample-200	
*	cilium-sample2	cilium-sample2	clusterUser_saDevNetwork_cilium-sample2	
	cilium-sample300	cilium-sample300	clusterUser_test-donna-300-rg_cilium-sample300	
	dsd-k8-cluster-100	dsd-k8-cluster-100	clusterUser_saDevNetwork_dsd-k8-cluster-100	

As you can see above, the asterisk (*) has changed positions to the desired context, cilium-sample2.

Uninstall xNIC DaemonSet on K8s

WHAT TO EXPECT

In this article, users will learn how to uninstall xNIC DaemonSet on Kubernetes (K8s).

To uninstall xNIC DaemonSet on K8s, please follow these steps:

- 1. Sign into **Cloud**.
- 2. Open **cloudShell** as Bash.
- 3. Run the following command in the terminal:

Shell



Bash	Copy
<pre>./xnic_ds_installer.sh -u</pre>	

- 4. xNIC DaemonSet on K8s should now be uninstalled.

Test xNIC with K8s

WHAT TO EXPECT

Before running your application in your preferred cloud, it is a good idea to test with swXtch.io's provided tools/examples.

In this article, you will learn how to test xNIC with K8s. Please complete the installation process outlined in [Install xNIC on K8s](#) before you begin testing.

Prerequisites

For this test to work, a user should have at least two nodes.

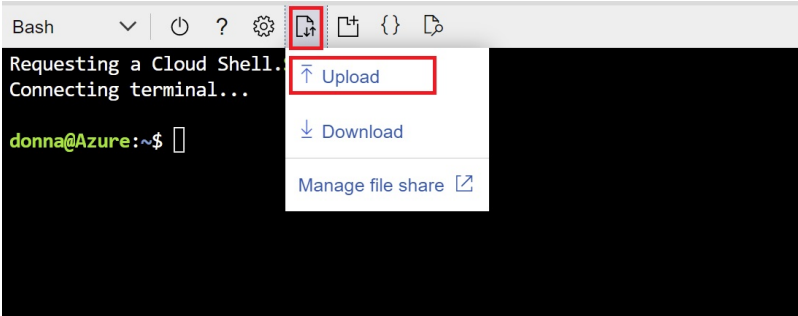
STEP ONE: Create A Consumer

- 1. Create a `TestConsumer.yaml` file using the example below.
- Replace the `XNIC_SWXTCH_ADDR` with the cloudSwXtch control address.

BashCopy

```
apiVersion: v1
kind: Pod
metadata:
  name: consumer-a
  labels:
    app: consumer-a
spec:
  affinity:
    podAntiAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        - labelSelector:
            matchExpressions:
              - key: app
                operator: In
                values:
                  - producer-a
                  - consumer-b
          topologyKey: kubernetes.io/hostname
  containers:
    - name: consumer-a
      image: ubuntu:20.04
      securityContext:
        privileged: true
      env:
        - name: IS_DAEMON
          value: "false"
        - name: PERF_TYPE
          value: "consumer"
        - name: PERF_NIC
          value: "eth0"
        - name: PERF_MCGIP
          value: "239.0.0.10"
        - name: PERF_MCGPORT
          value: "8410"
        - name: XNIC_SWXTCH_ADDR
          value: "10.224.0.115"
      command: ["/bin/bash"]
      args: ["-c", "apt update && apt install curl -y;
        curl http://$(XNIC_SWXTCH_ADDR)/services/install/swxtch-xnic-k8s-install.sh --output swxtch-xnic-k8s-install.sh;
        chmod +x swxtch-xnic-k8s-install.sh;
        ./swxtch-xnic-k8s-install.sh -v 2;
        sleep infinity"]
```

- 2. Upload the file into the Azure CloudShell.



STEP TWO: Create a Producer

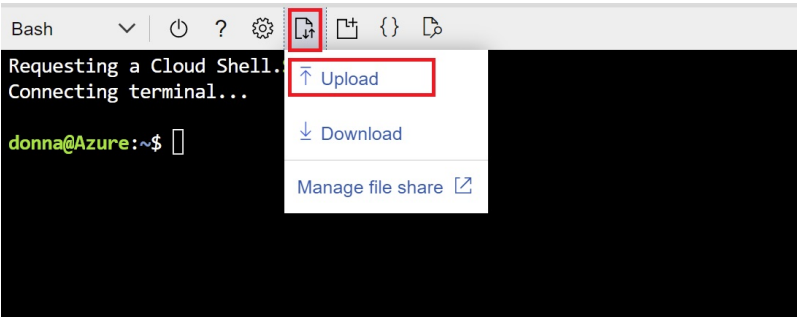
- 1. Create a `TestProducer.yaml` file using the example below.
- Replace `XNIC_SWXTCH_ADDR` with the cloudSwXtch control address.

Shell

Bash	Copy
<pre>apiVersion: v1 kind: Pod metadata: name: producer-a labels: app: producer-a spec: affinity: podAntiAffinity: requiredDuringSchedulingIgnoredDuringExecution: - labelSelector: matchExpressions: - key: app operator: In values: - consumer-a - producer-b topologyKey: kubernetes.io/hostname containers: - name: producer-a image: ubuntu:20.04 securityContext: privileged: true env: - name: IS_DAEMON value: "false" - name: PERF_TYPE value: "producer" - name: PERF_NIC value: "eth0" - name: PERF_MCGIP value: "239.0.0.10" - name: PERF_MCGPORT value: "8410" - name: PERF_PPS value: "100" - name: XNIC_SWXTCH_ADDR value: "10.224.0.115" command: ["/bin/bash"] args: ["-c", "apt update && apt install curl -y; curl http://\$(XNIC_SWXTCH_ADDR)/services/install/swxtch-xnic-k8s-install.sh --output swxtch-xnic-k8s-install.sh; chmod +x swxtch-xnic-k8s-install.sh; ./swxtch-xnic-k8s-install.sh -v 2; sleep infinity"]</pre>	

2. Upload the file into the Azure CloudShell.



STEP THREE: Run Test

- Run the producer by running this command in your preferred cloud's cloudShell Bash window.
Wait for the cursor to return to know it is fully created.

Shell

Bash	Copy
<pre>kubectl create -f TestProducer.yaml</pre>	

- Run the consumer by running this command in your preferred cloud's cloudShell bash window.
Wait for the cursor to return to know it is fully created.

Bash	Copy
<pre>kubectl create -f TestConsumer.yaml</pre>	

Validate they are running using this command:

Bash	Copy
<pre>kubectl get pods -o wide -A</pre>	

Below is an example in Azure showing the consumer-a and producer-a running:

Bash								Copy
<pre> donna@Azure:~\$ kubectl get pods -o wide -A NAMESPACE NAME READY STATUS RESTARTS AGE IP NODE NOMINATED NODE READINESS GATES kube-system consumer-a 1/1 Running 0 15m 10.0.1.91 aks-nodepool1-23351669-vmss000006 <none> <none> kube-system producer-a 1/1 Running 0 15m 10.0.1.90 aks-nodepool1-23351669-vmss000005 <none> <none> kube-system cilium-node-init-kbql4 1/1 Running 0 27h 10.2.128.101 aks-nodepool1-23164585-vmss00000j <none> <none> kube-system cilium-node-init-sg4vc 1/1 Running 0 27h 10.2.128.100 aks-nodepool1-23164585-vmss00000i <none> <none> kube-system cilium-nx7vl 1/1 Running 0 27h 10.2.128.100 aks-nodepool1-23164585-vmss00000i <none> <none> kube-system cilium-operator-6485c89c66-748tz 1/1 Running 0 27h 10.2.128.101 aks-nodepool1-23164585-vmss00000j <none> <none> kube-system cilium-vv4qs 1/1 Running 0 27h 10.2.128.101 aks-nodepool1-23164585-vmss00000j <none> <none> kube-system cloud-node-manager-mncgk 1/1 Running 0 27h 10.2.128.100 aks-nodepool1-23164585-vmss00000i <none> <none> kube-system cloud-node-manager-qg5wf 1/1 Running 0 27h 10.2.128.101 aks-nodepool1-23164585-vmss00000j <none> <none> kube-system coredns-autoscaler-569f6ff56-qtqpr 1/1 Running 0 28h 10.0.0.121 aks-nodepool1-23164585-vmss00000i <none> <none> kube-system coredns-fb6b9d95f-blk6j 1/1 Running 0 28h 10.0.0.236 aks-nodepool1-23164585-vmss00000i <none> <none> kube-system coredns-fb6b9d95f-pxzh2 1/1 Running 0 28h 10.0.0.131 aks-nodepool1-23164585-vmss00000i <none> <none> </pre>								

Step Four: Validate The Test Is Running

You can validate it is working by viewing logs with this command:

Bash								Copy
<pre> kubectl logs pods/producer-a -f </pre>								

2nd screen

Alternatively, you can log into your cloudSwXtch and run this command to see data flowing between nodes:

Bash								Copy
<pre> swxtch-top </pre>								

swXtch-top should show the traffic coming in and out of the nodes with either the producer or the consumer.

<pre> Information - dev.3645.v2 dsd.core-100 dev.3645.v2(dsd-100-core-azure-April-5-2033) Auth. Type License File Marketplace SubscriptionId b10209ad-ad22-4c26-8aef-be93b2f0bb58 Max Bandwidth 2000 Mbps ResourceGroupName TEST-DONNA Max Clients 10 SwxtchId 2369a922-410a-40bf-8e6e-630efe0ee70a Status OK </pre>							
<pre> Totals Producers 99 pps (136.7K bps) Consumers 99 pps (137.6K bps) Bridge RX 0 pps (0 bps) Bridge TX 0 pps (0 bps) Mesh RX 0 pps (0 bps) Mesh TX 0 pps (0 bps) Switch RX 100 pps (139.1K bps) Switch TX 99 pps (137.7K bps) </pre>							
<pre> xNIC clients Name ip Version (xNIC) RX pps RX bps TX pps TX bps HA aks-nodepool1-23351669-vmss000004 10.2.128.95 dev.3645.v2 (v2) 99 137.6K 0 0 N aks-nodepool1-23351669-vmss000005 10.2.128.96 dev.3645.v2 (v2) 0 0 99 136.7K N </pre>							

Step Five: Cleaning the Pods

- Stop the test consumer by running this command back in your preferred cloud's CloudShell bash window.
 - Wait for the cursor to return to know it is deleted fully.

Bash								Copy
<pre> kubectl delete -f TestConsumer.yaml </pre>								

- swXtch-top should no longer show the consumer. Additionally, running `kubectl get pods -o wide` should now show just the test consumer as shown below.

Bash								Copy
<pre> donna@Azure:~\$ kubectl get pods -o wide -A NAME READY STATUS RESTARTS AGE IP NODE GATES consumer-a 1/1 Terminating 0 15m 10.0.1.91 aks-nodepool1-23351669-vmss000006 producer-a 1/1 Running 0 15m 10.0.1.90 aks-nodepool1-23351669-vmss000005 swxtch-xnic-46qgg 1/1 Running 0 39m 10.2.128.96 aks-nodepool1-23351669-vmss000005 swxtch-xnic-szdk7 1/1 Running 0 40m 10.2.128.95 aks-nodepool1-23351669-vmss000004 </pre>								

2. Stop the test producer by running this command in your preferred cloud's CloudShell bash window.
- a. Wait for the cursor to return to know its fully deleted.

Bash

Copy

```
kubect1 delete -f TestProducer.yaml
```

- b. swXtch-top should no longer show the producer. This may take a minute to display. Additionally, running **kubect1 get pods -o wide** should now show just the test producer as shown below:

Bash

Copy

```
donna@Azure:~$ kubect1 get pods -o wide -A
NAME                READY   STATUS    RESTARTS   AGE   IP            NODE                                           NOMINATED NODE   READINESS
GATES
producer-a          1/1     Terminating    0         15m   10.0.1.90     aks-nodepool11-23351669-vmss000005          <none>            <none>
swxtch-xnic-46qgg   1/1     Running         0         42m   10.2.128.96   aks-nodepool11-23351669-vmss000005          <none>            <none>
swxtch-xnic-szdk7   1/1     Running         0         42m   10.2.128.95   aks-nodepool11-23351669-vmss000004          <none>            <none>
```

Now that the system is validated using swXtch.io, you can test with your K8s application.

Upgrade xNIC nodes on K8s

WHAT TO EXPECT

The nodes upgrade is automatic based on the restart of the nodes containing the xNIC.
In this article, you will learn how you can use this method to upgrade your xNIC nodes on K8s to match the version of your cloudSwXtch.

Before you upgrade the xNIC nodes on K8s, you need to upgrade the cloudSwXtch to the latest version. See [Upgrading cloudSwXtch](#) for more information

Restarting the xNIC DaemonSet

- 1. Sign into your preferred cloud's portal.
- 2. Open cloudShell as Bash.
- 3. Run the following command:

Bash	Copy
<code>kubectl rollout restart ds swxtch-xnic -n kube-system</code>	

- a. This will restart the Kubernetes swxtch-xnic DaemonSet and update the version of the xNIC to match that of the cloudSwXtch.

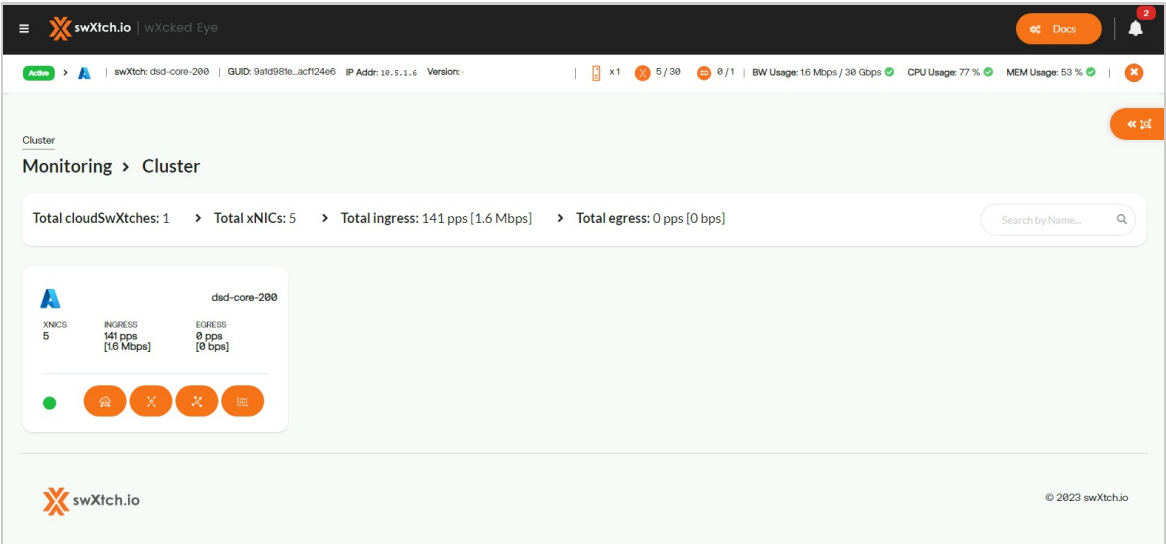
Using wXcked Eye for cloudSwXtch

WHAT TO EXPECT

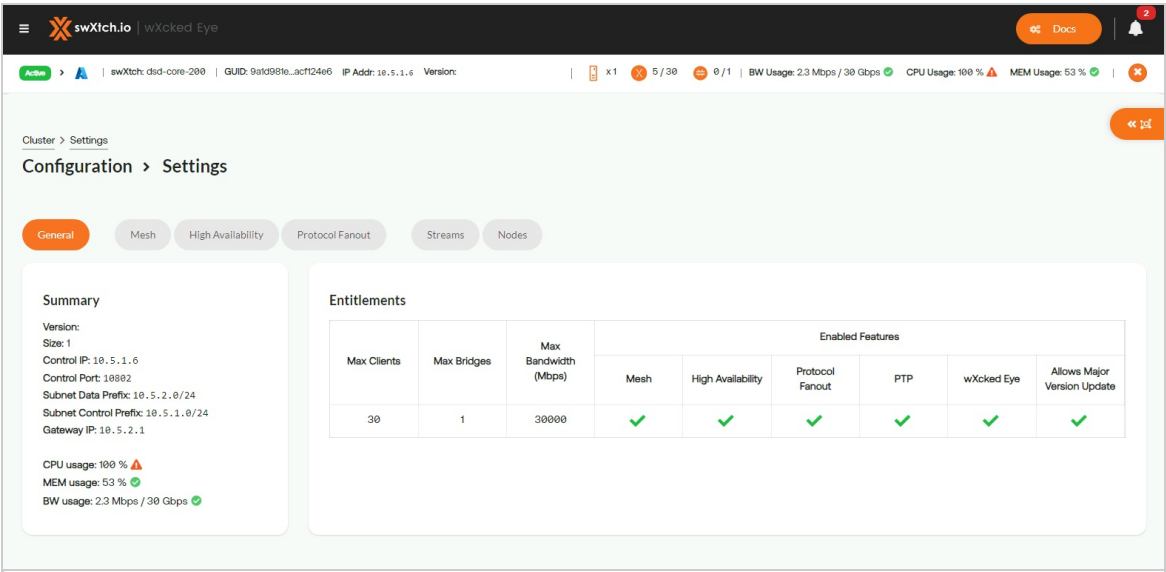
In this article, you will learn more about wXcked Eye and the benefits of using it to configure and monitor your cloudSwXtch environment.

What is wXcked Eye?

wXcked Eye is a web-based monitoring and configuration tool for cloudSwXtch. It presents users with a high-level view of their cloudSwXtch environment with an interactive graph detailing connections to different endpoints. With an expansive look at performance metrics, users can ensure that their data is flowing as expected.



In addition, wXcked Eye unlocks the ability to configure Mesh, High Availability, Protocol Fanout and Conversion, and Precision Time Protocol (PTP) from the comfort of a web browser.



How to Access wXcked Eye

To access the wXcked Eye UI, users will need to enter the following URL into a web browser of a VM in the cloudswXtch environment. They should use the IP address of their cloudSwXtch to prefix the URL.

```
Bash
```

```
<swxtch-ip-address>/wxckedeye/
```

Copy

Monitor cloudSwXtch with wXcked Eye

To learn more about the monitoring capabilities of wXcked Eye such as the Network Graph or the cloudSwXtch and xNIC metrics views, see [Monitor cloudSwXtch with wXcked Eye](#).

Configure cloudSwXtch with wXcked Eye

To learn more about the configuration capabilities of wXcked Eye such as Mesh, High Availability, Protocol Fanout and Precision Time Protocol, see [Configure cloudSwXtch with wXcked Eye](#).

Monitor cloudSwXtch with wXcked Eye

WHAT TO EXPECT

The **wXcked Eye** UI provides users with an additional way to monitor the performance of their cloudSwXtch network.

To learn more about how to configure your cloudSwXtch for high availability and protocol fanout with wXcked eye, please read the ["Configure cloudSwXtch with wXcked Eye"](#) article.

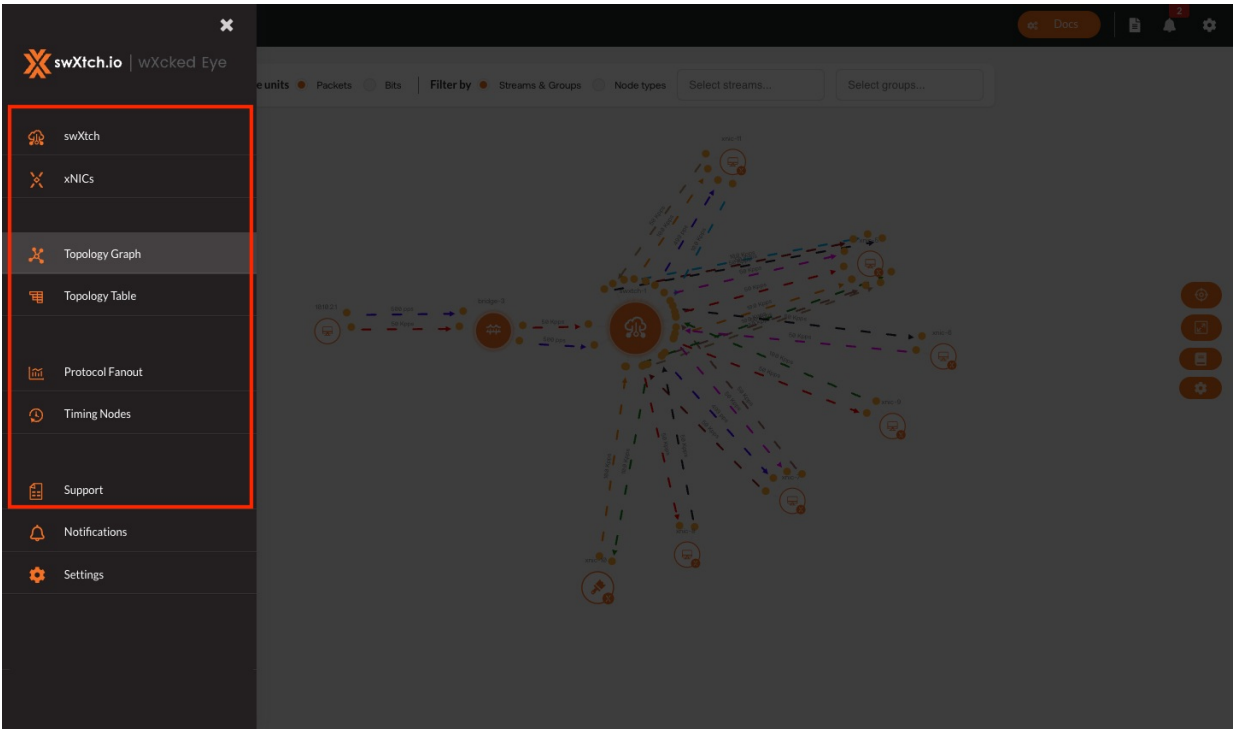
In this section, we will walk through the user interface, explaining overall functionality and how it provides users with additional control over their cloudSwXtch network.

Accessing the wXcked Eye UI

To access the wXcked Eye UI, users will need to enter the following URL into a web browser of a VM in their cloudswXtch environment. They should use the IP address of their cloudSwXtch to prefix the URL.

Plaintext	Copy
<cloudSwXtch-ip-address>/wxckedeye/	

Navigating the Monitoring pages



The wXcked Eye’s monitoring capabilities are organized into six pages. For more information on a page’s contents, please view their respective articles.

- [cloudSwXtch Stats](#)
- [xNICs Stats](#)
- [Topology](#)
- [Protocol Fanout Stats](#)
- [Timing Nodes](#)
- [Support](#)

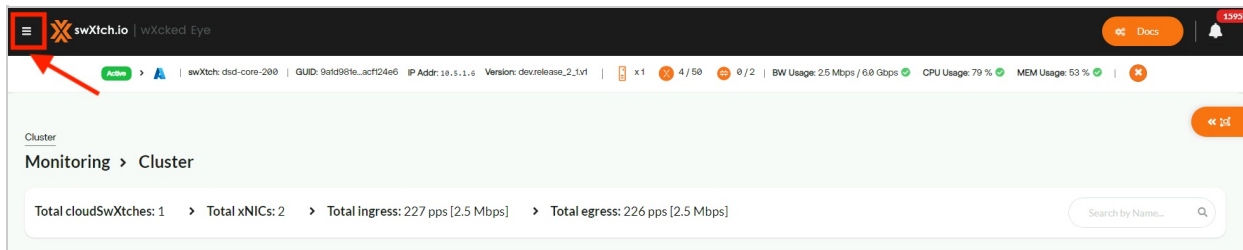
wXcked Eye cloudSwXtch Page

WHAT TO EXPECT

In this article, you will learn how to view the performance metrics for your cloudSwXtch and the xNICs associated with it.

Locating the wXcked Eye cloudSwXtch Page

To navigate to the cloudSwXtch page, users will need to click on the menu (≡) option at the top left hand corner by the swXtch.io logo.

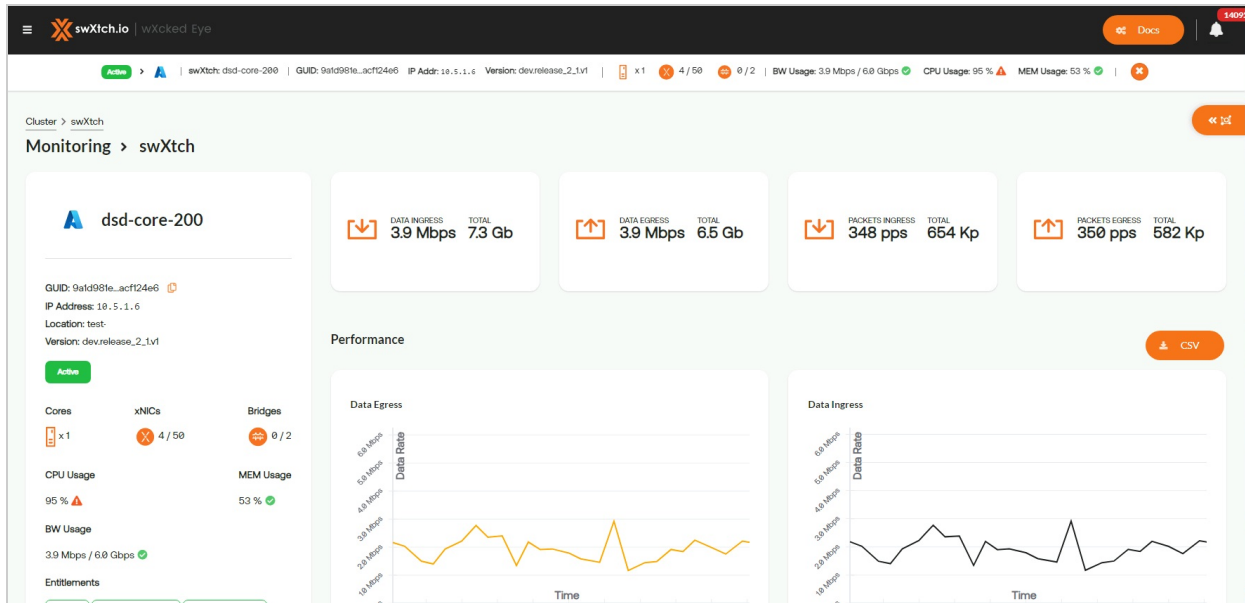


From there, select swXtch.

Alternatively, if a user is on the Cluster page, they can select the cloudSwXtch Stats button in a cloudSwXtch's Information card.



Navigating the wXcked Eye cloudSwXtch Page

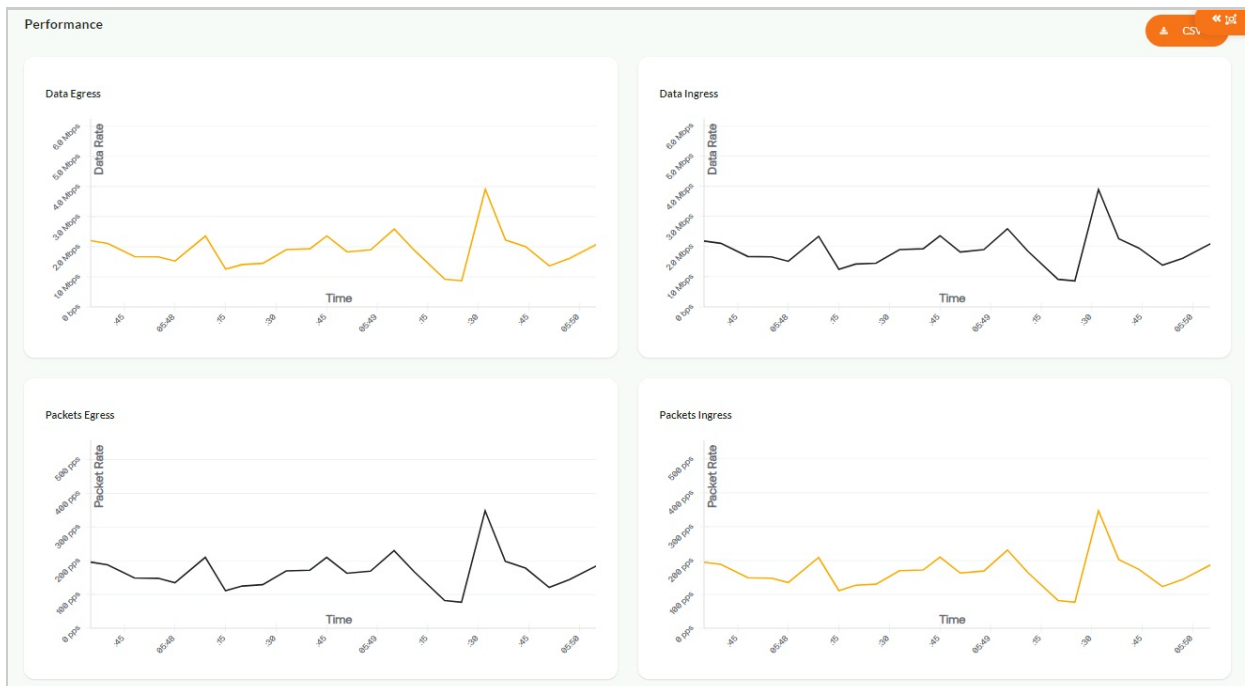


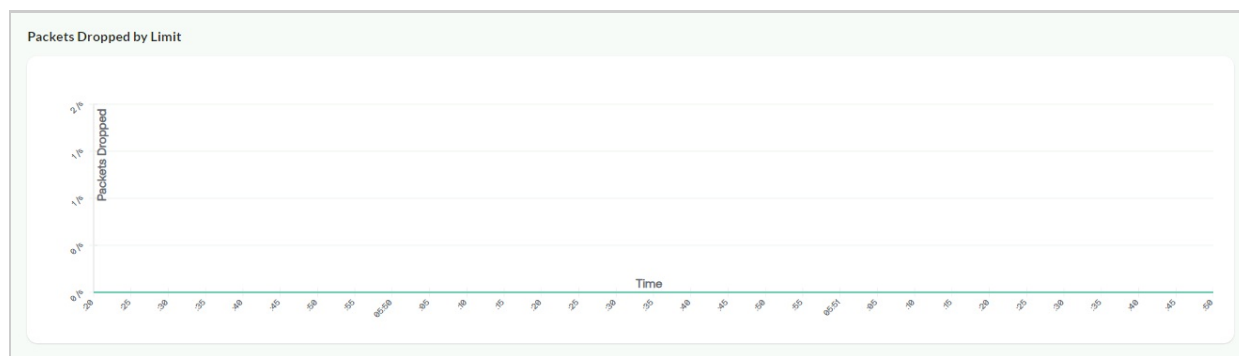
cloudSwXtch Key Performance Metrics

Once the page loads, users will be presented a high-level view of their selected cloudSwXtch's data flow. This page provides detailed information regarding the cloudSwXtch and illustrates 4 key performance metrics:

- Data Egress
- Data Ingress
- Packets Egress
- Packets Ingress

Data egress/ingress are displayed in bits per second (bps) while packets egress/ingress are displayed in packets per second (pps). In addition to the rate, the total number of bits and packets are displayed for the user. These metrics are further explored in the **Performance** section with four related graphs and an additional Packets Dropped graph.





cloudSwXtch Information Panel

ip-172-52-137-127

GUID: i-03a389...26cbf76e
IP Address: 172.52.137.127
Location: 63972066639/us-west-2
Version: dev.44af38

Active

Cores
 x 4

xNICs
 5 / 50

Bridges
 0 / 1

BW Usage

1.9 Mbps / 50 Gbps

Entitlements

MESH

PROTOCOL FANOUT

HIGH AVAILABILITY

PTP

BRIDGE

WICKED EYE

MAJOR VERSION UPDATE

TACHYON LIVE

Important network information of the cloudSwXtch such as the **GUID**, **IP address**, **location (resource groups)**, **cloud provider**, **version**, & **replicator status** is shown in the top left card along with its name, which in this case is core-200. In addition, the **number of cores** and **number of associated xNICs** to the cloudSwXtch will also be displayed.

Bandwidth usage is also listed. In the event that a cloudSwXtch exceeds its allotted bandwidth, a warning symbol will appear.

xNICs Panel

xNICs (4)

Search by Name...

Name	Version	Ingress 172 Gb [17 Mp]	Egress 47 Gb [4.2 Mp]	Drops 0
▼ 172.41.128.113		5.1 Mbps [451 pps]	0 bps [0 pps]	0 /s
Multicast Groups				
IP: 224 . 2 . 2 . 2		Ingress: 0 bps [0 pps]	Egress: 0 bps [0 pps]	
IP: 239 . 255 . 255 . 250		Ingress: 0 bps [0 pps]	Egress: 0 bps [0 pps]	
► 172.41.129.42		1.6 Mbps [11 Kpps]	4.9 Mbps [440 pps]	0 /s
▼ 172.52.132.19		4.8 Mbps [430 pps]	0 bps [0 pps]	0 /s

At the bottom of the cloudSwXtch Stats page, users will be able to see the **xNICs** panel. This panel lists the agents that are connected to the cloudSwXtch. Each listed xNIC is accompanied with its version, ingress/egress rates, and packet drops. When using the dropdown feature for an agent, a user can see the multicast groups associated with the xNIC and its ingress/egress rates.

wXcked Eye xNICs Page

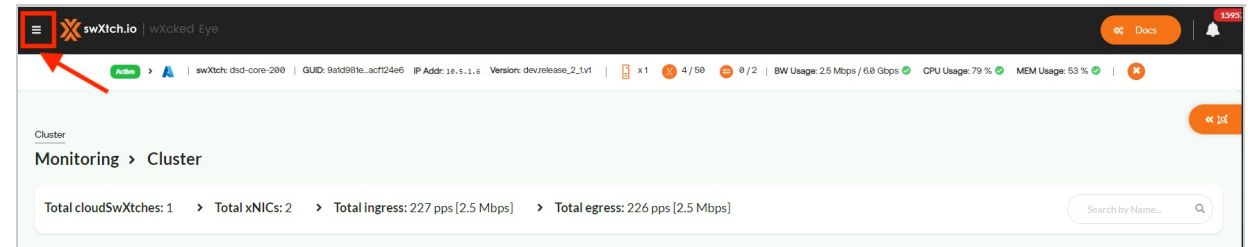
WHAT TO EXPECT

In this article , users will learn how to view performance metrics from the xNICs perspective.

Locating the xNIC View Page

The xNIC page provides users with a look of their cloudSwXtch environment from the xNICs' perspective, breaking down performance at an agent's level.

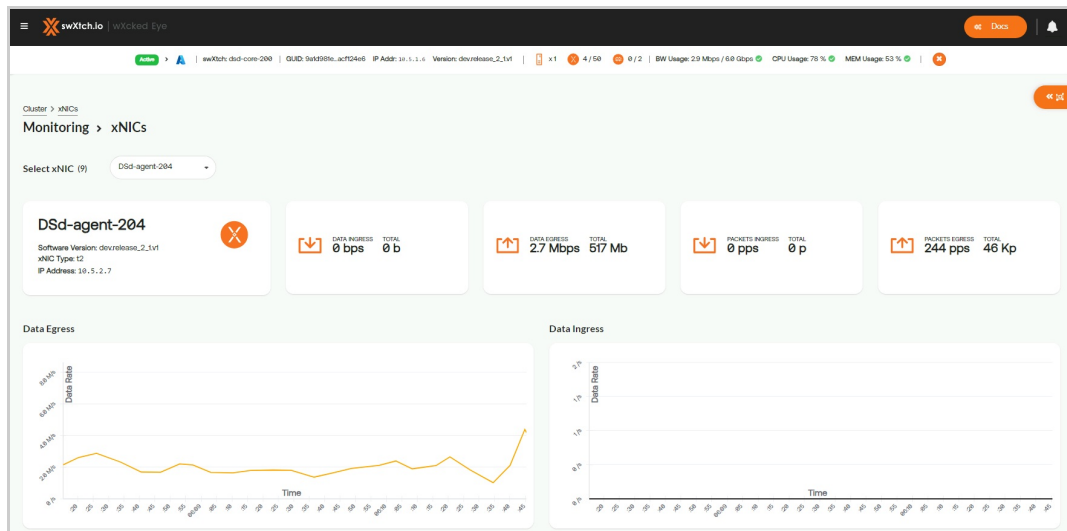
To navigate to the xNIC page, users will need to click on the menu option at the top left hand corner by the swXtch.io logo.



The navigation menu will open, revealing the other wXcked Eye pages. Select xNICs to view the xNIC page.



Navigating the xNIC page



At first glance, the xNIC page looks very similar to the [cloudSwtch view](#). However, instead of focusing on the cloudSwtch, users are given key information and performance metrics for a single xNIC.

- **Data Ingress (bps)** - Data being consumed by the xNIC
- **Data Egress (bps)** - Data being sent from the xNIC
- **Packets Ingress (pps)** - Packets being consumed by the xNIC
- **Packets Egress (pps)** - Packets being sent from the xNIC

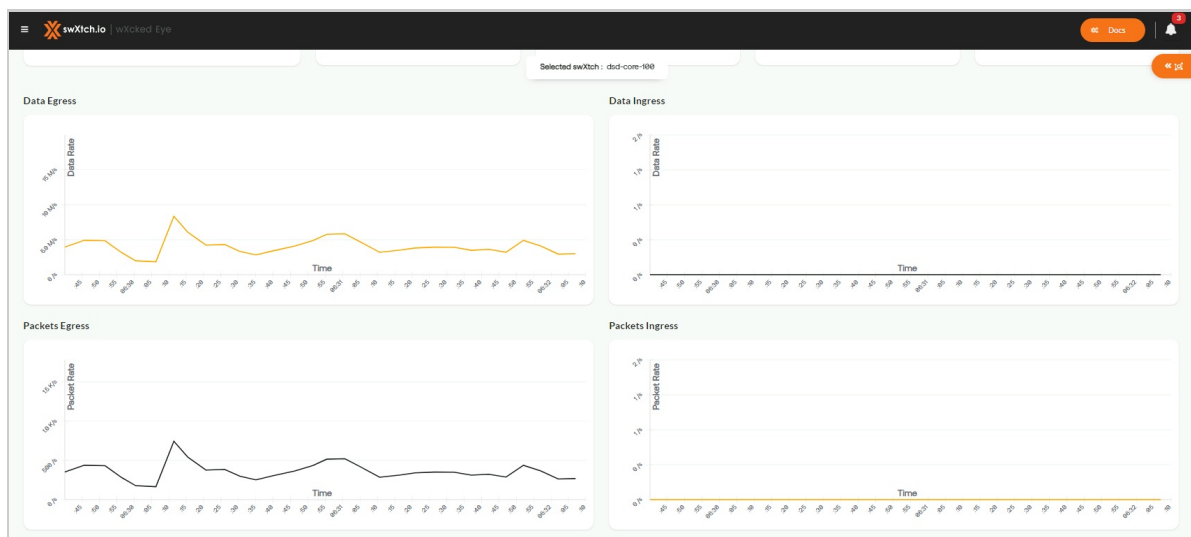
In the example above, one noticeable difference is the inclusion of the Select an xNIC dropdown menu next to "Select xNIC." Here, a user can select an agent they wish to monitor (Dsd-agent-204).

After selecting an xNIC, the agent's information will display in the same area as the cloudSwtch on the main page. The information includes the software version, xNIC version and the IP address.

Just like the xNIC panel in the wXcked Eye main page, users are able to see the Multicast Groups associated with the xNIC and their ingress/egress rates.

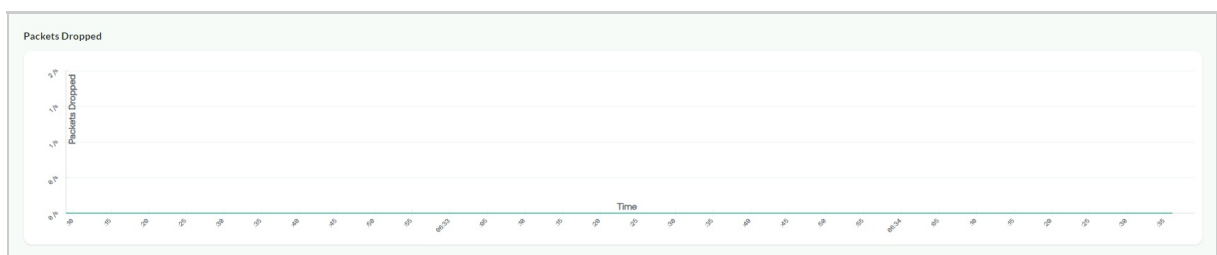
Performance

The xNIC view provides users with another way to visualize data flow. Towards the bottom of the page, users will be able to see the 5 key performance metrics displayed as active histograms. The first four deal with data and packet egress/ingress over 15 second increments.



Performance: Data Egress/Ingress and Packets Egress/Ingress

The bottom graph displays the number of packets dropped over time. A successful stream would show no packets dropping like the example below. The X-Axis is organized into 5 second increments.



Packets Dropped by Limit (:05 second increments)


Multicast Groups

The Multicast Groups panel lists the IP addresses of different data streams related to the cloudSwXtch with the ingress/egress rates displayed.

Multicast Groups (5)

Search by IP...

IP Address	Ingress 149 Mb [75 p]	Egress 40 Gb [3.4 Mbps]
10.2.131.255	0 bps [0 pps]	0 bps [0 pps]
224.0.0.251	0 bps [0 pps]	0 bps [0 pps]
10.2.195.255	0 bps [0 pps]	0 bps [0 pps]
239.255.255.250	0 bps [0 pps]	0 bps [0 pps]
239.1.1.4	0 bps [0 pps]	66 Gbps [574 Kpps]

swXtch.io

© 2023 swXtch.io

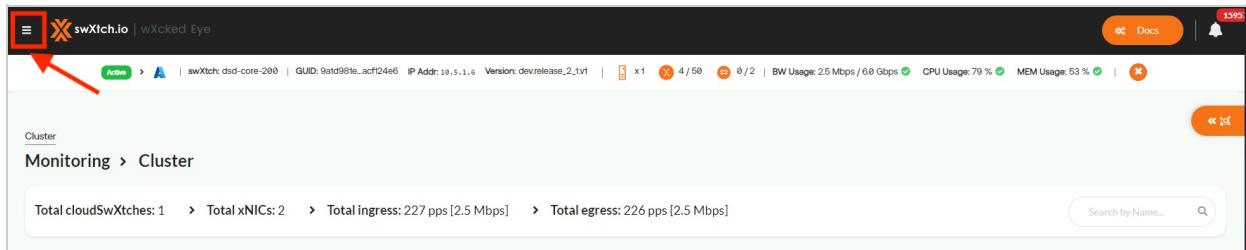
wXcked Eye Topology Graph

WHAT TO EXPECT

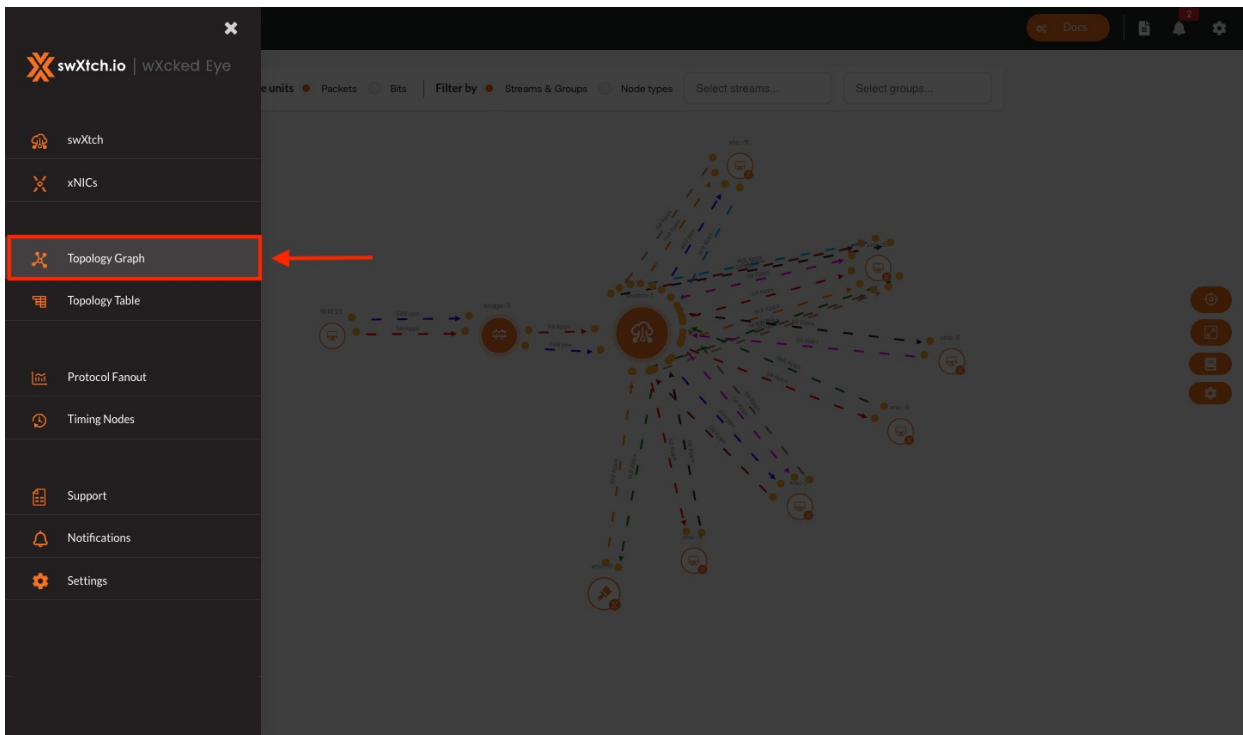
In this article, users will learn how to use the wXcked Eye Topology Graph and how to reformat it for their needs.

Locating the wXcked Eye Topology

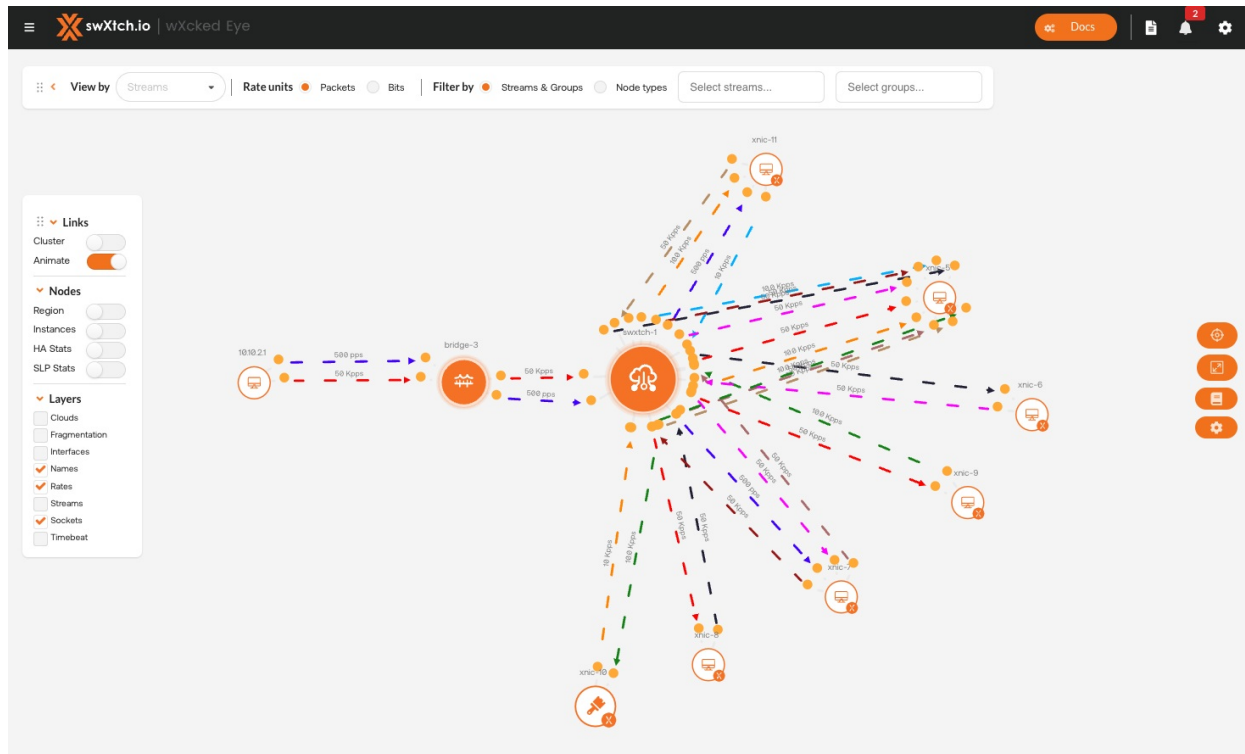
To navigate to the wXcked Eye Topology Graph page, users will need to click on the menu (≡) option at the top right hand corner by the swXtch.io logo.



From there, select **Topology Graph**.



Using the wXcked Eye Topology



The **wXcked Eye Topology Graph** page displays a network graph, providing a high level view of the cloudSwXtch environment. The center of the graph will display the cloudSwXtch you are currently on with user-assigned colored lines indicating traffic flowing either to or from it. Next to each line, users will be able to see the flow's transmission rates (either in pps or bps). The endpoints can be either xNICs, cloudSwXtch Bridges or other cloudSwXtch instances.

Reformatting the Topology Graph

It is very simple to alter the Topology Graph for a user's desired configuration. In addition to being able to physically drag and rearrange the icons and filtering panels in the graph, users can zoom in and out, refresh the page, and toggle labels on and off.

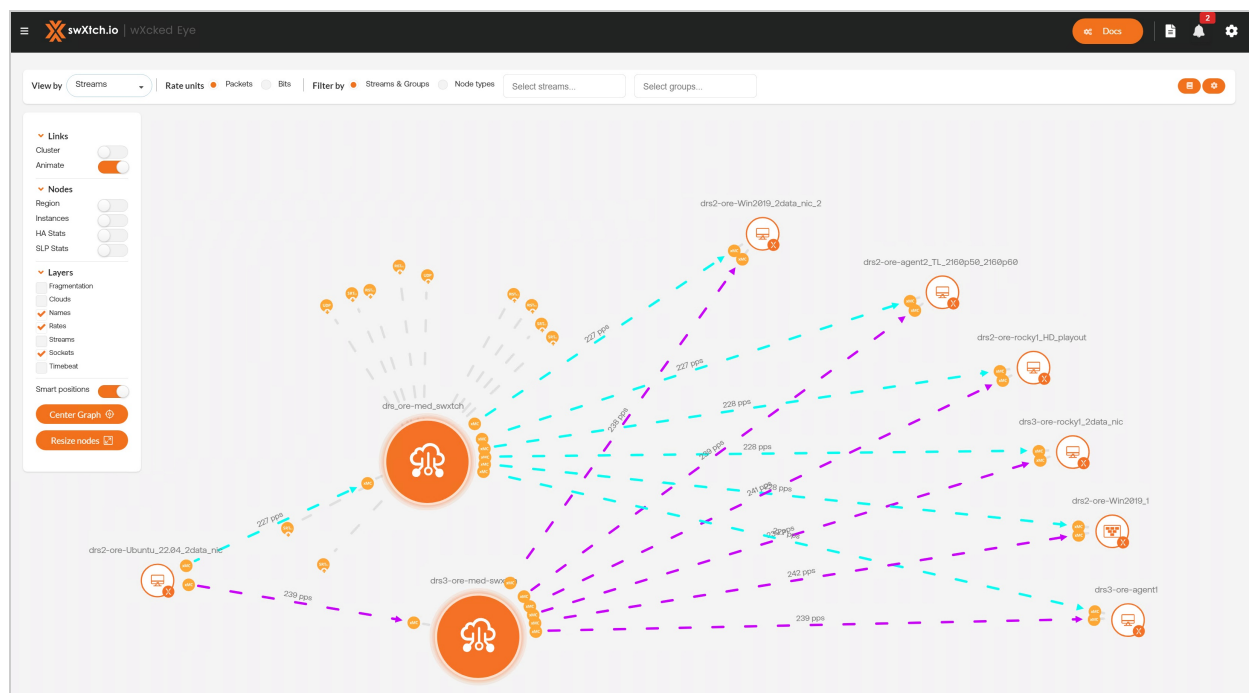
In both the top **View By/Filter By Bar** and the **Links, Nodes, Layers** panel, users can find several options to alter the appearance of their topology graph.

View By/Filter By Bar

There are three views available for users: **Streams**, **Channels** and **HA**.

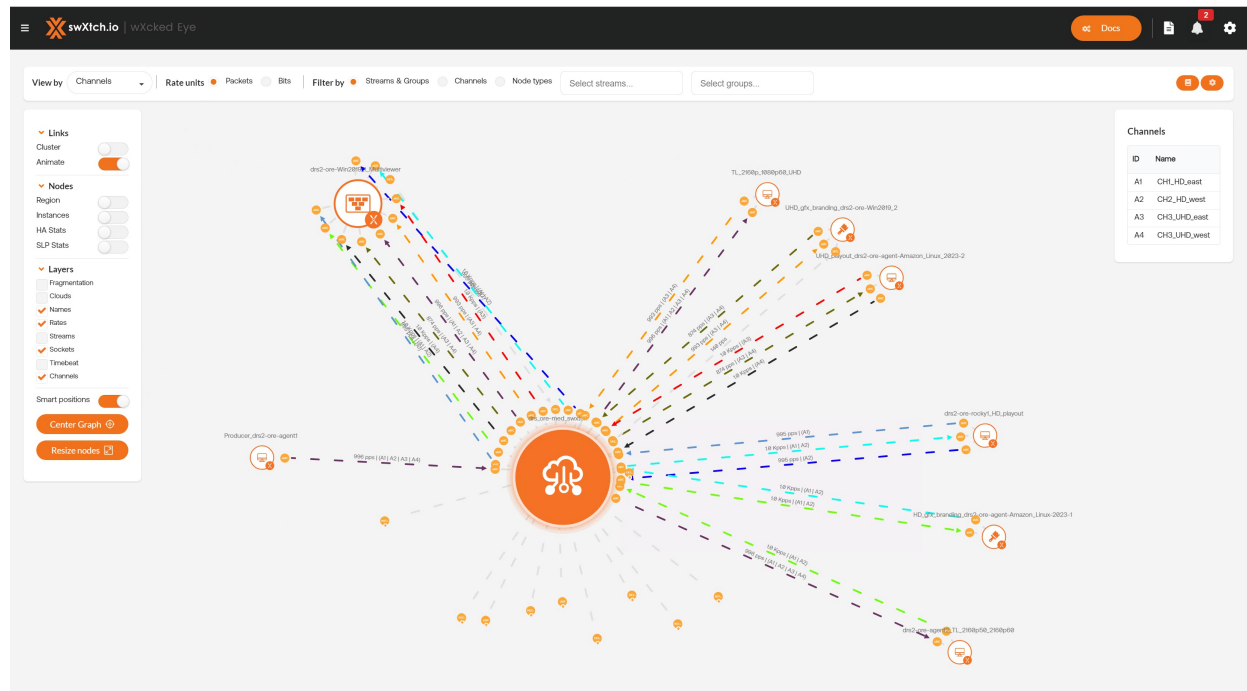
View by Streams

The **Streams** view provides users with a high level view of all the streams associated with the cloudSwXtch.



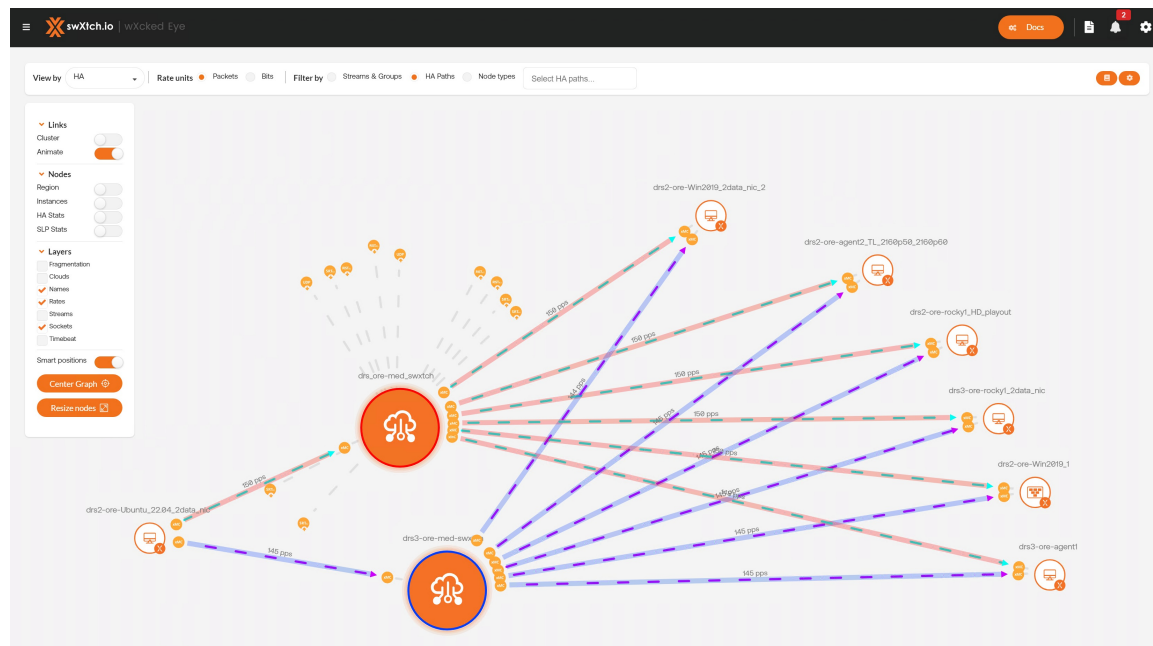
View by Channels

The Channels view highlights streams associated with a specific channel distinction. Each relevant line will have a Channel ID attached to it. A Channels key will also appear.



View by High Availability (HA)

Finally, the HA view highlights the multiple paths in an HA configuration.



Streams, Channels and HA links can be formatted in the wXcked Eye [Settings](#) page.

Rates units

For Rates, users can select between Packets and Bits for their unit.

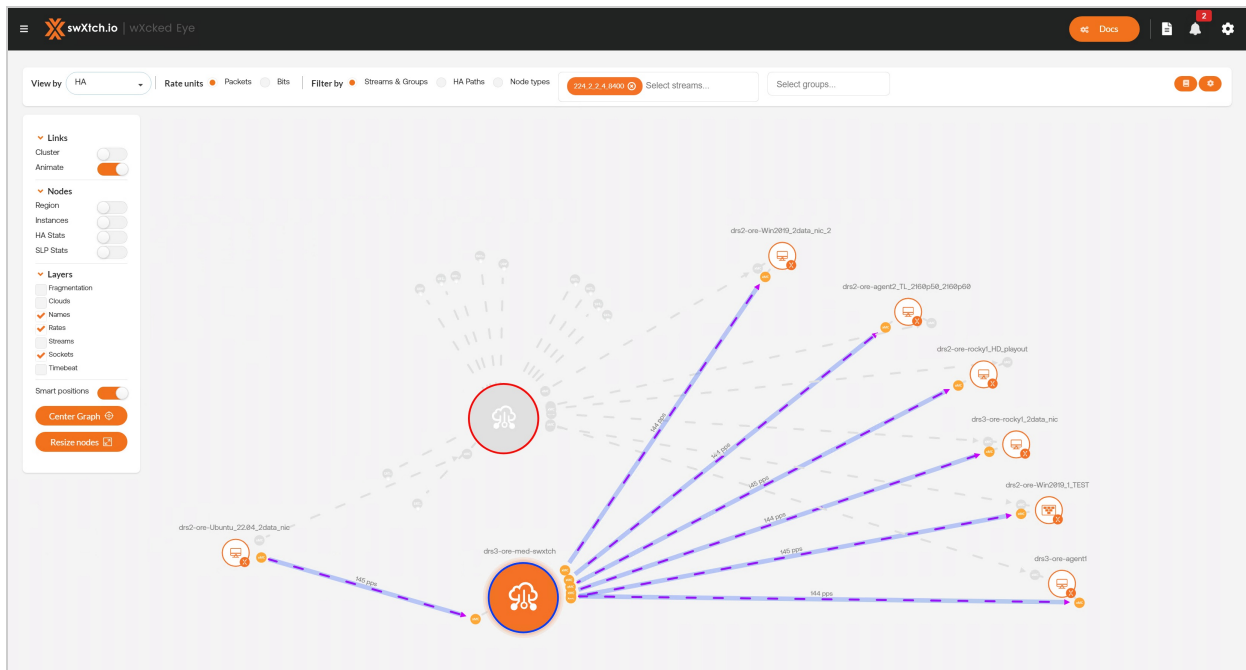
Filter by

Depending on the View selected (Streams, Channels, and HA), the Filter By options will change reflecting that choice. The two main options, Streams & Groups and Node Types, will appear for all view types. Choosing either category will alter the subsequent "Select" prompts. Here, users can specify the streams, groups, channels, or node types they wish to see in the graph.

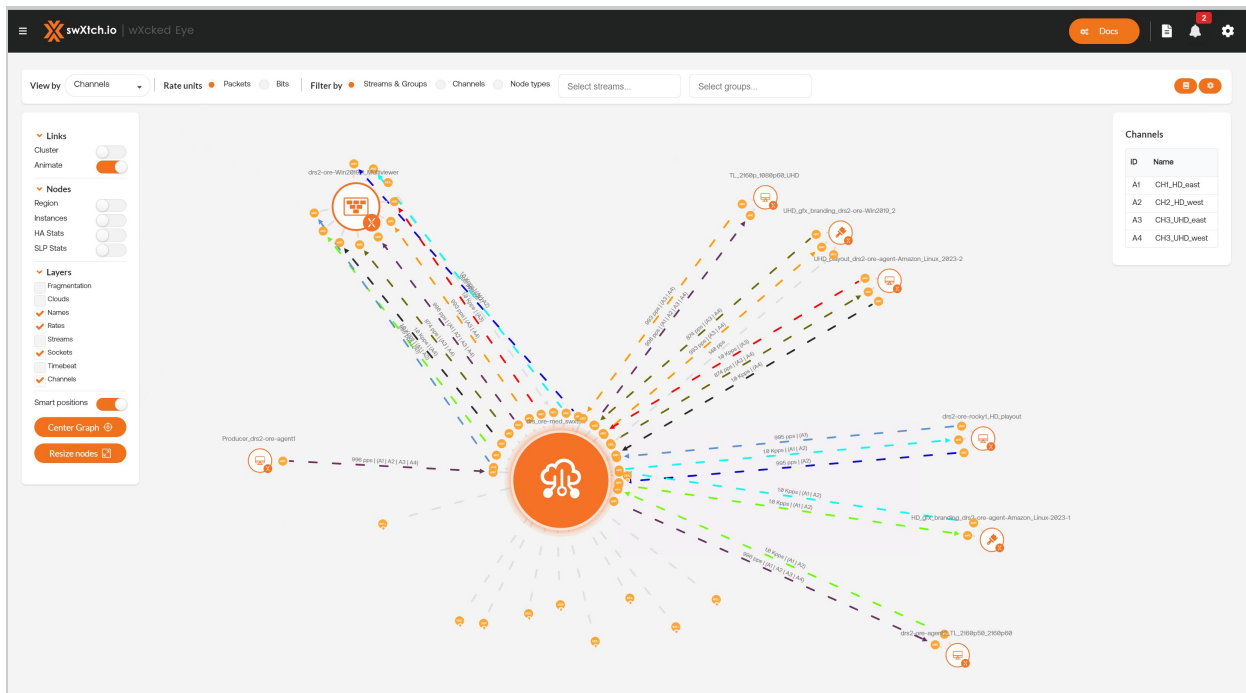
Filter By Streams and Groups

Filtering by Streams and Groups will allow users to specify streams or stream groups they would like to see highlighted in the Topology Graph.

In this example, the user has selected one stream, 224.2.2.3:8400, to be highlighted. Since we are using the HA view, the paths are marked **Blue**. The other stream, 224.2.2.4:8400, would be marked in **Red**. If a user had created a Streams Group in the Streams tab on the wXcked Eye Settings page, the option to select a group will appear in a drop down menu.



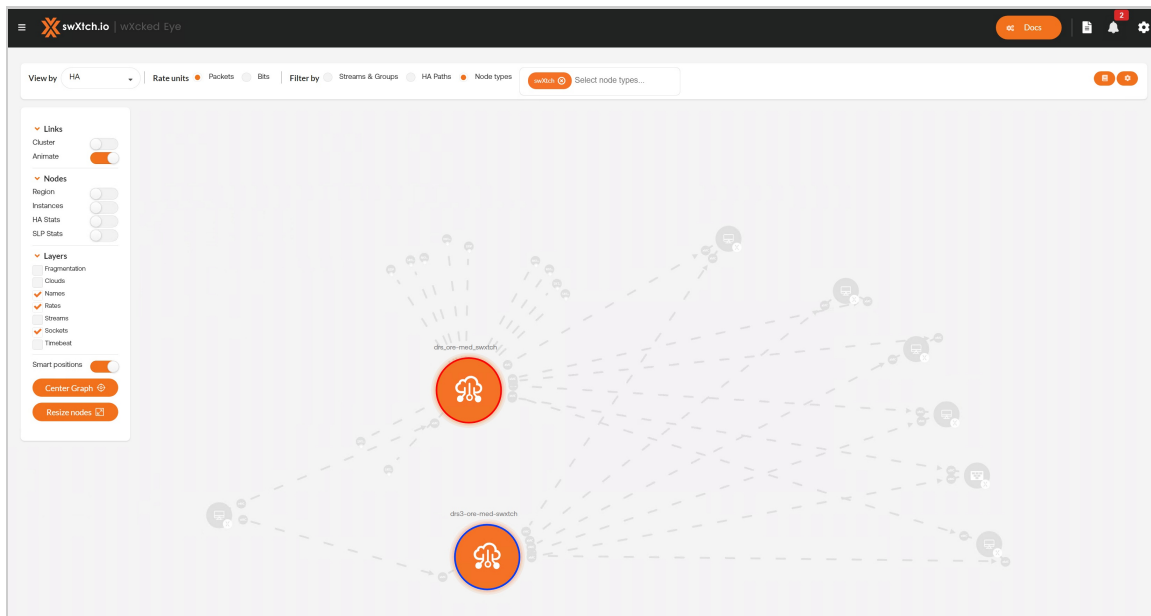
When the View is set to Channels, an additional Filter By Channel option will activate, allowing users to highlight streams related to a specific channel. In the example below, the user filtered on the channel "(A3) HD_east." All related streams are highlighted in the graph.



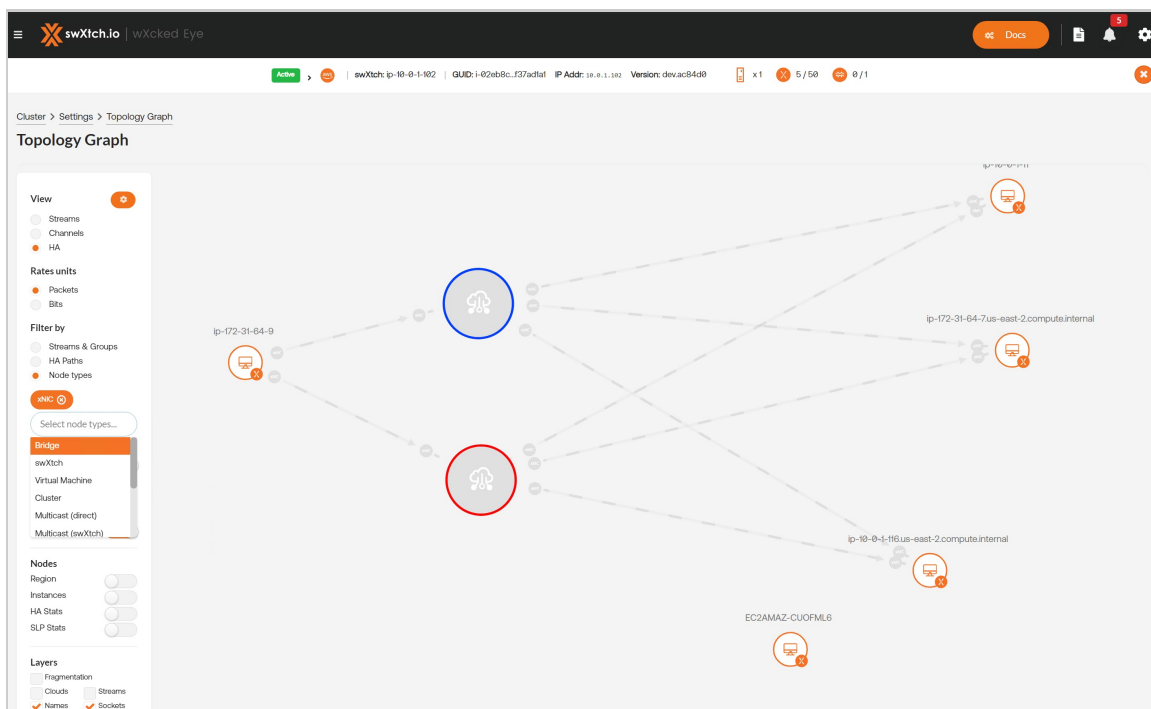
Filter by Node Types

With Node Types selected, users can highlight specific nodes in the graph based on their type: cloudSwitches, cloudSwitch Bridges, VMs with xNICs, and non-xNIC VMs.

In the example below, the user has chosen the cloudSwitch node to be highlighted in the Topology Graph. Since our main cloudSwitch is connected to another, there are two displayed in the Topology Graph.



In this next example, the user has all the xNIC VMs highlighted. User can distinguish between xNIC VMs and non-xNIC VMs by the xNIC symbol on the node.



Additional Options

The Topology Graph also provides users with some additional options to alter how the data is visualized.

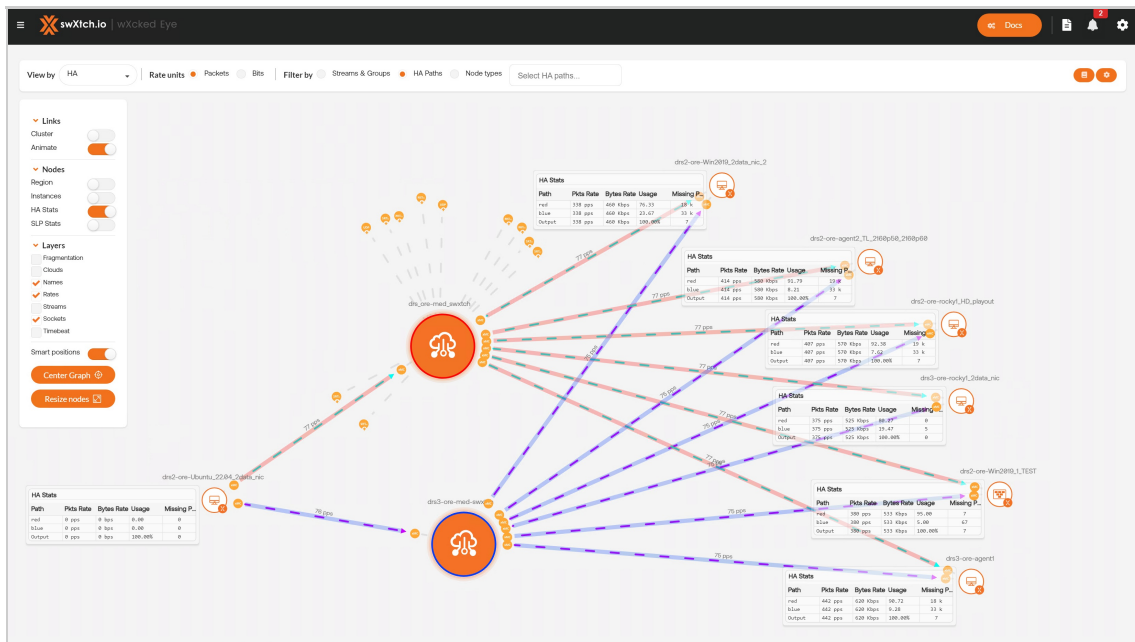
- **Smart positions:** Components in the Topology Graph will snap into a user-friendly position. **Note:** All positions are saved continually every 5 seconds, regardless of whether or not this feature is toggled on.
- **Cluster links:** When activated, links between nodes will cluster depending on the number set in the topology settings.
- **Animate links:** When activated, the links illustrating data flow will animate the direction.

Node Details

Under Node Details, users can toggle certain information about the nodes on/off. This includes:

- **Fragmentation:** This will display the amount of fragmented packets at the node.
- **Region:** This will display where the node is located.
- **Instances:** This will display information about a virtual machine's instance.
- **HA Stats:** When activated, information about data flow in an HA configuration at the node will be displayed.
- **SLP Stats:** When activated, information about lossless data flow at the node will be displayed.

In the example below, the user has **HA Stats toggled on**. At each node, information regarding HA is displayed.



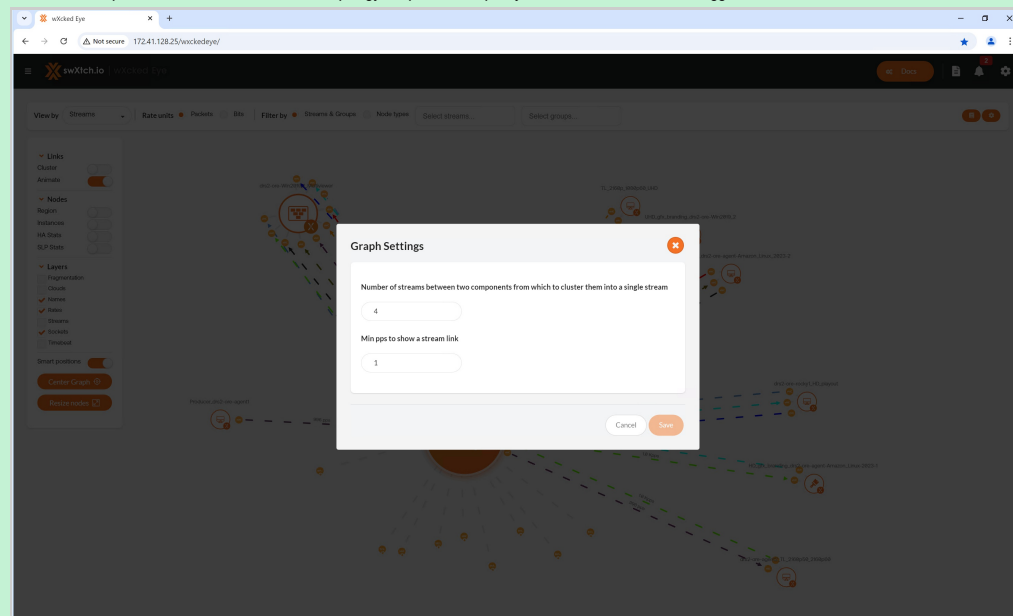
Layers

Layers in the network graph can be toggled on and off depending on the information a user wishes to display:

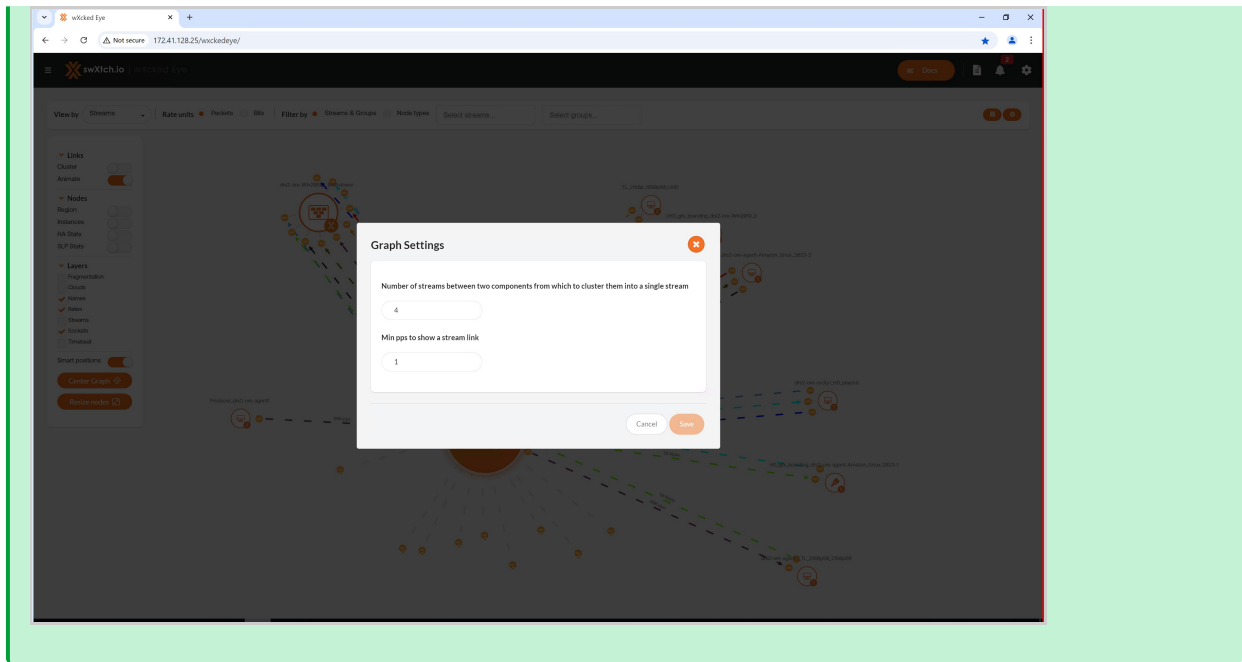
- Names
- Streams
- Rates
- Sockets
- Clouds
- Timebeat

Setting Up Cluster Links

Cluster links groups together streams of a similar direction into a single link. Users can set a specific number to trigger this clustering feature. To do this, select the Gear symbol next to the View in the filter panel on the left-hand side of the Topology Graph. There, specify the number of links that will trigger a cluster.



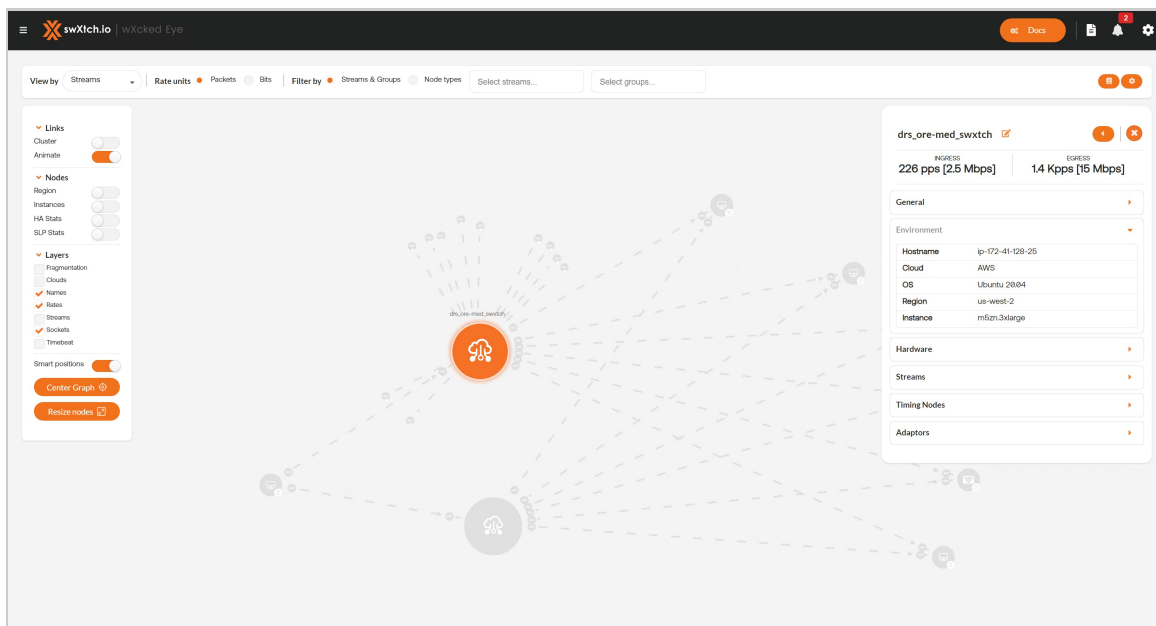
In the example below, the user set the number of streams to 7, causing the number of links to drastically decrease between each component.



Component Information Sidebar

When selected, each component in the Topology Graph will open an information sidebar. This will display information regarding the component, organized into eight categories: Statistics, General, Environment, Hardware, Streams, Adaptors, High Availability, and Timebeat.

At the top of the component sidebar, users will see a high level view of Total Rx and Total Tx statistics from the perspective of the selected component. In the example below, the user has the cloudSwtch in the center selected and can see the data being received and transmitted.



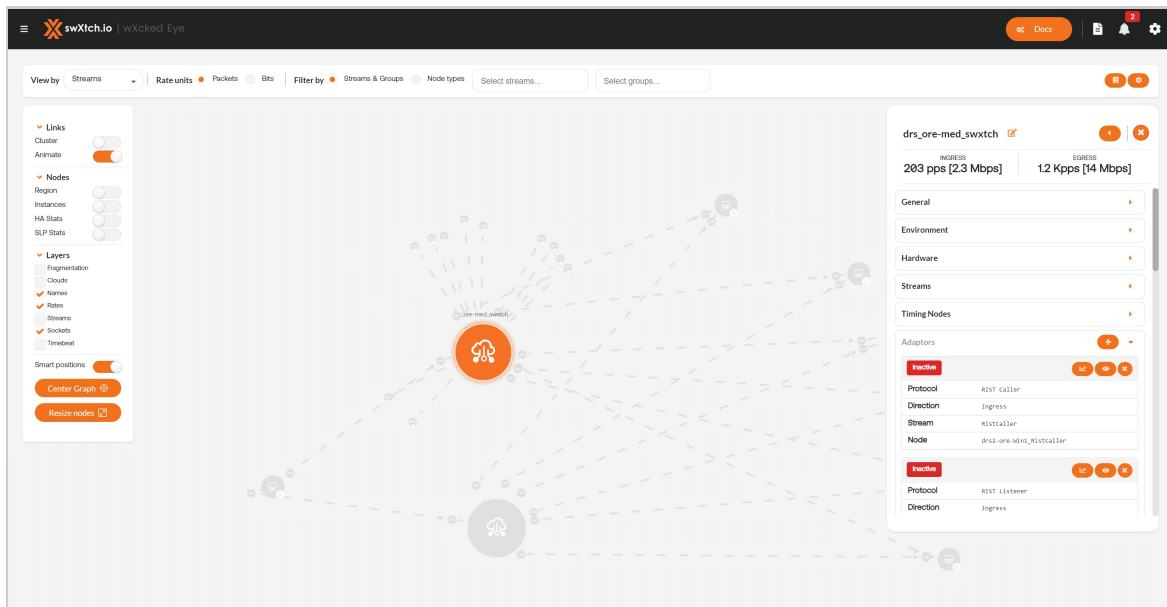
The **General** section details the type of component (cloudSwtch, xNIC, cloudSwtch Bridge, etc.), the data eth, data port and the version. The **Environment** provides information regarding the cloud provider, including the host name, cloud, OS, Region, and Instance type. The **Hardware** section goes deeper into the information regarding the virtual machine, specifically the NICs.

The **Streams** section provide users with a detailed list of all the streams going through the component and the direction of data flow.

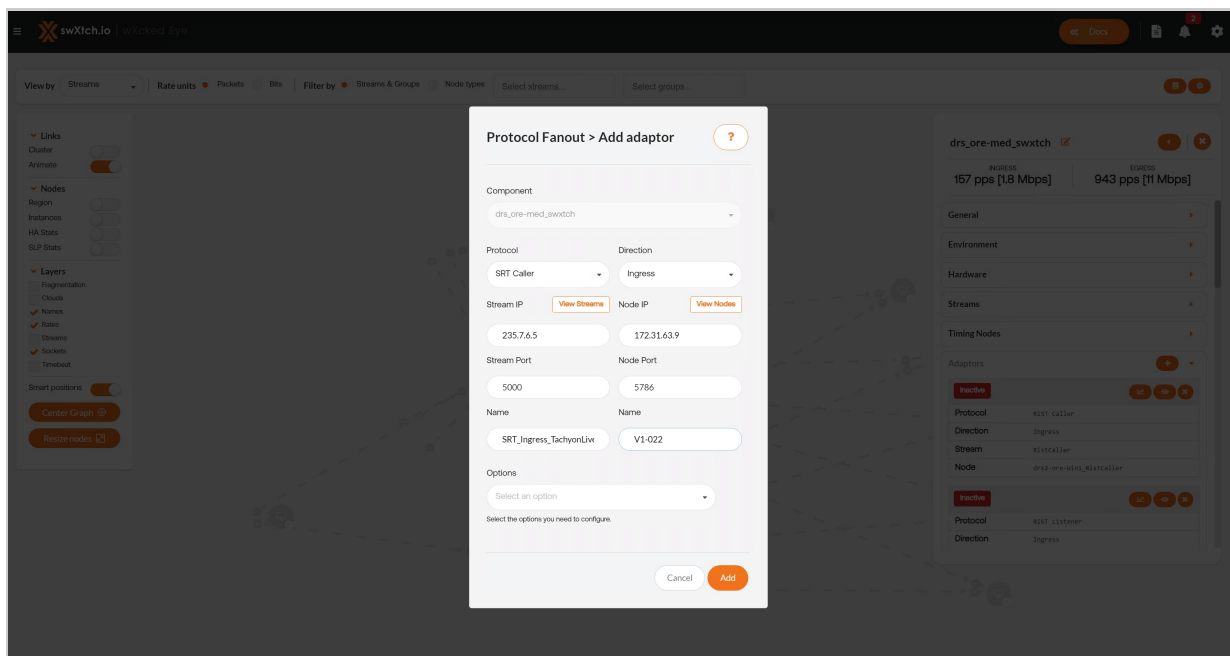
The **Timing Nodes** section will populate with information if any are configured in the cloudSwtch network for Precision Time Protocol.

Adding Adaptors

Under **Adaptors**, users can view adaptor details, add, edit, remove, and link to adaptor stats. Adding an adaptor will open a window directly in the Topology Graph.

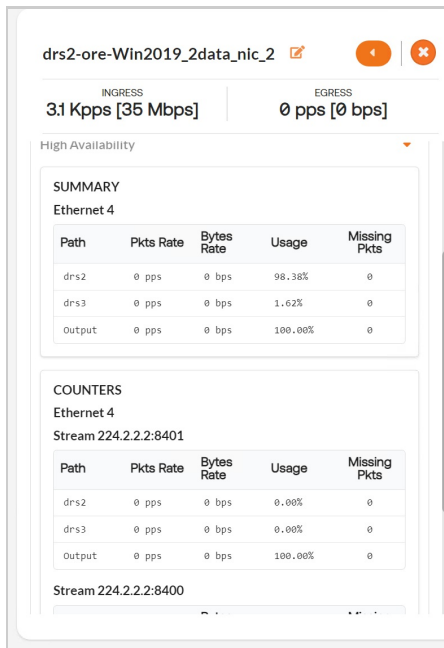


The pop-up window is similar to what can be found under the [Protocol Fanout tab on the Settings page](#). Users can specify the protocol, direction, and stream information necessary to add an adaptor.



Reviewing High Availability Stats at xNIC Node

When selecting a node configured for **high availability**, a user can see information regarding data flow between the two paths. This is organized into two subsections: Summary and Counters. Summary takes a holistic look at the two paths while Counters displays information at the stream-level.



Adding Cloud To Ground (C2G) Subscriptions for cloudSwXtch Bridge

When selecting a cloudSwXtch bridge component in the Topology Graph, an additional Subscriptions (C2G) section will be listed. Here, users can add cloud to ground subscriptions directly in the UI and apply them to the bridge configuration JSON.

To do this:

1. Click the cloudSwXtch Bridge component in the Topology Graph to open the Component Information panel.
2. Go to the Subscriptions section and click the "+" button to open a new window.

swXtch.io | wXcked Eye

View by: Streams | Rate units: Packets | Filter by: Streams & Groups

bridge-3

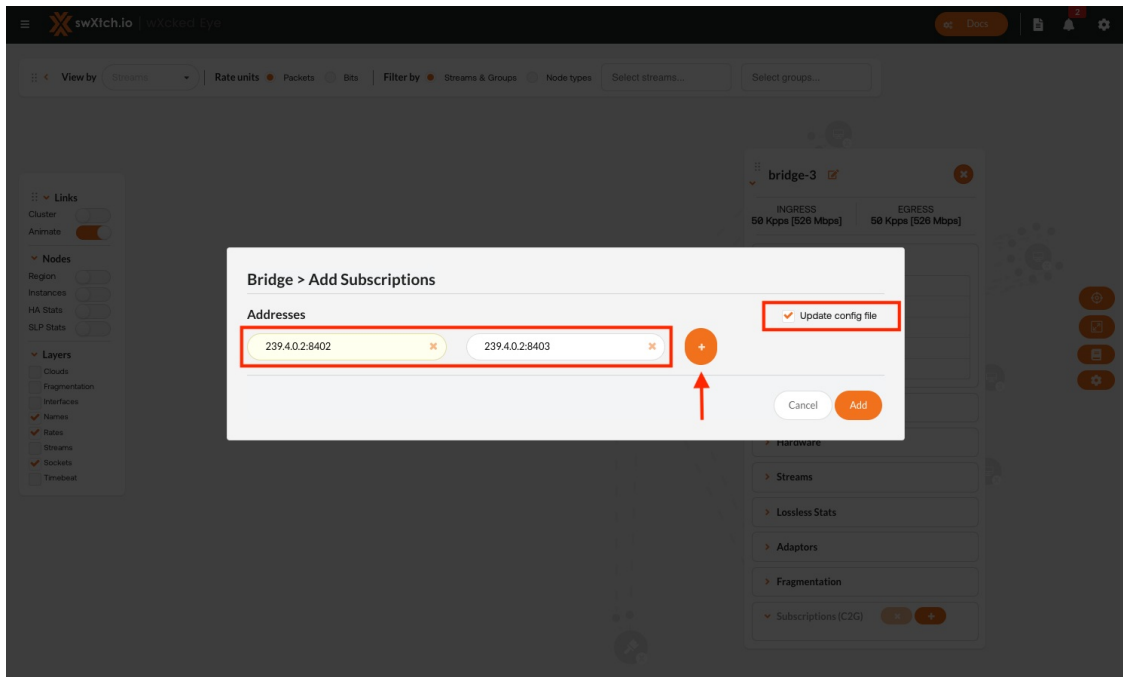
INGRESS: 50 Kpps [524 Mbps] | EGRESS: 50 Kpps [524 Mbps]

General

Class	Bridge
Type	Unknown
Data Eth	data
Data Port	9093
Version	demo

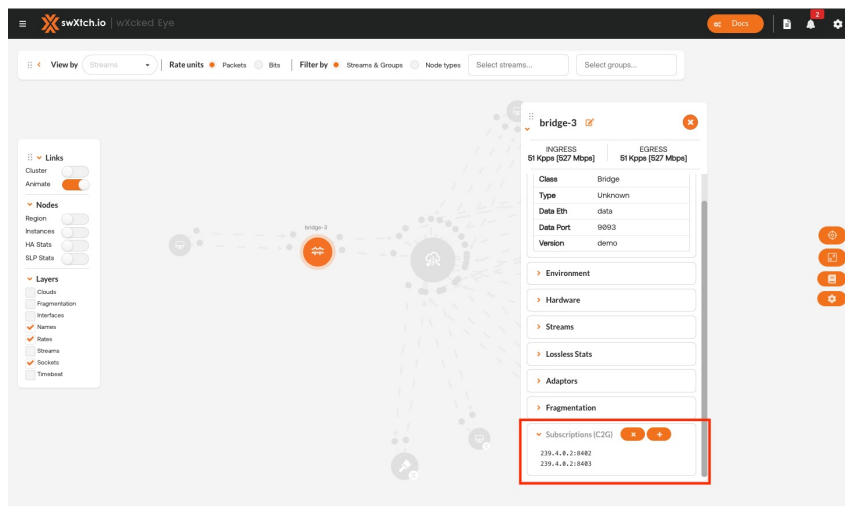
Subscriptions (C2G) +

3. Add the IP addresses of the C2G subscriptions by clicking the "+" button.
 - a. Optionally, you can select Update config file to update the bridge config JSON file with the new C2G Subscriptions.



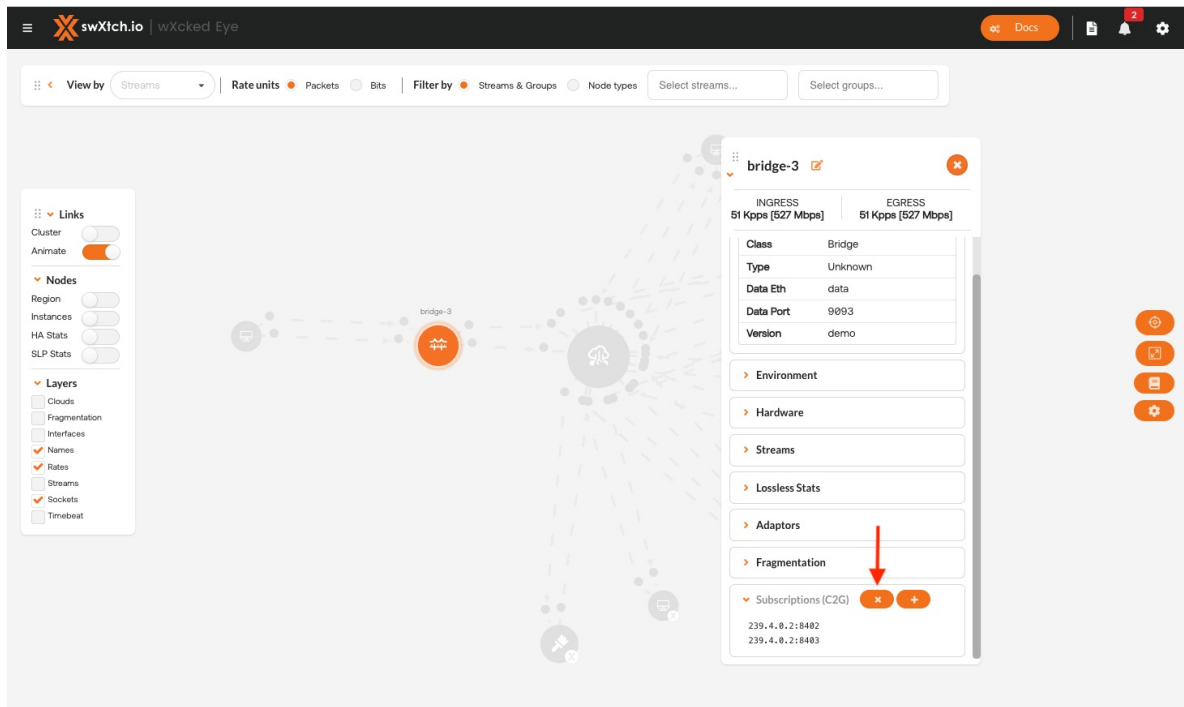
4. Click Add to confirm.

The new subscriptions should now list in the Component Information panel.

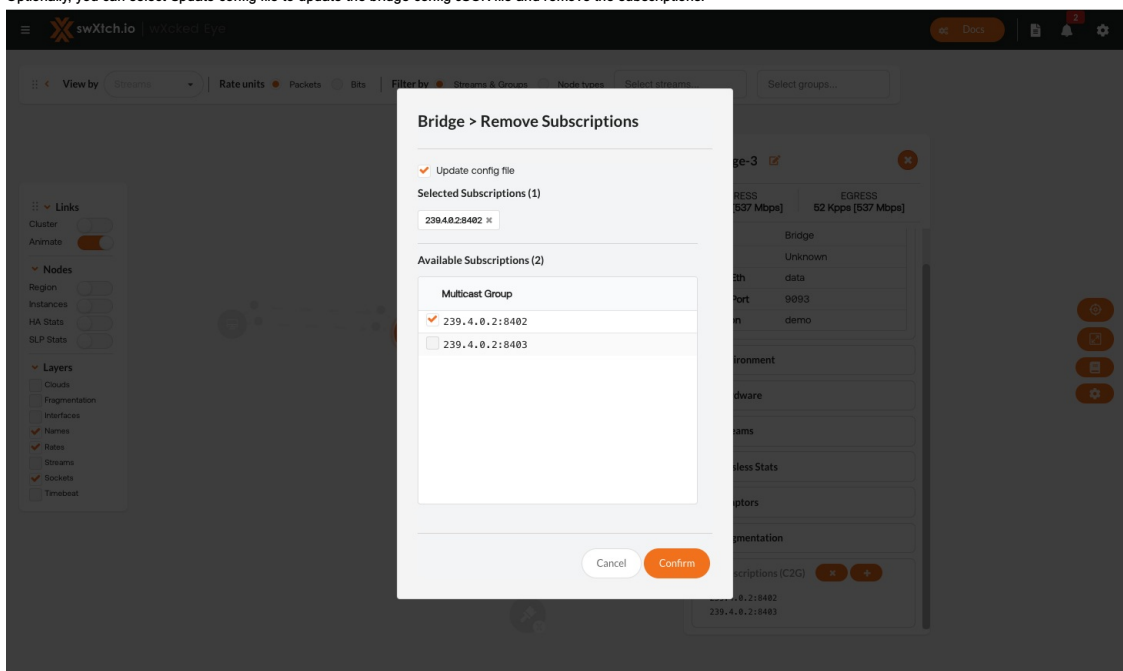


To remove:

1. Click the "X" button next to the Subscriptions (C2G). A Remove Subscriptions window will open.

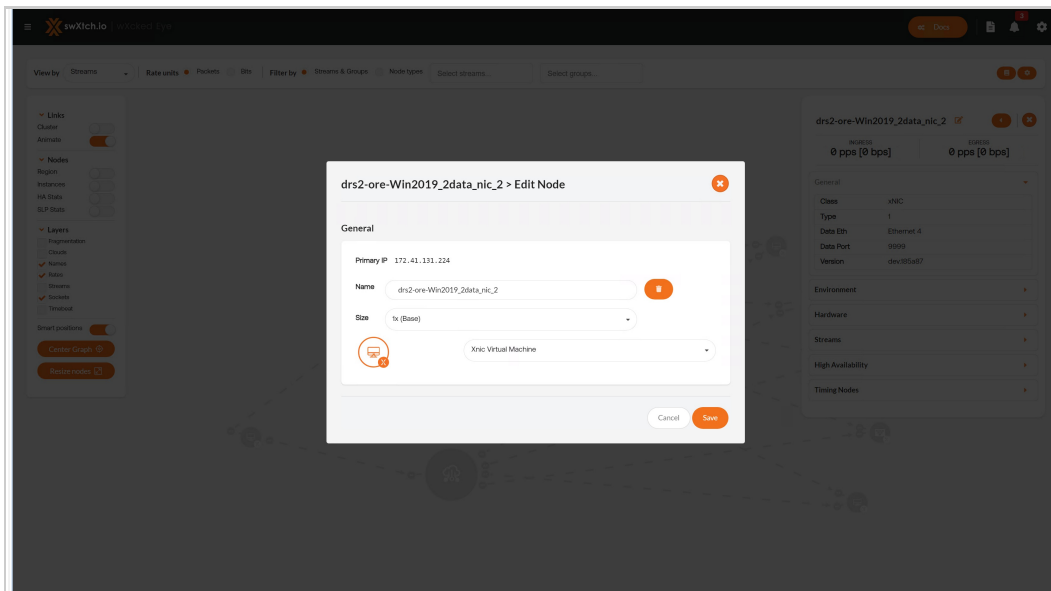


2. Select the multicast groups you will like to remove.
 - a. Optionally, you can select Update config file to update the bridge config JSON file and remove the subscriptions.



3. Click Confirm to remove it.

Editing a Node's Appearance



Users can change the appearance of each node or component in the Topology Graph. By selecting the **Edit Pad** next to the component's name in the Information Sidebar, a user can edit the size, the name and the icon for each node.

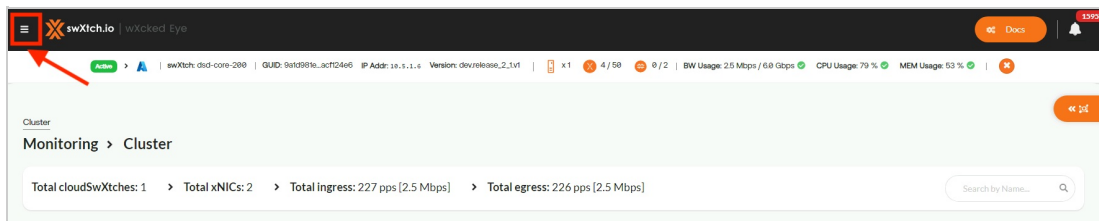
wXcked Eye Topology Table

WHAT TO EXPECT

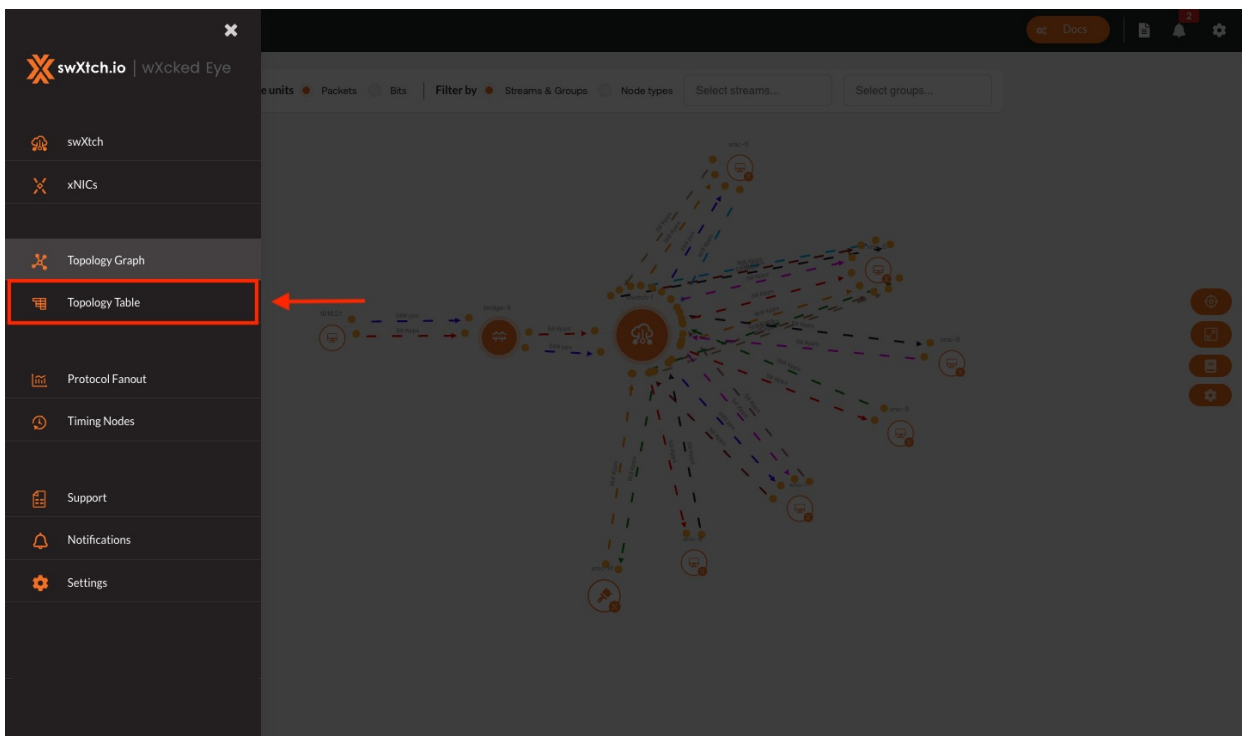
In this section, users will learn how to navigate the wXcked Eye Topology Table. Similar to the Topology Graph, the table organizes statistics by either Streams or Channels in a tree view.

Locating the wXcked Eye Topology Table

To navigate to the wXcked Eye Topology Table page, users will need to click on the menu (≡) option at the top right hand corner by the swXtch.io logo.



For there, select **Topology Table**.

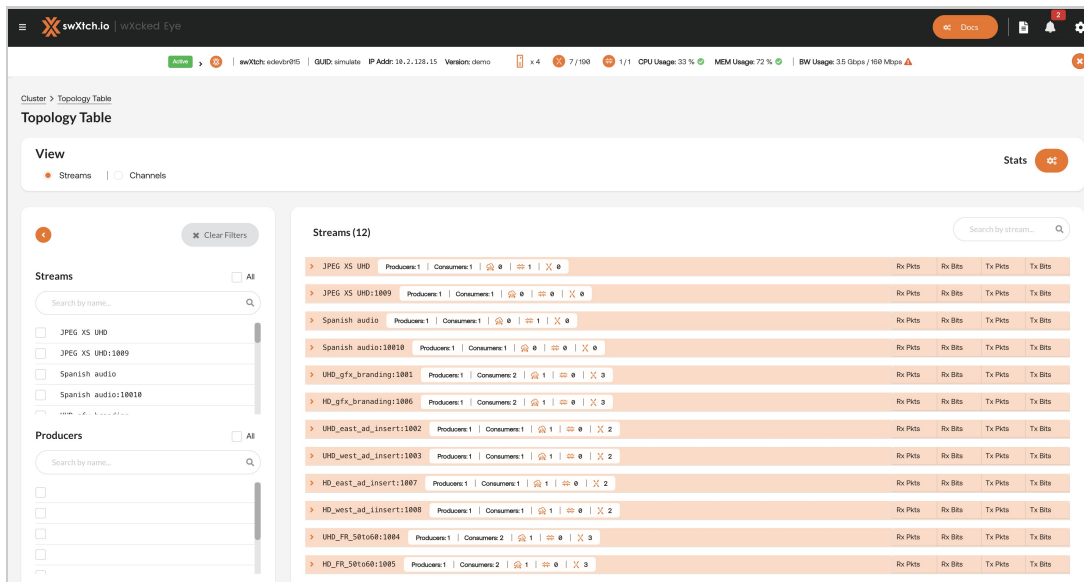


Using the wXcked Eye Topology Table

The wXcked Eye Topology Table provides users with an alternative way of viewing how data flows in and out of their cloudSwXtch. Instead of a Topology view with nodes and steam links, the table organizes streams and channels in tree view where they can easily see the origin of the data flow and how it moves through different components. This gives users a more focused look into a single stream or channel.

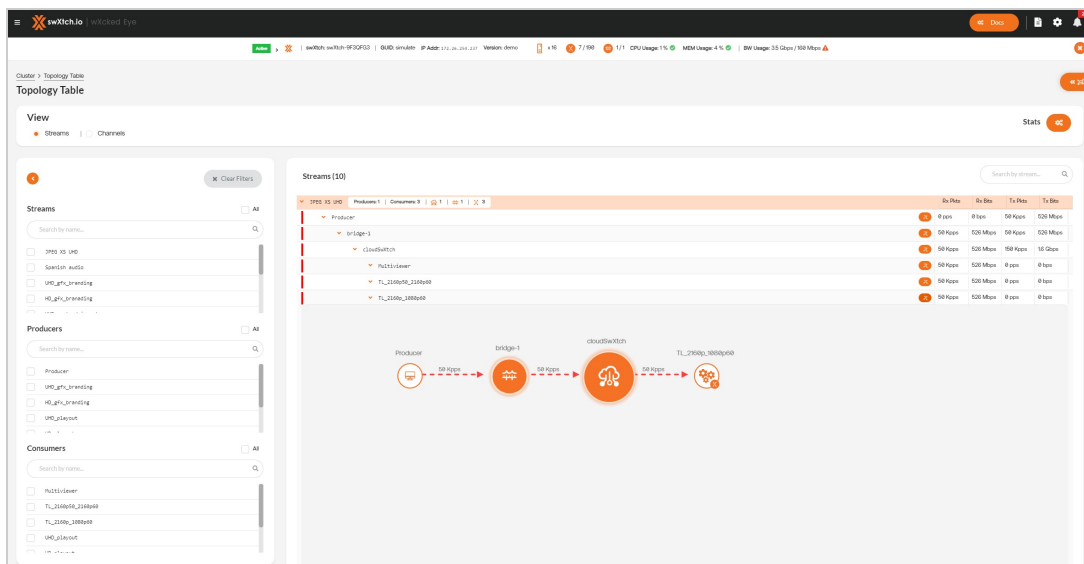
Streams

In the example below, the user is viewing a list of streams. Similar to the Topology Graph, a user can filter specific streams, consumers, and producers. The list will adapt to those selections. This can be helpful when dealing with a large amount of line items.



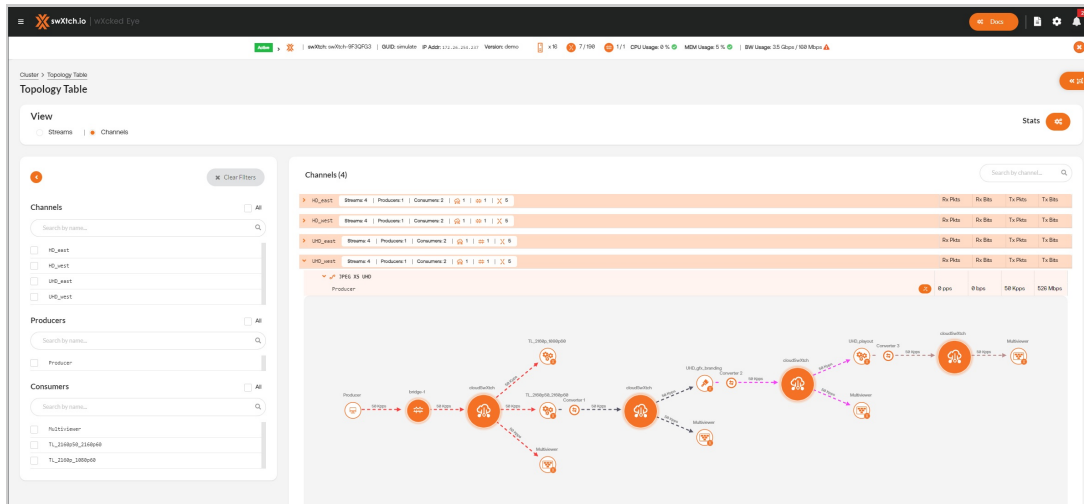
Each item in the stream list can be expanded to reveal a tree level view of the data flowing through the various components. The JPEG XS UHD stream is expanded to show how data flows through the various cloudSwTch components. Starting from the producer, the stream moves through the cloudSwTch Bridge up into the cloudSwTch, where it feeds the stream into a multiviewer and Tachyon Live endpoints.

Selecting the Topology button next to the component name will reveal a visualization similar to the Topology graph up until that point.



Channels

The Channels view organizes channels into a similar list for the Topology table. However, one big difference is that the number of streams associated with a single channel is also displayed. The table will list those streams in a tree view, displaying how they progress through the various cloudSwTch components. In the example below, the UHD_west channel has been expanded, revealing the JPEG XS UHD stream with a topology graph view of data flow.



Multiple cloudSwXtch Nodes

In the Channels view example above, the cloudSwXtch node repeats. This does not mean that multiple cloudSwXtches are being used but rather the stream is flowing back and forth to the same cloudSwXtch.

wXcked Eye Protocol Fanout Stats

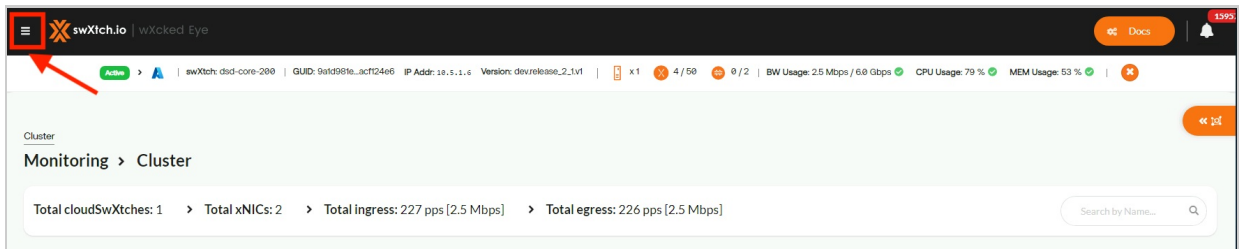
WHAT TO EXPECT

The **Protocol Fanout Stats** page allows users to see metrics for non-Multicast and non-Broadcast data flows. This includes protocols, such as **SRT Caller**, **SRT Listener**, **RIST Caller** and **RIST Listener**.

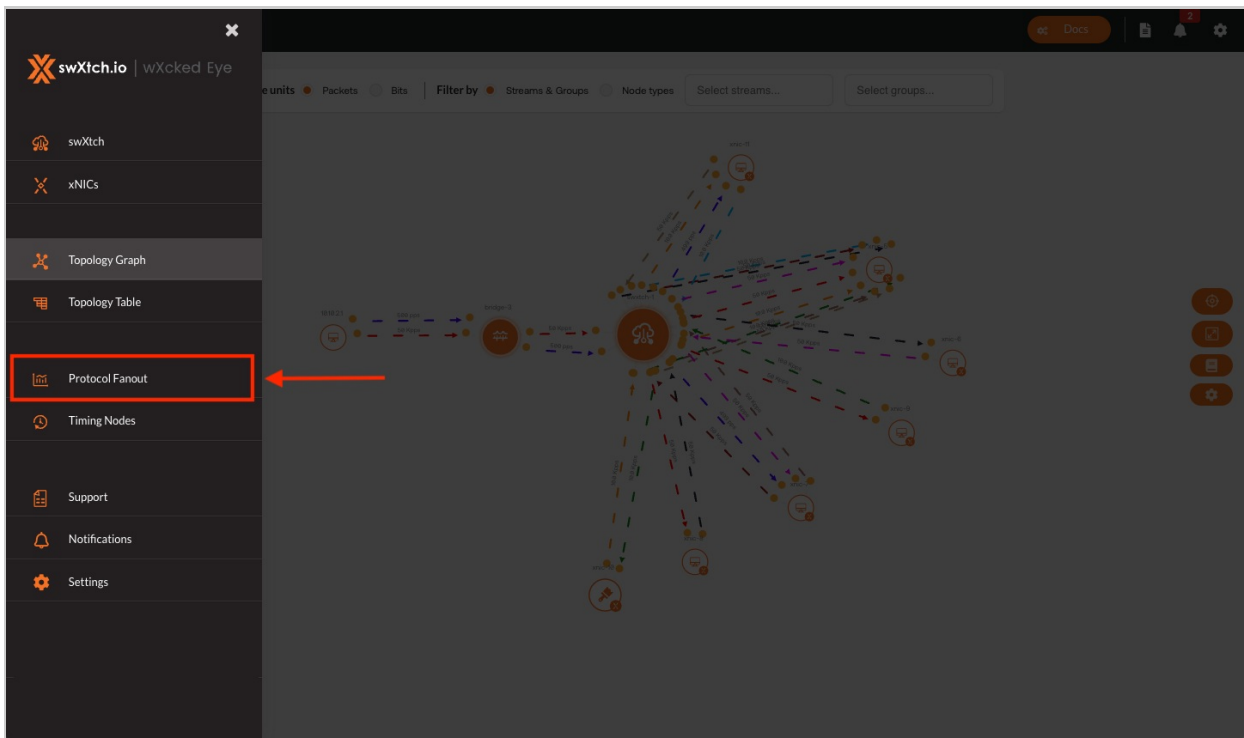
In this section, users will learn how to navigate between different protocols and their adaptors in order to better visualize their packets' movements. **Please note:** This page only shows the stats for protocol fanout configurations. To learn how to configure your cloudSwXtch for Protocol Fanout and Conversion in the wXcked Eye UI, please read [this article](#).

Locating the Protocol Fanout Stats Page

To navigate to the **Protocol Fanout Stats** page, users will need to click on the menu option at the top right corner by the swXtch.io logo.



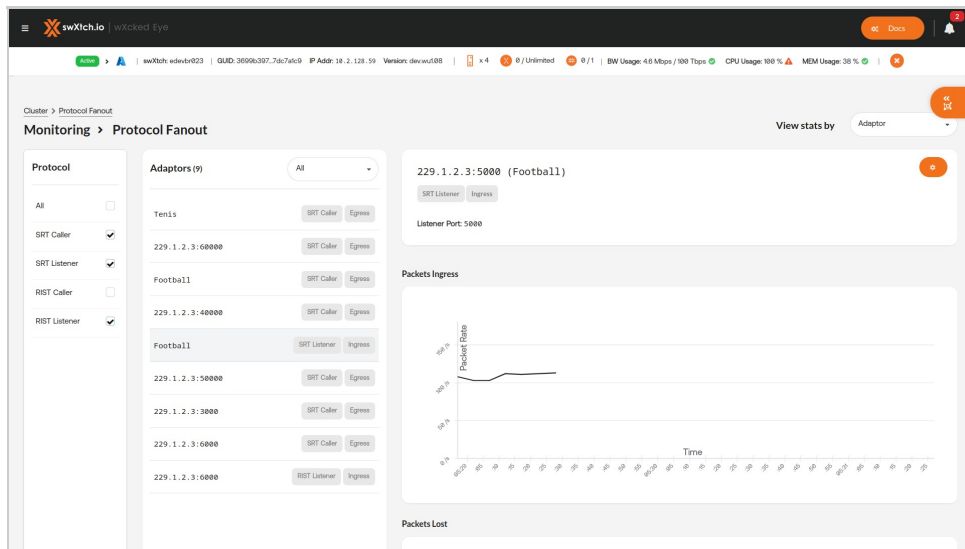
From there, select **Protocol Fanout**.



Navigating the Protocol Fanout Stats page

Statistics displayed through the wXcked Eye UI focus primarily on multicast and broadcast data flow. The **Protocol Fanout Stats** page provides users with a dedicated area to see data flow for alternative protocols like SRT and RIST.

Adaptor View



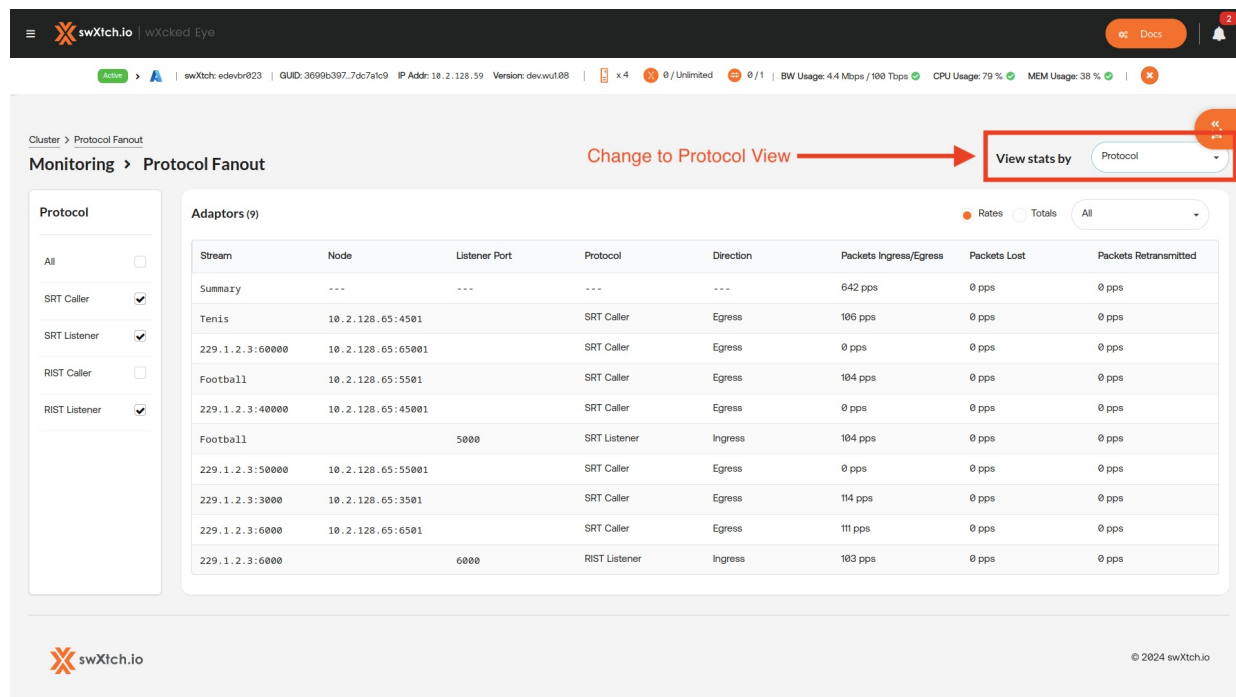
Adaptor View

At start-up, the page will be in **Adaptors** view and display 3 graphs:

1. Packets Egress
2. Packets Lost
3. Packets Retransmitted

A user can select the **Protocols** they want listed in the left hand side to filter Adaptors or return to "All" to see them all listed. At the top of the Adaptors panel, users can filter the list further by direction -- either Ingress or Egress.

Protocol View



A user can select to **View Stats by Protocol** by clicking the dropdown menu in the upper right hand corner. This will provide users with a table view, listing adaptors by protocol. This allows for closer comparison between adaptors. The table can be configured to display in both **Rates (pps)** or **Total Packets**.

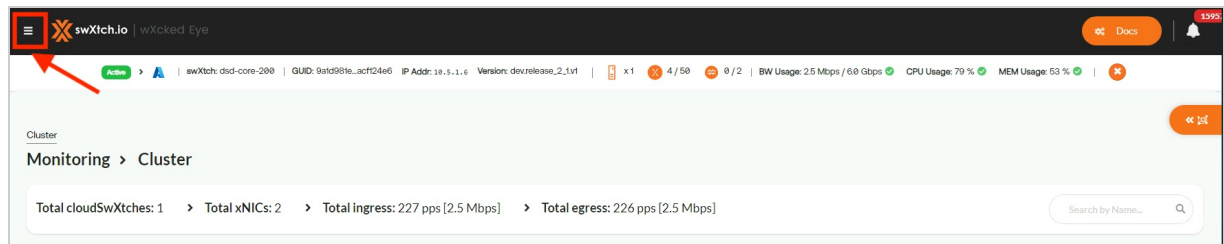
Similar to the Adaptors view, users can group adaptors by protocol. The Protocol panel on the left hand side enables users to filter our certain protocols so that they list in the center table. In addition, users can also group protocols by direction -- either Ingress or Egress.

wXcked Eye Timing Nodes

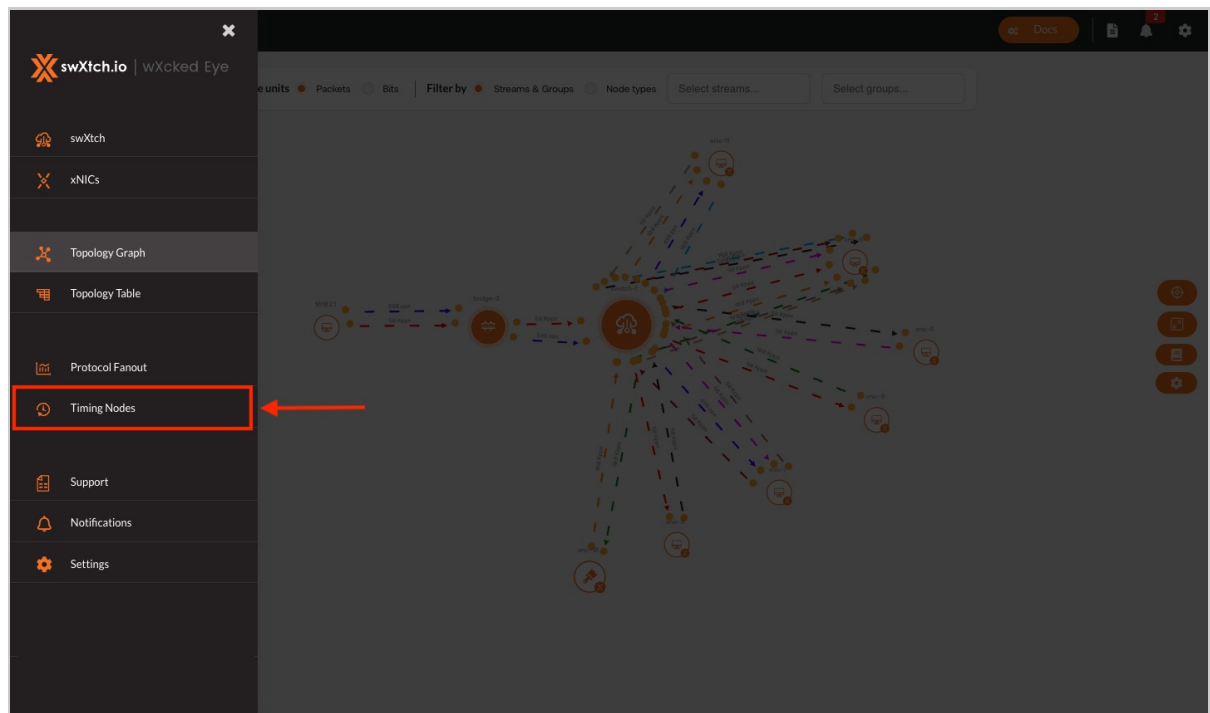
Finding the Timing Nodes page

To find the **Timing Nodes** page in the wXcked Eye UI:

- 1. Click the menu icon (≡) next to the swXtch.io logo.



- 2. Select **Timing Nodes**.



Understanding the Timing Nodes page

Cluster > Settings > Timing Nodes

Monitoring > Timing Nodes

Update credentials

Master Node

Name: dsd-core-100 Time Sync Service: phc/dev/ptp_hyperv

Follower Nodes (8)

CSV

Name	Status	Local Offset	Root Offset
DSd-agent-104	Present	5.62 μ s	16.39 μ s
DSd-agent-105	Present	3.54 μ s	26.03 μ s
DSd-agent-106	Present	9.03 μ s	19.29 μ s
DSd-agent-204	Not present	--	--
DSd-agent-205	Not present	--	--

The Timing Nodes page displays information regarding clock sync configuration for the cloudSwXtch. This page in wXcked Eye will only populate with information if the user has the PTP feature enabled.

In the example above, the cloudSwXtch (DSd-core-100) is acting as the Master Node.

- **Master Node**- The Master Node is what the PTP configuration sets as the most reliable time source. This will send the true time it receives from the source clock to the Follower Nodes.

- **Name** - The name of the cloudSwXtch
- **Time Sync Service** - The source clock
- **Follower Nodes** - The Follower Nodes lists the agents/VMs that subscribe to the Master Node for accurate timing.
 - **Name** - The name of the endpoints
 - **Status** - The status of the endpoints, noting if the node is active in the PTP configuration
 - **Local Offset** - The local offset denotes the offset in time from the cloudSwXtch to the xNIC.
 - **Root Offset** - The root offset denotes the offset in time from the GrandMaster clock to the cloudSwXtch and its follower nodes (xNIC). Note how the root offset is larger than the local offset. This is normal behavior since the distance between the follower node and the Grandmaster clock is greater than the offset between a cloudSwXtch and xNIC.

Timing Nodes Stabilization

After upgrading or rebooting your cloudSwXtch system, you may notice that the local and root offset values are much larger than they actually are. It can take up to 30 minutes for the values to stabilize and return back to normal levels.

Exporting your Timing Nodes

You can export your timing nodes by hitting the **CSV button** next to **Follower Nodes**.

Formatting CSV Timing Nodes file in Excel

To prevent incorrect formatting in your CSV Timing Nodes file in Excel, complete the following steps:

1. **Make sure** your Timing Nodes CSV file is already downloaded from wXcked Eye.
2. **Select** "Data" from the top ribbon of a new Excel spreadsheet.
3. **Click** "Get Data (Power Query)."
4. **Select** "Text/CSV" from the "Choose data source" options.
5. **Browse** for your file and click "Get Data."
6. **Click** "Next."
7. **Select** "Unicode (UTF-8)" from the File Origin dropdown menu. This ensure your data displays as it was intended.
8. **Click** "Load."

wXcked Eye Support Page

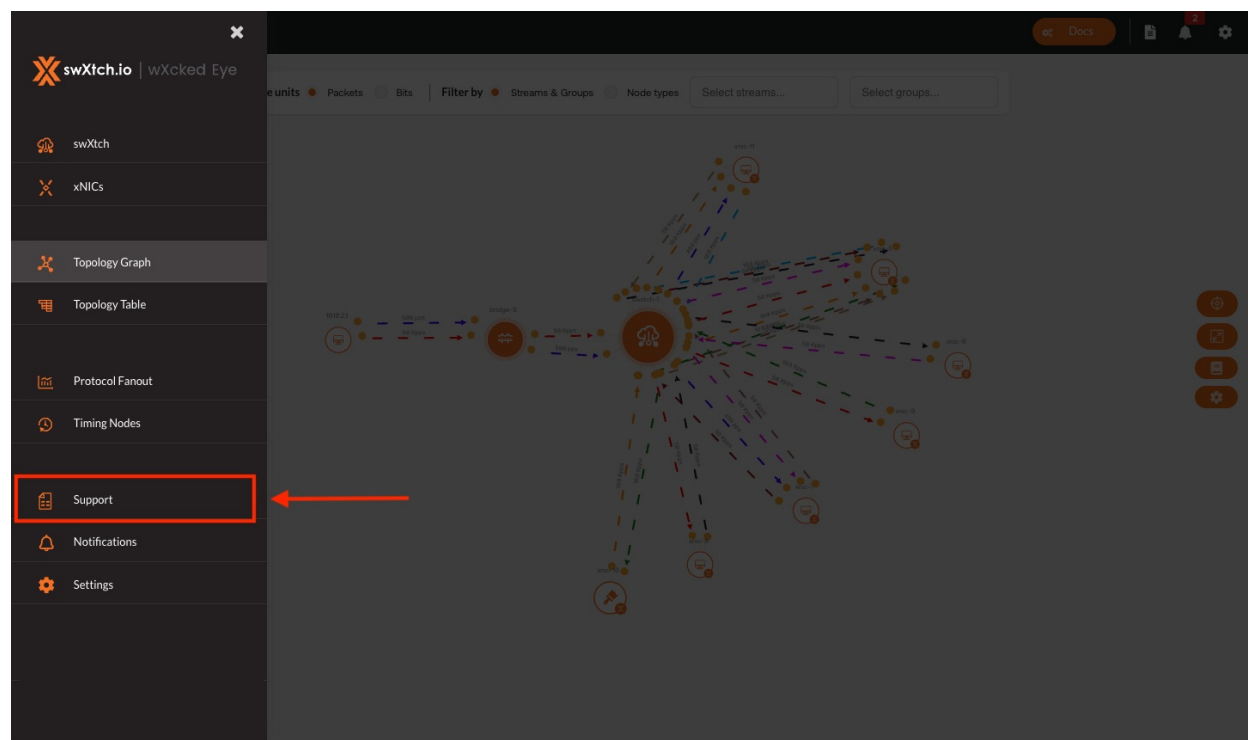
WHAT TO EXPECT

The wXcked Eye Support Page allows users to export a report detailing the statistical data stored within the cloudSwXtch over a set period of time. This report includes JSON files containing cloudSwXtch information, Max Highmarks, List Highmarks, Repl and Control logs, and xNIC Logs -- all in a compressed file. This report should be provided to swXtch.io Support when troubleshooting an issue.

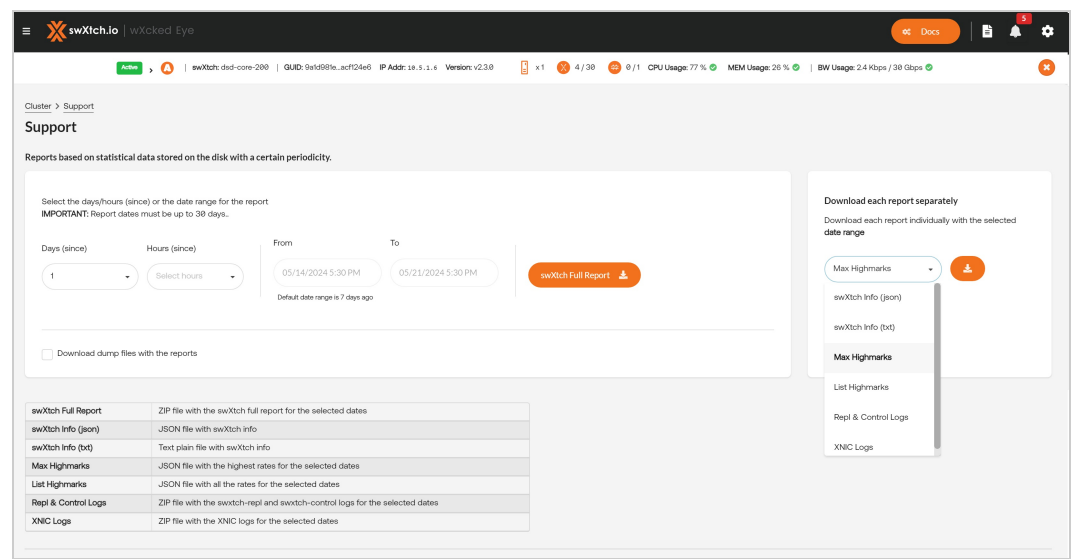
In this section, we will walkthrough exporting both a full report and also individual files. Alternatively, you can use the swx support command in the cloudSwXtch VM to export a report. For more information, see [How to View cloudSwXtch Logs for Troubleshooting](#).

Navigating the wXcked Eye Support Page

The wXcked Eye **Support** page can be located in the wXcked Eye navigation menu.



The page has two functionalities: **exporting a full report** or **selecting individual JSON files to download separately**. When troubleshooting, it is **recommended** for users to send the complete report so that swXtch.io Support can fully understand the situation.



To do this, simply set a **start** and **end time** for the report and select the download button, **SwXtch full report**. User should set the duration to **at least 24 hours of time**, spanning from a little before the issue began to up until now. The output file will be named **swxtch-report-date_from.tar.gz**.

In the event that a user only wants a specific section of the report, they can use the dropdown menu after **Download each report separately** and download their desired JSON file. The wXcked Eye UI will then export using the time period set.

Contacting swXtch.io Support

For all troubleshooting requests, email the compressed file to support@swxtch.io for further instructions.

Configure cloudSwXtch with wXcked Eye

WHAT TO EXPECT

wXcked Eye allows users to configure their cloudSwXtch directly from a graphical user interface (GUI).

To learn how to use wXcked Eye to monitor your cloudSwXtch, please see the ["Monitor cloudSwXtch with wXcked Eye"](#) article.

In this article, users will learn how to navigate the "Settings" option in the wXcked Eye UI and to configure their cloudSwXtch for mesh, high availability and protocol fanout. To learn how to access the wXcked Eye UI, please review the following article.

Accessing wXcked Eye

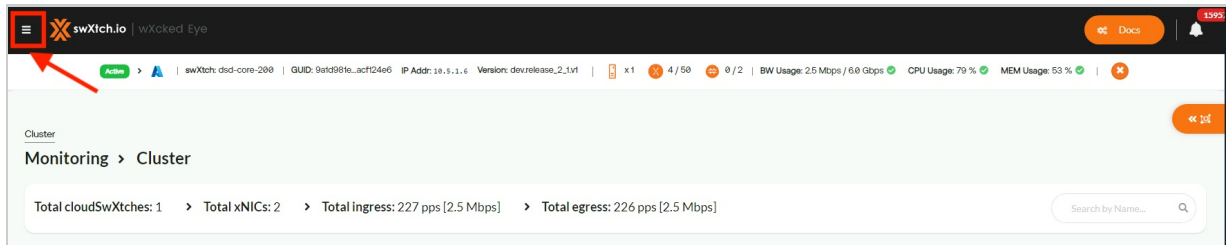
To access the wXcked Eye UI, users will need to enter the following URL into a web browser of a VM in their cloudswXtch environment. They should use the IP address of their cloudSwXtch to prefix the URL.

Custom	Copy
<swxtch-ip-address>/wxckedeye/	

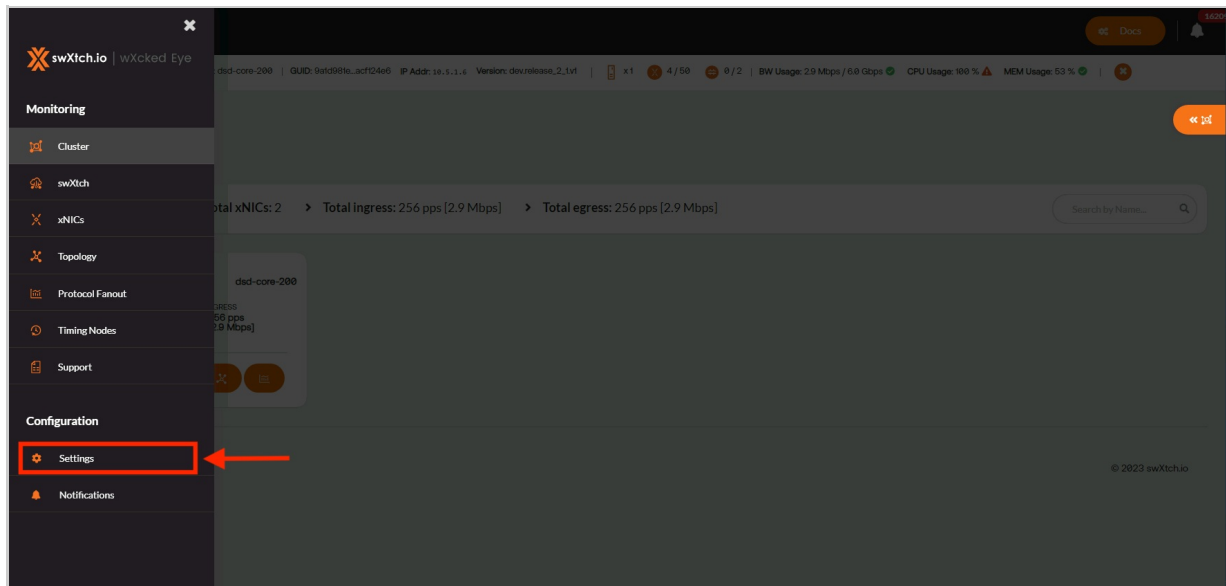
Finding the Settings page

To find the Settings page in the wXcked Eye UI:

1. Click the menu icon (≡) next to the swXtch.io logo.



2. Select **Settings** under **Configuration**.



3. You should now be on the **Settings** page.

Navigating Settings

The Settings page is organized into five tabs with varying functionalities:

- [General](#)
- [Mesh](#)
- [High Availability](#)
- [Protocol Conversion and Fanout](#)
- [Aliases](#)

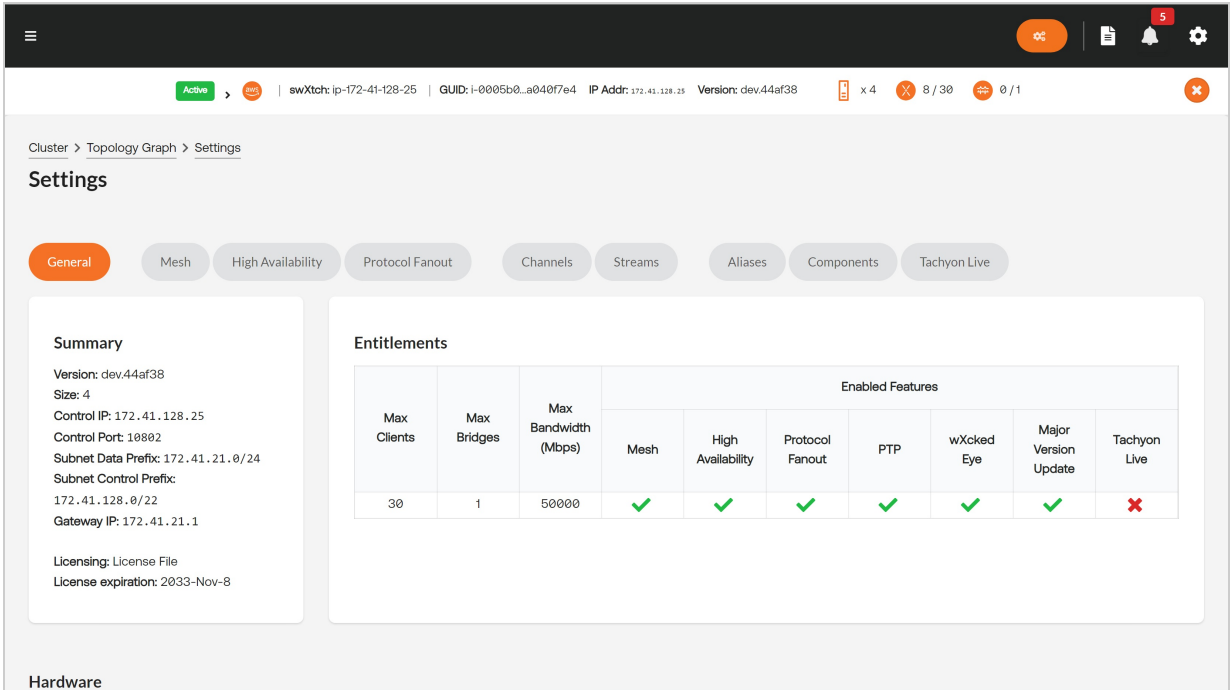
In this section, we will discuss each tab and how it offers the user additional control over their cloudSwXtch network.

General

How to Navigate to the General tab

To learn how to navigate to Settings from the wXcked Eye main page, please review the [Configure cloudSwXtch with wXcked Eye](#) article.

Navigating the General Tab



The General tab is organized into four sections:

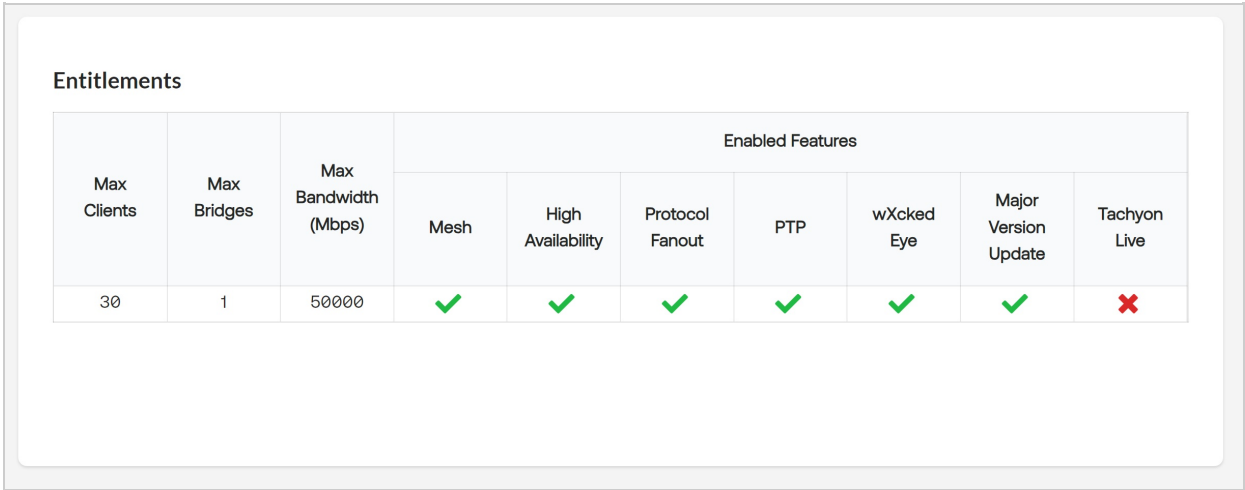
- [cloudSwXtch Summary](#)
- [Entitlements](#)
- [Hardware](#)
- [Actions](#)

cloudSwXtch Summary

The Summary panel details basic information regarding the cloudSwXtch, specifically on the data and control subnets configured during installation. The panel also tells the user the Licensing type and the expiration date.

Entitlements

The **General** tab is designed to give users a detailed look into the entitlements associated with their network. In the example, the user has a license that enables mesh, high availability, protocol fanout, PTP, and wXcked Eye with a max of 30 clients, 1 cloudSwXtch bridges, and 50000 Gbps max bandwidth. For Tachyon Live, the user does not have the entitlement unlocked.



Hardware

The Hardware section of the General tab gives an extensive look at the Data and Control planes with each split into Meta and Operating System (OS) data.

Hardware

Control

Meta

IP Address

10.2.128.10

IP Broadcast

10.2.131.255

IP Subnet

10.2.128.0

MAC Address

60:45:bd:a8:a6:ab

Subnet Mask

60:45:bd:a8:a6:ab

Os

Driver

hv_netvsc

Master of

MTU

1500

Name

eth0

PCI Address

Data

Meta

IP Address

10.2.192.23

IP Broadcast

10.2.195.255

IP Subnet

10.2.192.0

MAC Address

60:45:bd:a8:af:c5

Subnet Mask

60:45:bd:a8:af:c5

Os

Driver

hv_netvsc

Master of

MTU

1500

Name

eth1

PCI Address

Actions

The General tab also allows users to adjust the Data Refresh period for all Monitoring pages in wXcked Eye. This gives users control on how often the data is updating with the default value set to the minimum of 5 seconds.

Actions

Update swXtch

Data refresh period

This slider allows you to change how often the data is updated across all monitoring pages.

5 secs

High Availability with wXcked Eye

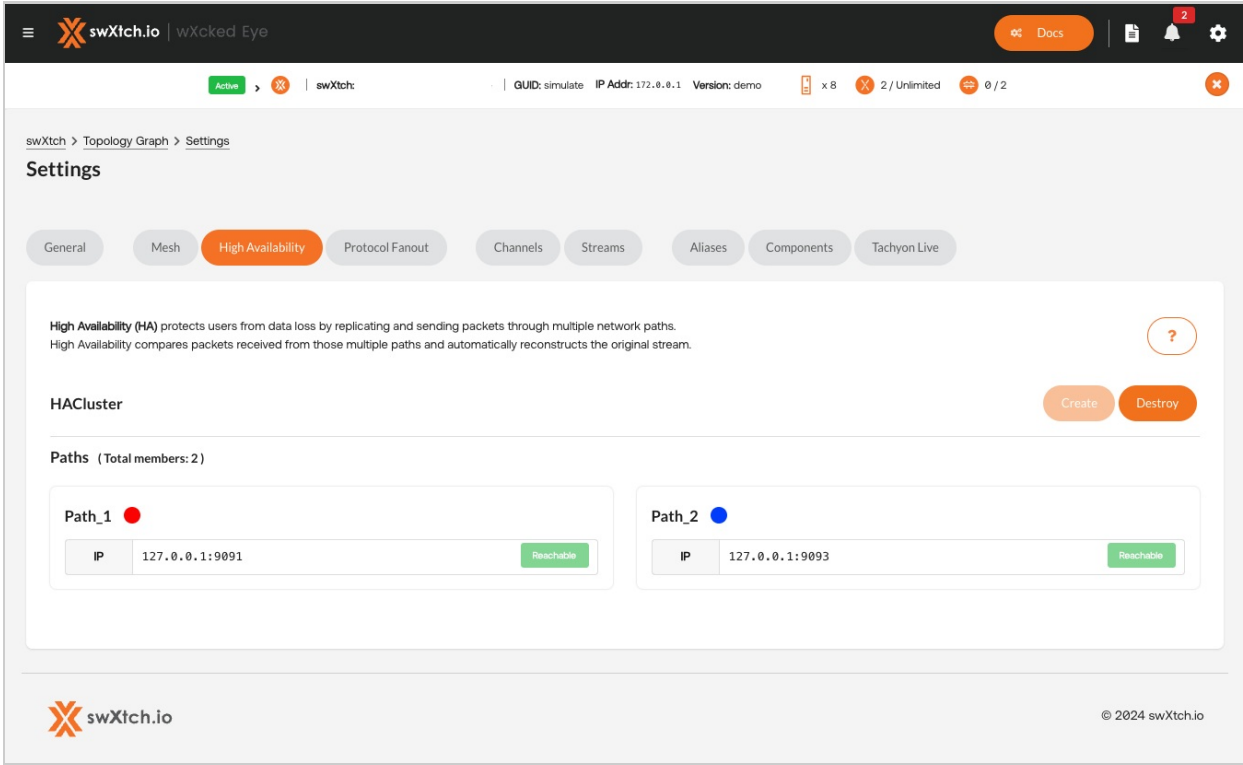
Navigating to the High Availability tab

The High Availability tab is located in the Settings page on wXcked Eye. To learn how to navigate there, please review the [Configure cloudSwXtch with wXcked Eye](#) article.

The High Availability tab is organized into 2 functions:

- [Create HA](#)
- [Destroy HA](#)

In this section, we will discuss each tab and how it offers a user additional control over their cloudSwXtch network.

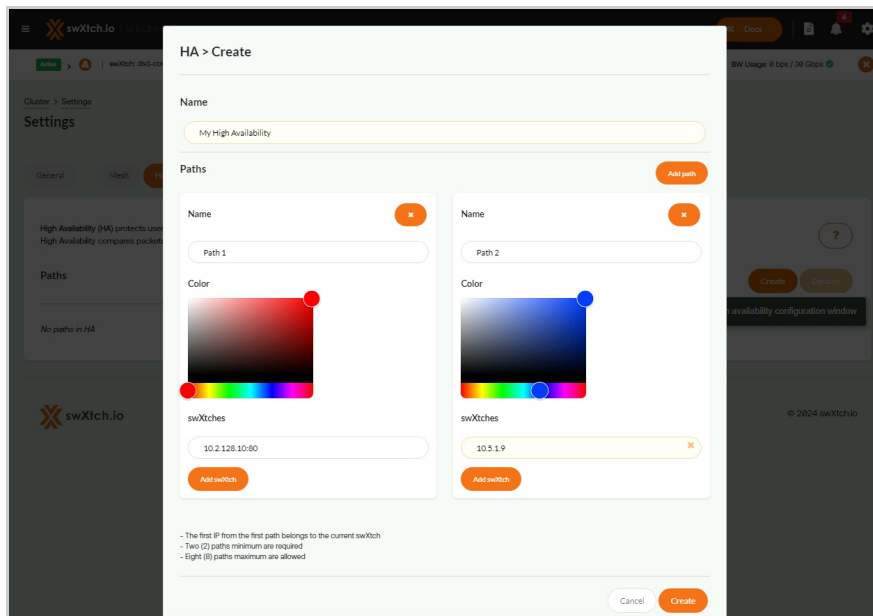


High Availability Command-Line Alternatives

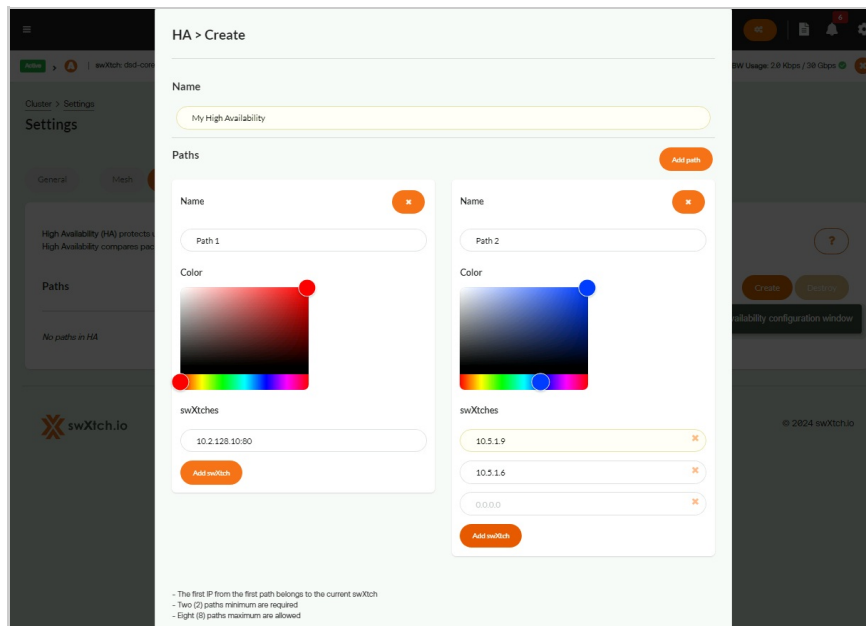
In addition to configuring your high availability through the wXcked Eye UI, users can also swXtch specific commands in their terminal. To learn more, please visit the [High Availability](#) article under Configuring cloudSwXtch.

Create HA

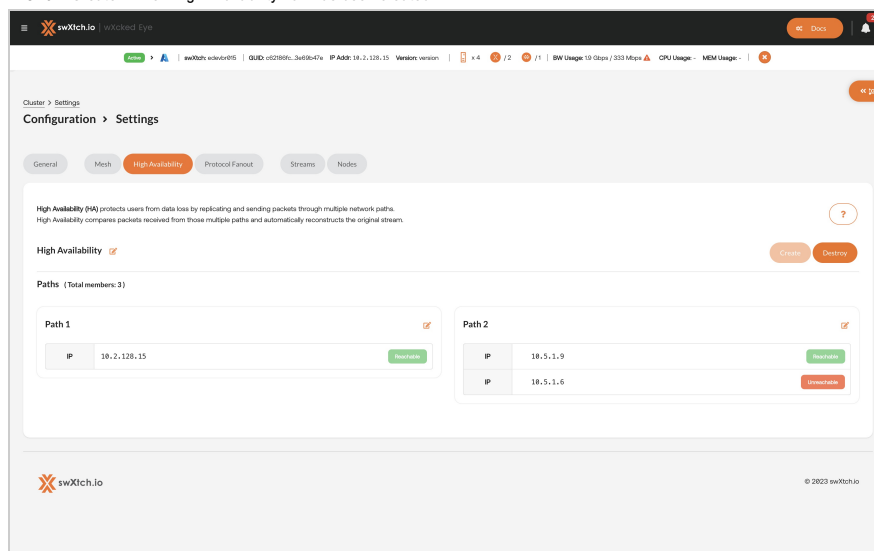
1. Click the "Create" button in the cloudSwXtch you wish to include in your high availability configuration. A pop-up will open.
2. **Name** your HA configuration. In this example, the HA is named "My High Availability."



3. **Name** the first path. The IP address of the cloudSwTch you are currently on will populate. Enter **IP addresses** for any additional relevant cloudSwTches.
4. **Name** the second path and enter the **IP addresses** for the relevant cloudSwTches.
- Note:** You must have *more than 1* path in order to have a working HA flow.
5. **OPTIONAL:** Add an additional cloudSwTch to a Path by clicking "Add SwTch." In this example, the user assigned 2 cloudSwTches to Path 2.
6. Assign a color for both paths. Here, the user selected Red for Path 1 and Blue for Path 2.



7. Click "Create." A new High Availability flow has been created.



With High Availability now configured, users can switch between different cloudSwXtches in their HA, refresh the HA page and see the members listed with their associated paths. For example, if a user were to look at the wXcked Eye for cloudSwXtch 10.5.1.6 instead of the above 10.2.128.10, they will see the same My High Availability member list.

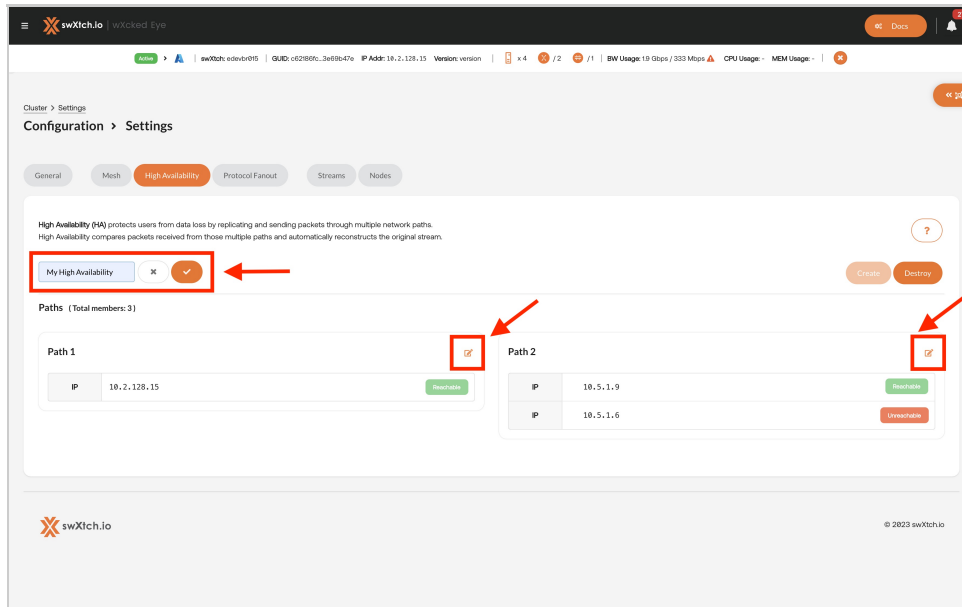
From any of the connected cloudSwXtches, users can destroy their HA configuration.

Warning

If you try to create another HA, the wXcked Eye UI will destroy the current HA and replace it with the new configuration.

Renaming High Availability and Path Names

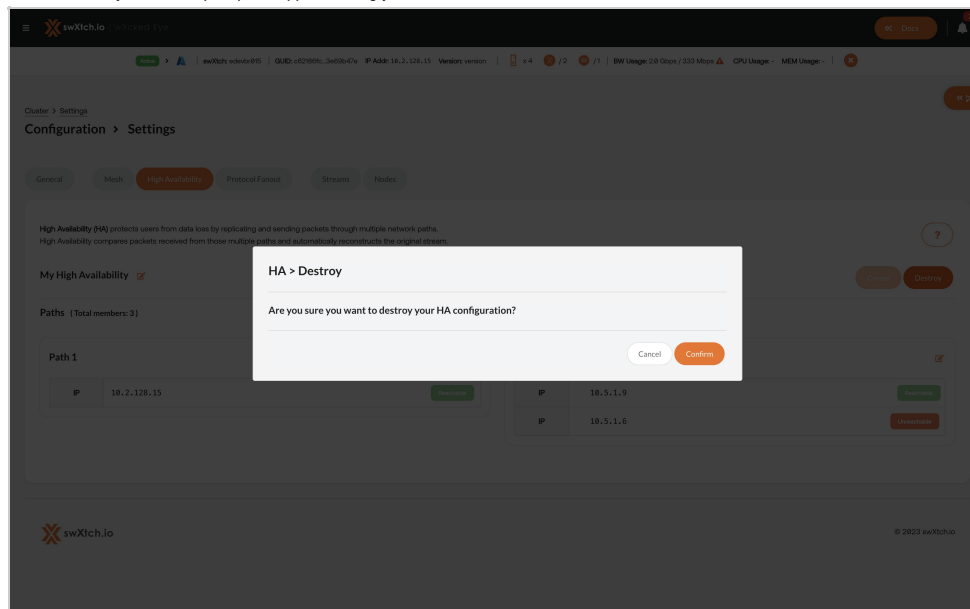
Users can dynamically change their high availability and paths' names directly in the UI. To do this, simply click the edit button next to the paths.



Destroy HA

To destroy an HA configuration, a user will need to go to the HA page of any associated cloudSwXtches.

1. Click "Destroy HA." A new prompt will appear, asking you to confirm the action.



2. Select "Confirm."

Your HA is now destroyed. All associated cloudSwXtches will show a blank list for HA and your Cluster page will be empty.

Removing a cloudSwXtch from an HA configuration

To remove a cloudSwXtch from a user's HA configuration, they will need to delete and recreate their HA cluster without that cloudSwXtch.

Protocol Conversion and Fanout with wXcked Eye

WHAT TO EXPECT

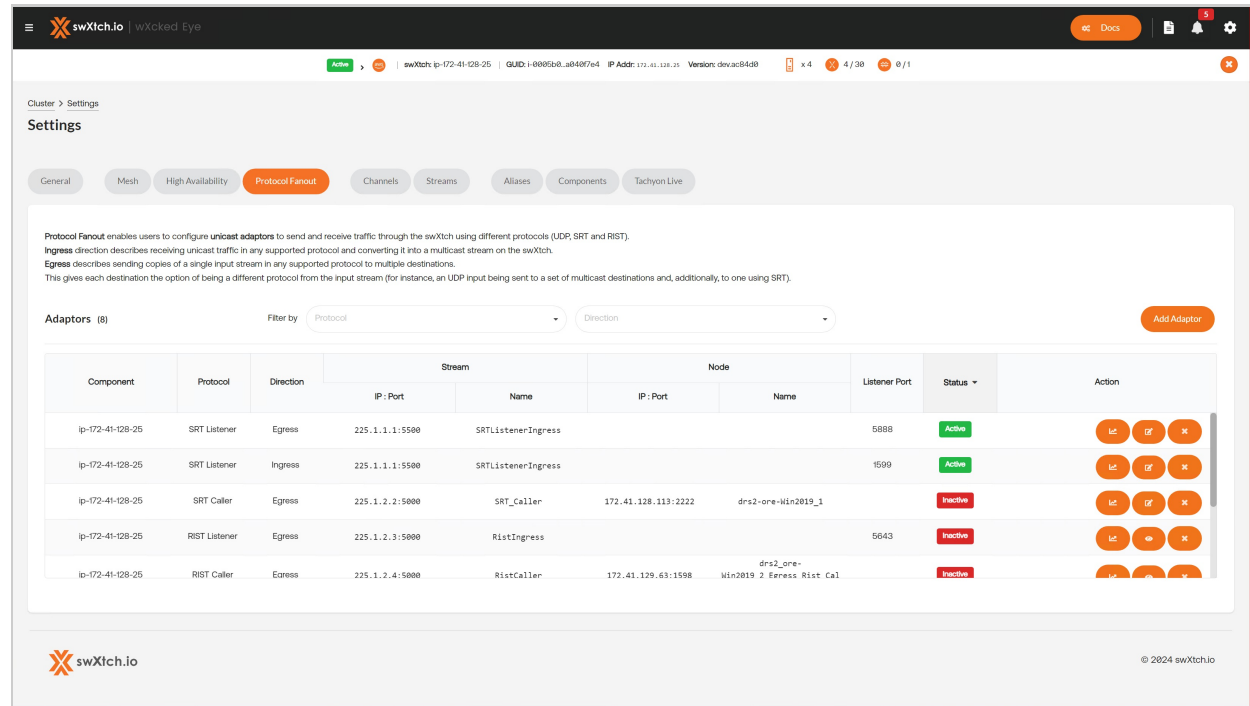
The Protocol Fanout feature allows users to configure unicast (UDP, SRT, RIST) for the cloudSwXtch and the cloudSwXtch Bridge and fan it out to multiple instances as UDP, SRT, RIST or multicast.

Configuration for Protocol Fanout for cloudSwXtch and cloudSwXtch Bridge can be done in wXcked Eye from the Settings page and from [the Topology Graph](#). To learn more about how to navigate there, please review the [Configure cloudSwXtch with wXcked Eye](#) article.

In this article, users will learn how to establish UDP, SRT and RIST connections via the Protocol Conversion and Fanout feature on wXcked Eye. It can also be used to convert multicast into one or more of these protocols. For example, imagine you have an SRT stream coming into the cloudSwXtch with five different clients requiring different protocols (one needing SRT, another RIST, another UDP and the last two for multicast). This can be accomplished by using this tool. For a walkthrough on configuring Protocol Conversion and Fanout in wXcked Eye, see the [Protocol Conversion and Fanout Example](#) article.

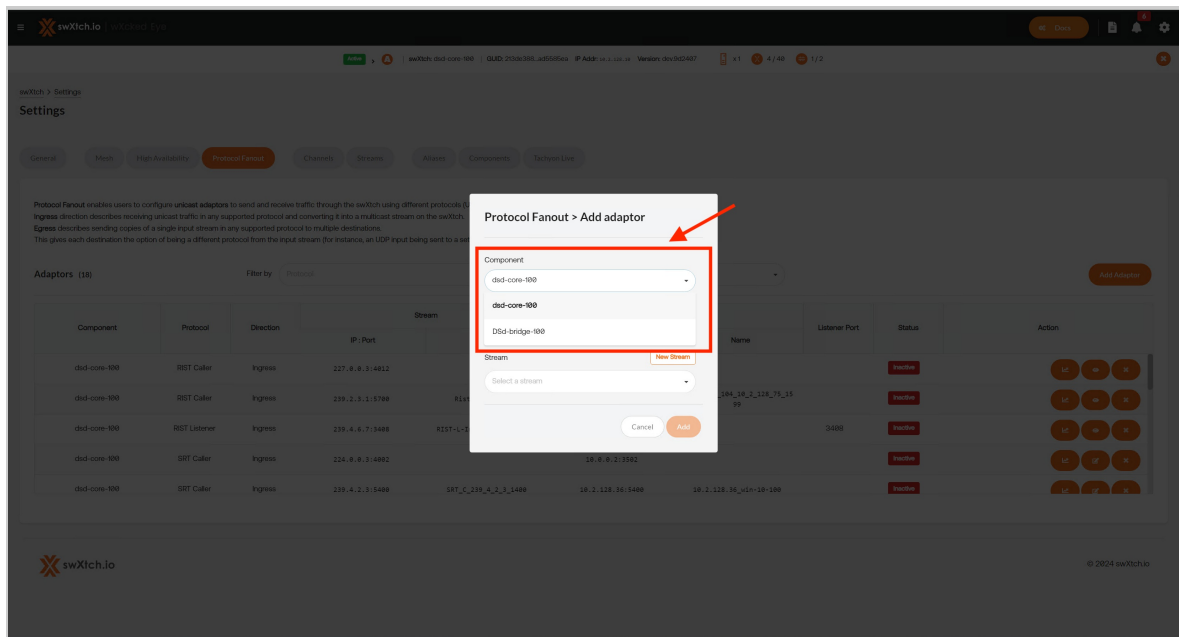
Setting Up Aliases

The Protocol Fanout tab utilizes Stream and Node names set up in the Aliases tab. For more information on how to do this, please see the [Aliases](#) article.



Protocol Fanout and Conversion is a cloudSwXtch feature that allows users to send copies of a single input stream in any supported protocol to multiple destinations. In wXcked Eye, users can send/receive UDP traffic or establish SRT/RIST caller/listener connection methods.

To add an adaptor, select "Add Adaptor." Here, a user can select either a cloudSwXtch or a cloudSwXtch Bridge (only if it is installed) as the component.



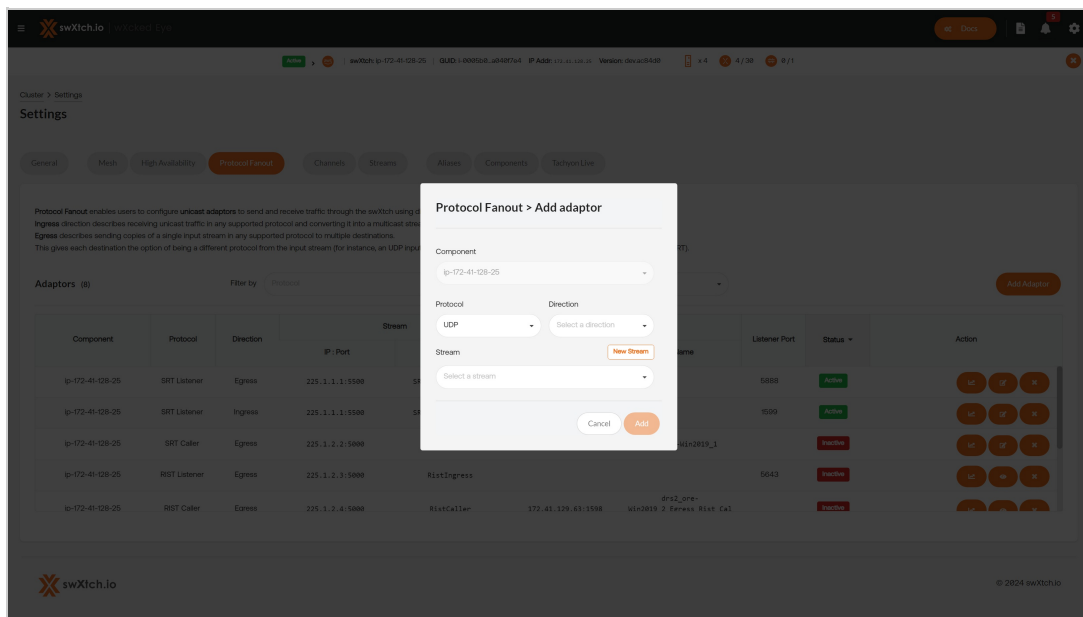
Then, users can select between three different protocols for their adaptor: UDP, SRT, and RIST.

UDP

Ingress vs. Egress

Differentiating between ingress and egress can be difficult. It is important to imagine it in relation to the cloudSwXtch.

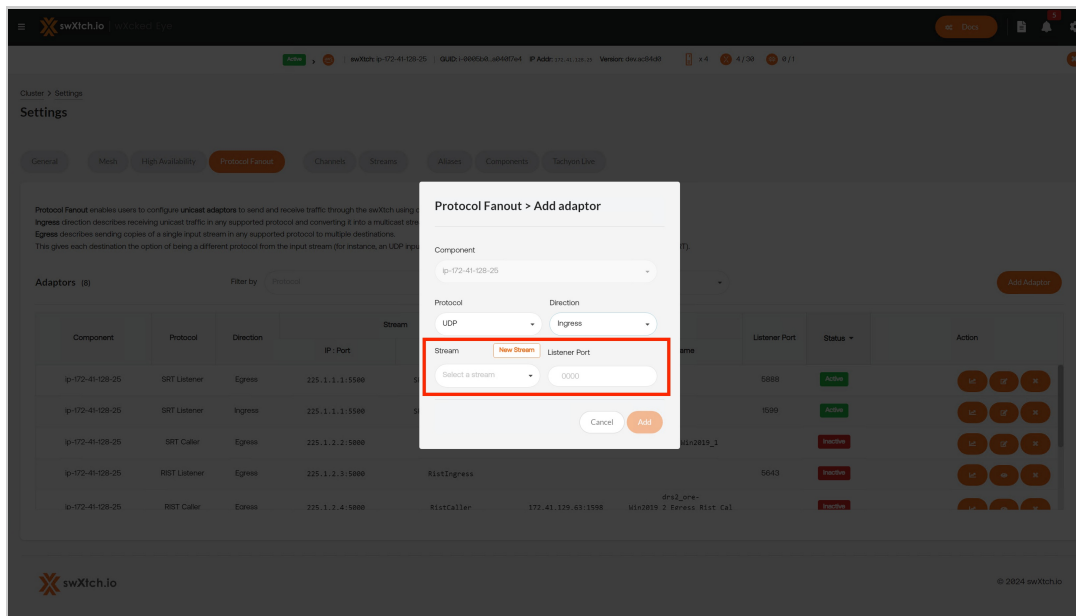
- **Ingress:** Data is coming into the cloudSwXtch.
- **Egress:** Data is leaving the cloudSwXtch.



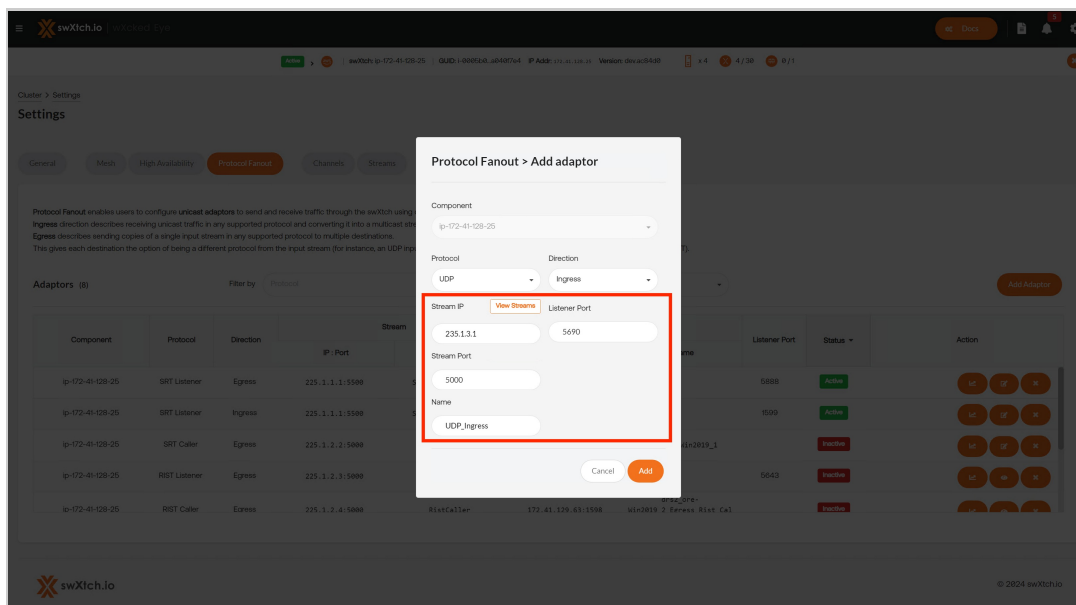
Selecting UDP under Protocol allows users to add mapping for UDP traffic entering and leaving the cloudSwXtch. Depending on the direction of the data, a user will have to add additional information to set up a successful connection.

UDP Ingress

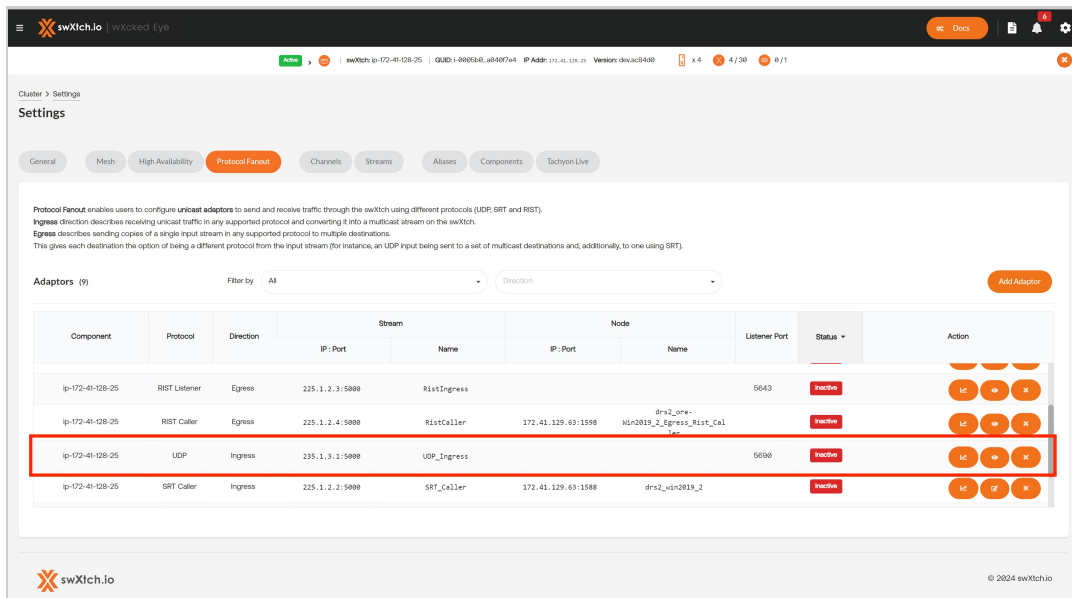
For Ingress, a user will select one of the **Streams** created in the **Streams** tab from the dropdown and designate a **Listener Port**.



Alternatively, if a user does not have an alias set up for their desired stream, they can manually specify a **Stream IP**, **Listener Port**, a **Stream Port** and an **alias name** to add one. To do this, they would have to select the **New Stream** option in the panel.



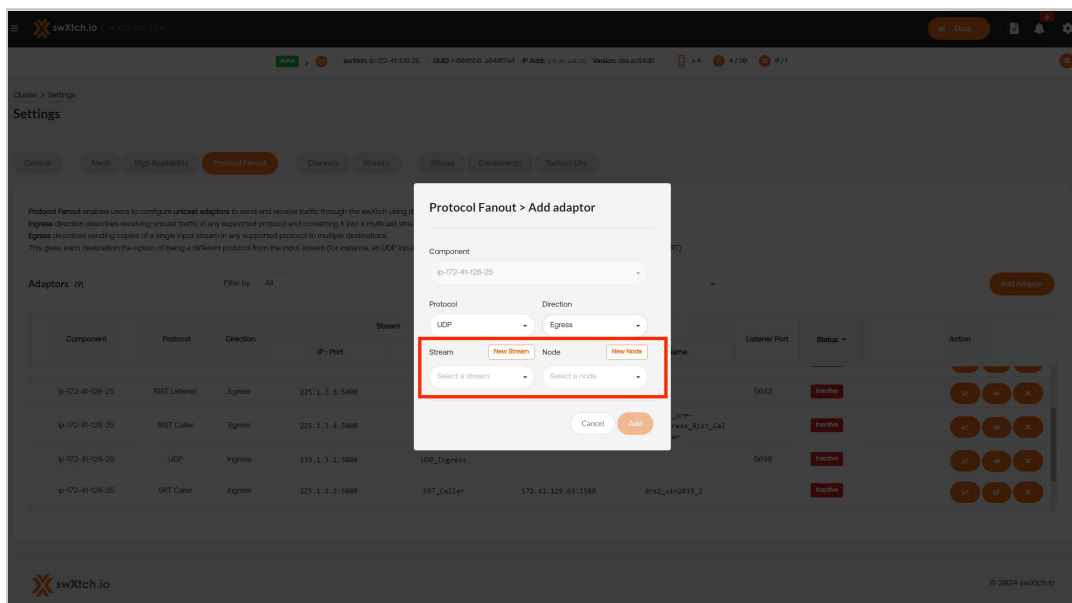
By clicking Add, the new adaptor will appear in the list on the Protocol Fanout tab.



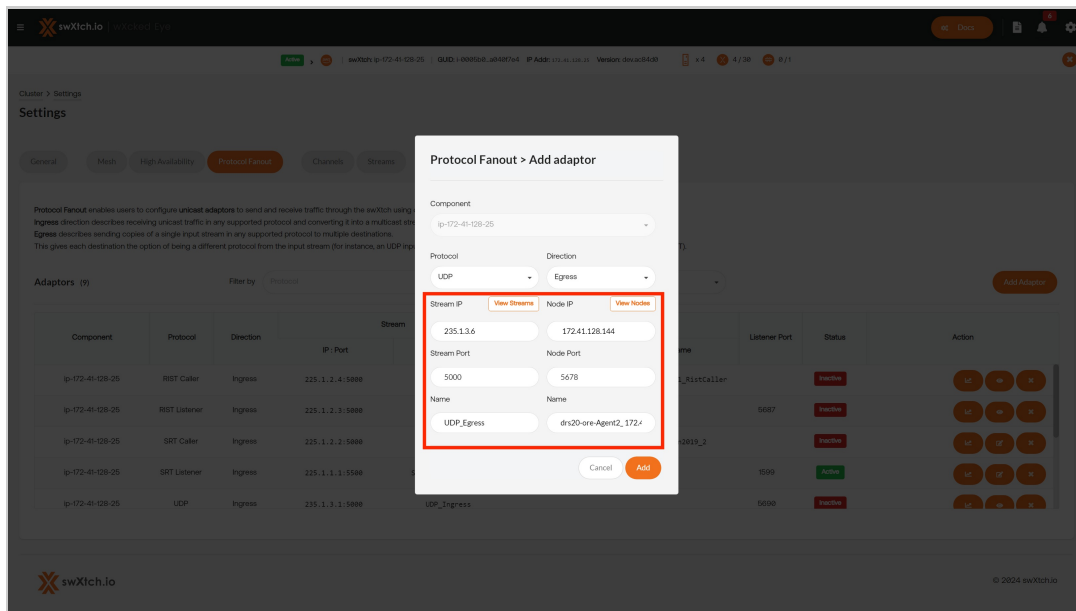
Both methods will allow endpoints to send unicast data. Once that connection has been established, the cloudSwXtch will be able to ingest the unicast data as multicast.

UDP Egress

For Egress, a user will set the parameters for fanning out a multicast stream as unicast. To do this, the cloudSwXtch would need to select a **Stream** and **Target Node** from their respective **Alias** dropdown.



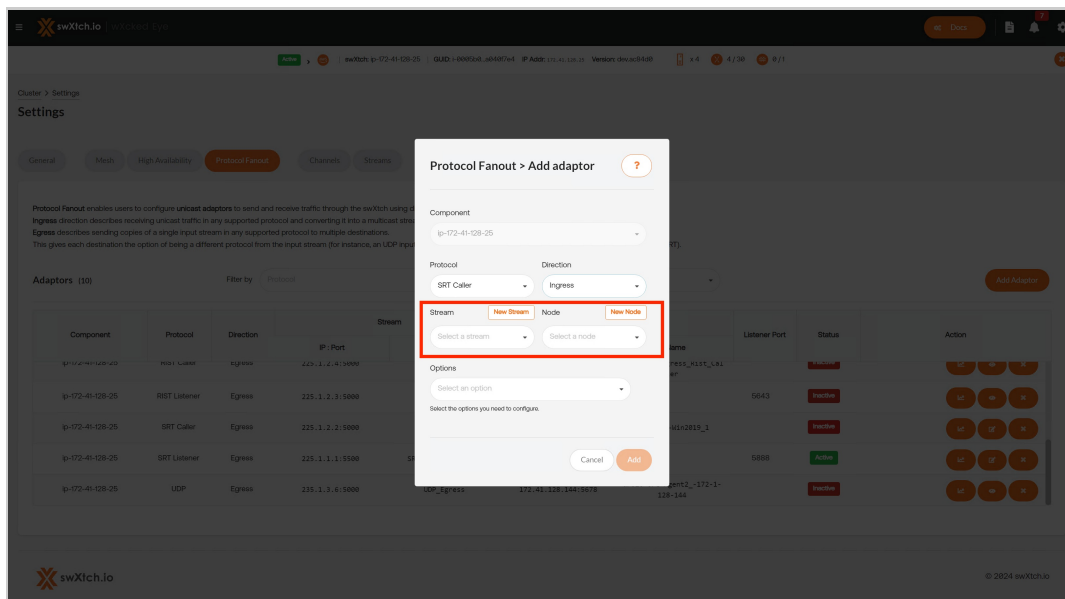
If an **Alias** was not created prior, they can manually add a **New Stream** and **New Node** by entering the appropriate information for both. To create a **New Stream**, a user will need the **Stream IP**, **Stream Port** and **Alias name**. Similarly, a **New Node** would require a **Node IP**, **Node Port** and **Alias name**.



Whether it is from existing Aliases or ones manually created, this will allow the cloudSwXtch to transmit a multicast stream as unicast to a desired endpoint.

SRT and RIST Caller

To set up SRT or RIST Caller, a user will need to choose what direction they will like their data to flow: Ingress or Egress. Regardless of their choice, both Ingress and Egress requires a selection for Stream and Node. These can either be selected from the Alias dropdown (assigned in the Alias tab under Settings) or manually created in the panel.



If a user wants to manually create a stream and node, they will need to select the "New Stream" and "New Node" buttons. This will reveal additional fields necessary for both. For Stream, a user will need to enter a Stream IP, Stream Port and Alias name. For Node, they will need a Node IP, Node Port and Alias. This will act as a Target Node. It is the source of where the traffic will be coming from outside the cloudSwXtch. This information is crucial since it will dictate where the cloudSwXtch sends the caller message.

Protocol Fanout > Add adaptor

Component: ip-172-41-128-25

Protocol: SRT Caller Direction: Ingress

Stream IP: 236.1.1.1 Node IP: 172.41.128.144

Stream Port: 5000 Node Port: 4568

Name: SRTCallee Name: drs2-one-agent2

Options: Select the options you need to configure.

Cancel Add

The **Options** dropdown allows for additional fine tuning based on standards of the protocol. Selection of an option will open another field.

After filling out all the required fields, select "Add." The cloudSwXtch will then call out to the target source and receive multicast traffic through the designated node.

For **Egress**, the user will be specifying the Target Node for an endpoint to receive an SRT or RIST stream from the cloudSwXtch. The cloudSwXtch will then call to the Destination or Target Node to establish a connection before transmitting the stream.

SRT and RIST Listener

Similar to the SRT or RIST Caller panel, Listener requires the user to specify the direction of their data flow: Ingress or Egress. However, what differs is that the SRT or RIST Listener is essentially "listening" for any incoming messages from endpoints ready to send/receive SRT or RIST data. This method of transmission is considered to be more user-friendly since a user will not have to worry about pointing to a specific IP address. It places the burden of targeting the endpoint instead.

Protocol Fanout > Add adaptor

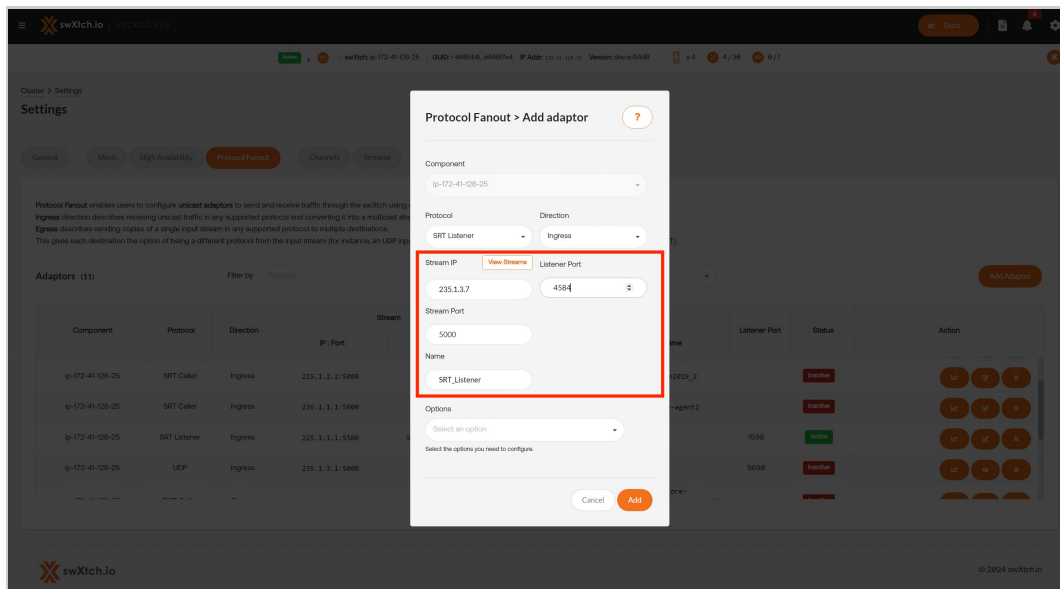
Protocol: SRT Listener Direction: Ingress Options: Select an option

Stream: Select an alias Listener Port: 0000

Cancel Add

Both Ingress and Egress will require a user to select a multicast Stream from the Alias dropdown. Stream Aliases are assigned in the Alias tab under Settings.

Alternatively, users can enter a new Stream by selecting the New Stream button and entering the following information: Stream IP, Stream Port and Alias Name. In addition to the Stream, a user will also need to specify a Listener Port where an endpoint can send data through.



The **Options** dropdown allows for additional fine tuning based on standards of the protocol. Selection of an option will open another field.

For Ingress, once the configuration is complete, the cloudSwXtch will now listen for producers of SRT or RIST traffic who connect to the port. When a connection has been established, the cloudSwXtch will begin ingesting data.

Likewise for Egress, the cloudSwXtch is listening for endpoints that are trying to receive SRT or RIST data. From there, depending on the user's bandwidth, they can create up to 32 Listener ports from which an endpoint can connect to the the steam. By setting the necessary parameters, a consumer will then be able locate a target port and begin streaming data from the cloudSwXtch.

Protocol Conversion and Fanout Example

Protocol Conversion and Fanout Example

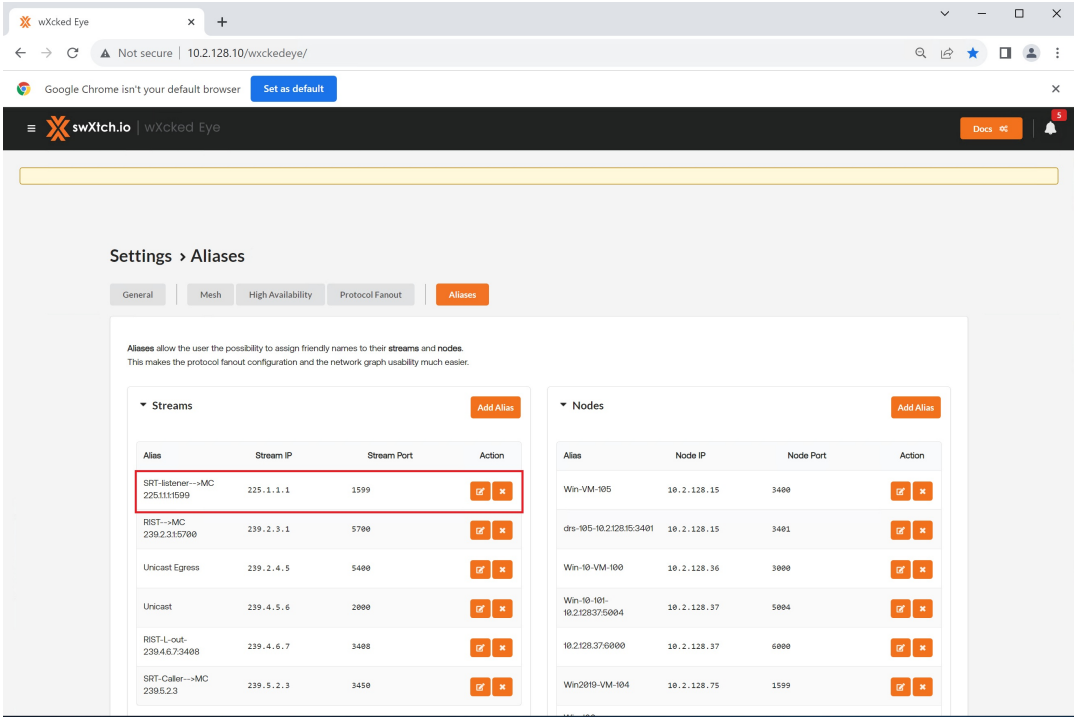
WHAT TO EXPECT

Navigating the Protocol Fanout and Conversion tab in wXcked Eye can be a little confusing when first starting out.

In this article, we will walk you through a typical SRT Listener configuration workflow to explain the various pieces that go into setting it up. We will look at the differences between ingress and egress and what that means in relation to the cloudSwXtch.

Step One: Setting up Your Aliases

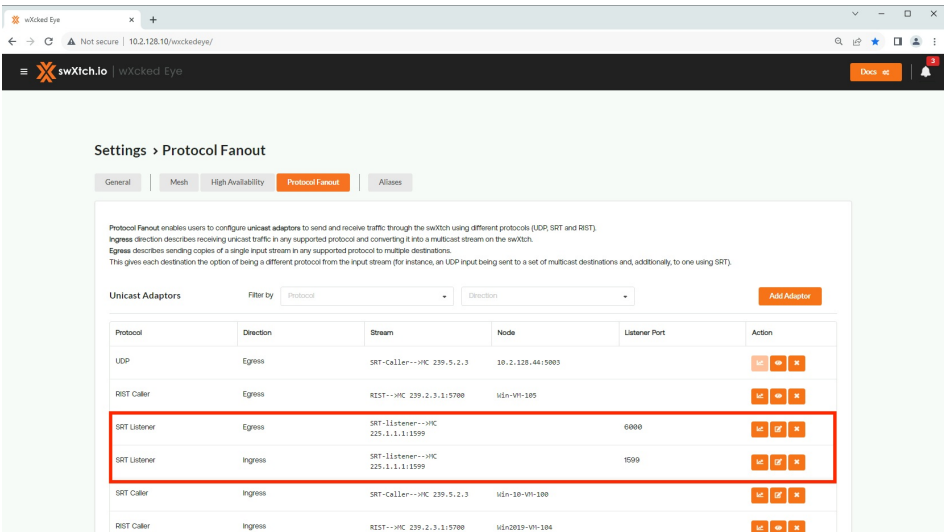
The Aliases tab allows users to set friendly names or "aliases" for their streams and nodes so that it is easier to organize them in the Protocol Fanout tab. In this example, the user has named their stream **SRT-listener -> MC 225.1.1.1:1599**, inputting a stream IP of 225.1.1.1 and a Stream Port of 1599. This name is helpful for the user because it illustrates what they hope to do when setting up for Protocol Fanout. They are going to set up for an SRT to MC conversion using the stream IP and Stream Port assigned to the name.



For more information about how aliases work, see the [Aliases](#) article under Configure cloudSwXtch with wXcked Eye.

Step Two: Adding Adaptors

In the Protocol Fanout Settings page, the user has set up two SRT Listeners. An SRT Listener configuration is telling the cloudSwXtch to listen for any incoming messages from endpoints ready to send/receive SRT data. This method of transmission is considered to be more user-friendly since a user will not have to worry about pointing to a specific IP address. It places the burden of targeting on the endpoint instead.

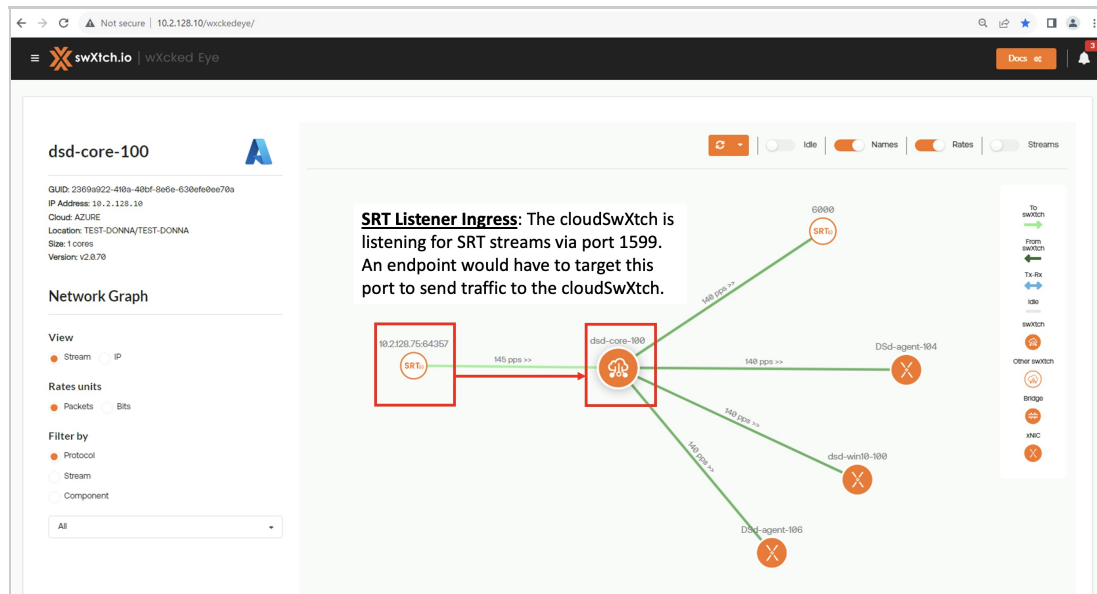


The user has set up SRT Listener for both Egress and Ingress using the Alias assigned earlier: **SRT-Listener -> Multicast 225.1.1.1:1599**. When differentiating between Egress and Ingress, always imagine it from the perspective of the cloudSwXtch:

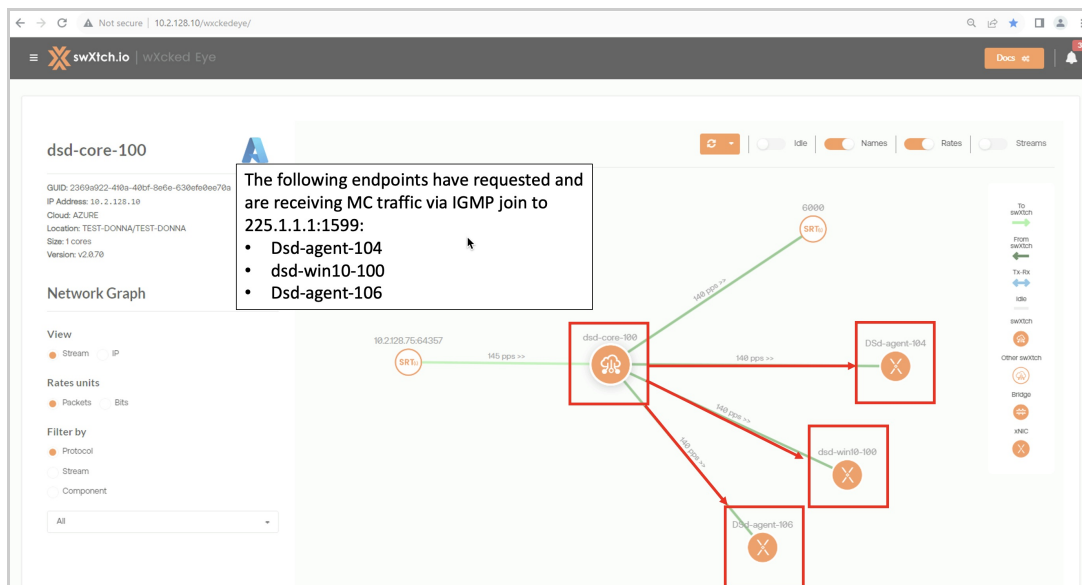
SRT Listener Ingress

For **Ingress**, think about the cloudSwXtch **Ingesting** a stream. The user has set up an SRT Listener Ingress using the **SRT-Listener -> Multicast 225.1.1.1:1599** stream with **Listener Port 1599**. That means that an endpoint will have to target port 1599 to send SRT traffic to the cloudSwXtch. Since it is ingress, the cloudSwXtch will automatically convert the SRT stream it receives into multicast.

Using the **Topology** in wXcked Eye, you can see how SRT Listener Ingress is set up in relation to the cloudSwXtch below.

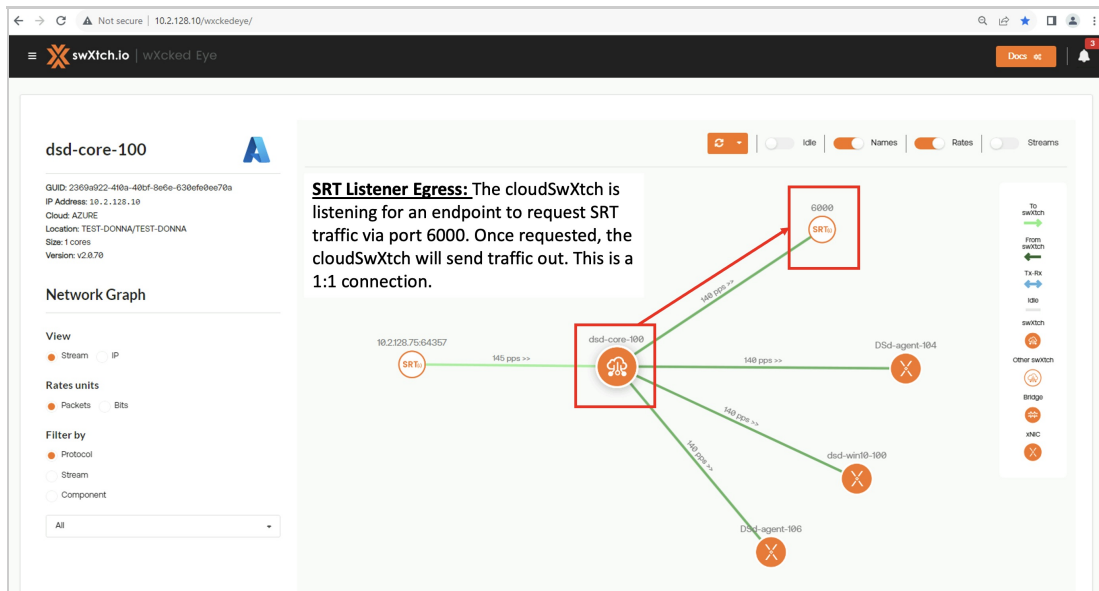


The endpoints highlighted will then request and receive multicast traffic via IGMP join to 225.1.1.1:1599.



SRT Listener Egress

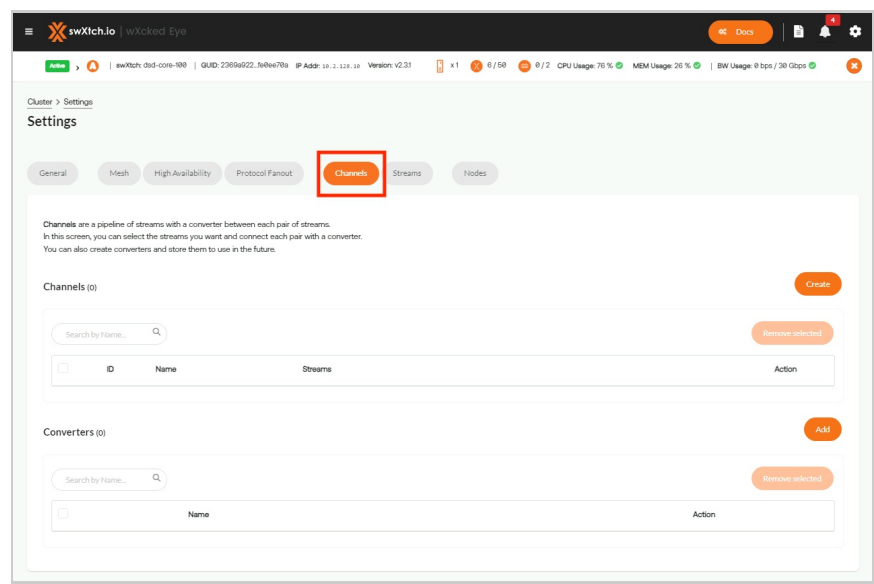
For **Egress**, imagine the stream **EXITING** the cloudSwXtch (->). In this example, the user has set up SRT Listener Egress using the **SRT-Listener -> Multicast 225.1.1.1:1599** stream and opening Listener Port 6000. That means that an endpoint will have to target port 6000 and let the cloudSwXtch know that it would like to receive SRT traffic. Note that **this is a 1:1 connection**, meaning only one SRT endpoint can use the listener port. In the example below, an endpoint has requested the SRT traffic and the cloudSwXtch is sending it out 140pps via port 6000.



Channels

WHAT TO EXPECT

The Channels tab on the Settings page allows users to create and remove channels from wXcked Eye. In addition, users can add and remove converters, which connect streams together to form a Channel.

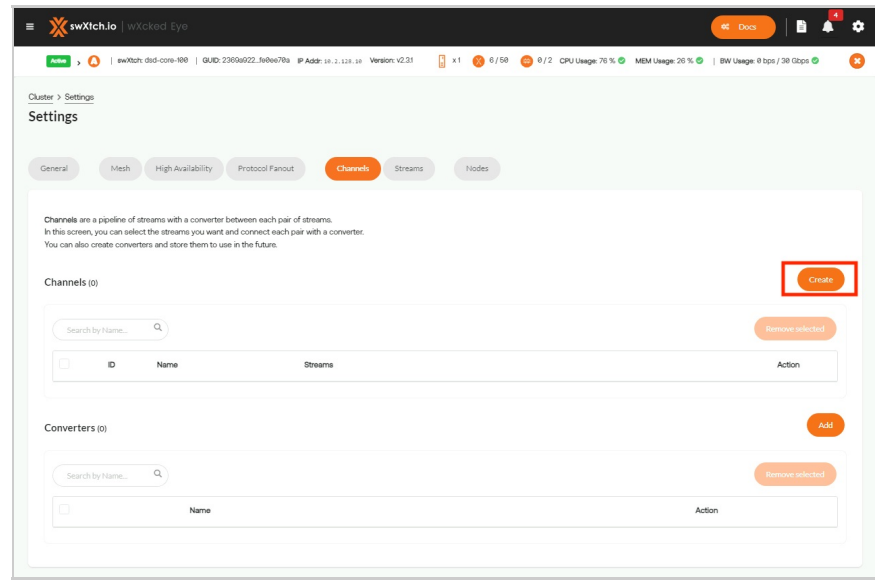


Channels

Channels are a pipeline of streams with a converter between each of them. On this page, a user will be able to Create Channels from a list of available streams and note the converters that connect them.

Creating Channels

1. Click **Create** above the Channels section. A new window will open.



2. Assign a name to your new Channel.

Channels > Create

Create your Channel with the name and streams you want.
 You can select a stream from the Available Streams table or create a new one.
 NOTE: Available Streams includes streams involved in the swtch network and also names defined by the user.

My Channel

Added Streams (7)

10.2.131.255:137 M → Converter 1 GF → 10.2.131.255:138 M → Converter 2 GF → 10.2.195.255:137 M → Converter 3 GF → 10.2.195.255:138 M

Available Streams (13)

☐ Show excluded streams

Search stream...

New Stream

Stream	Name
<input checked="" type="checkbox"/> 10.2.131.255:137	
<input checked="" type="checkbox"/> 10.2.131.255:138	
<input checked="" type="checkbox"/> 10.2.195.255:137	
<input checked="" type="checkbox"/> 10.2.195.255:138	
<input type="checkbox"/> 224.2.2.2:5000	test2
<input type="checkbox"/> 225.0.0.1	JPEG_XS_UHD
<input type="checkbox"/> 225.1.1.1:1599	CS_SRT_Lis from_dsd-agent-104
<input type="checkbox"/> 226.0.0.1	Spanish_audio
<input type="checkbox"/> 239.1.1.1:5004	
<input type="checkbox"/> 239.2.3.1:5700	RIST-C-239.2.315700
<input type="checkbox"/> 239.4.2.3:5400	SRT-Caller--MC_239.4.2.3
<input type="checkbox"/> 239.4.6.7:3400	Rist-L-ing-239.4.6.7.3400
<input type="checkbox"/> 239.255.255.250:1900	

Cancel Create

3. Select the streams that you wish to add to the Channel configuration from the Available streams list.

Channels > Create

Create your Channel with the name and streams you want.
 You can select a stream from the Available Streams table or create a new one.
 NOTE: Available Streams includes streams involved in the swtch network and also names defined by the user.

My Channel

Added Streams (7)

10.2.131.255:137 M → Converter 1 GF → 10.2.131.255:138 M → Converter 2 GF → 10.2.195.255:137 M → Converter 3 GF → 10.2.195.255:138 M

Available Streams (13)

☐ Show excluded streams

Search stream...

New Stream

Stream	Name
<input checked="" type="checkbox"/> 10.2.131.255:137	
<input checked="" type="checkbox"/> 10.2.131.255:138	
<input checked="" type="checkbox"/> 10.2.195.255:137	
<input checked="" type="checkbox"/> 10.2.195.255:138	
<input type="checkbox"/> 224.2.2.2:5000	test2
<input type="checkbox"/> 225.0.0.1	JPEG_XS_UHD
<input type="checkbox"/> 225.1.1.1:1599	CS_SRT_Lis from_dsd-agent-104
<input type="checkbox"/> 226.0.0.1	Spanish_audio
<input type="checkbox"/> 239.1.1.1:5004	
<input type="checkbox"/> 239.2.3.1:5700	RIST-C-239.2.315700
<input type="checkbox"/> 239.4.2.3:5400	SRT-Caller--MC_239.4.2.3
<input type="checkbox"/> 239.4.6.7:3400	Rist-L-ing-239.4.6.7.3400
<input type="checkbox"/> 239.255.255.250:1900	

Cancel Create

- a. Alternatively, select New Stream to add a new stream to the list. This will open a new window where you can name and assign a color to a Stream IP.

4. Edit the converter that links each stream by selecting the notepad by the placeholder.

Channels > Create

Create your Channel with the name and streams you want.
 You can select a stream from the Available Streams table or create a new one.
 NOTE: Available Streams includes streams involved in the swtch network and also names defined by the user.

My Channel

Added Streams (7)

10.2.131.255:137 M → Converter 1 GF → 10.2.131.255:138 M → Converter 2 GF → 10.2.195.255:137 M → Converter 3 GF → 10.2.195.255:138 M

Available Streams (13)

☐ Show excluded streams

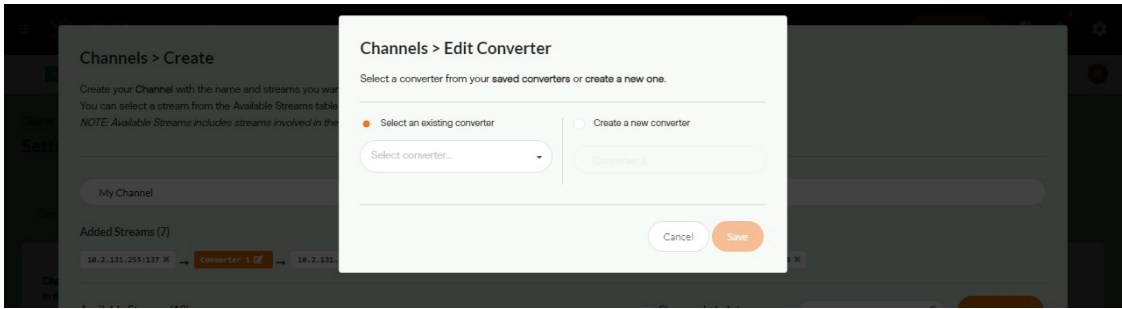
Search stream...

New Stream

Stream	Name
<input checked="" type="checkbox"/> 10.2.131.255:137	
<input checked="" type="checkbox"/> 10.2.131.255:138	
<input checked="" type="checkbox"/> 10.2.195.255:137	
<input checked="" type="checkbox"/> 10.2.195.255:138	
<input type="checkbox"/> 224.2.2.2:5000	test2
<input type="checkbox"/> 225.0.0.1	JPEG_XS_UHD
<input type="checkbox"/> 225.1.1.1:1599	CS_SRT_Lis from_dsd-agent-104
<input type="checkbox"/> 226.0.0.1	Spanish_audio
<input type="checkbox"/> 239.1.1.1:5004	
<input type="checkbox"/> 239.2.3.1:5700	RIST-C-239.2.315700
<input type="checkbox"/> 239.4.2.3:5400	SRT-Caller--MC_239.4.2.3
<input type="checkbox"/> 239.4.6.7:3400	Rist-L-ing-239.4.6.7.3400
<input type="checkbox"/> 239.255.255.250:1900	

Cancel Create

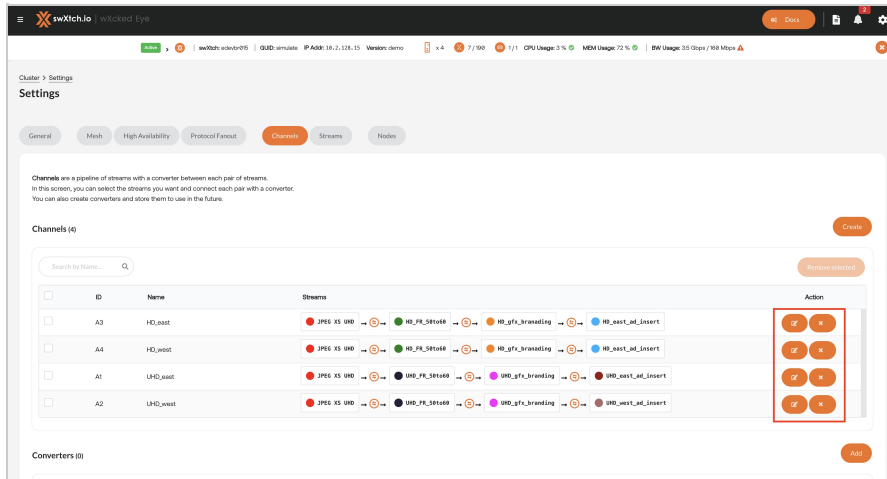
5. Select an existing converter from the dropdown menu.
 - a. Alternatively, select Create a new converter and name it.



b. **Please note:** wXcked Eye does not create these converters. This feature only allows users to name the existing converters in their network that will connect these streams.

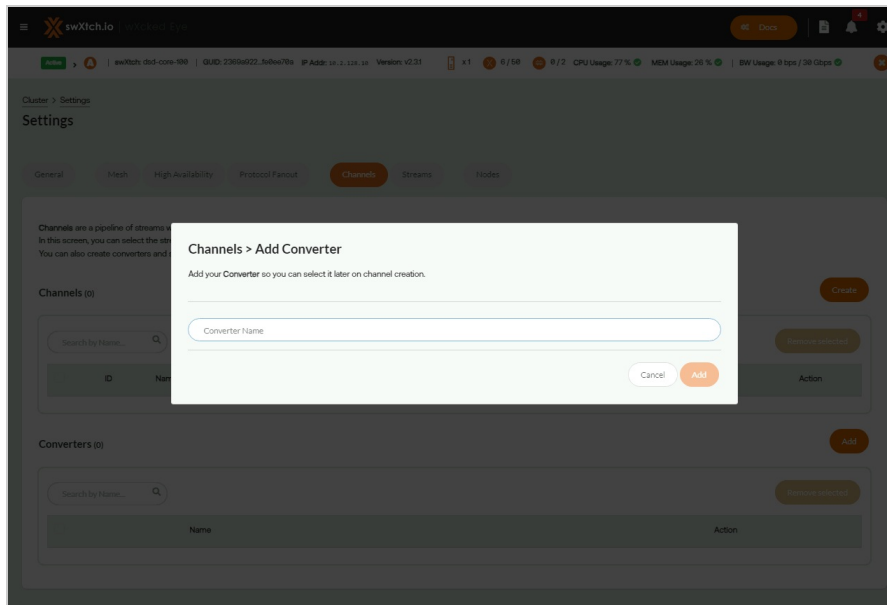
6. Click **Create**. A new Channel will appear.

wXcked Eye will automatically assign an ID to the Channel. These channel configurations will carry throughout the UI. Users can edit or remove the channel using the action buttons on the right hand side.

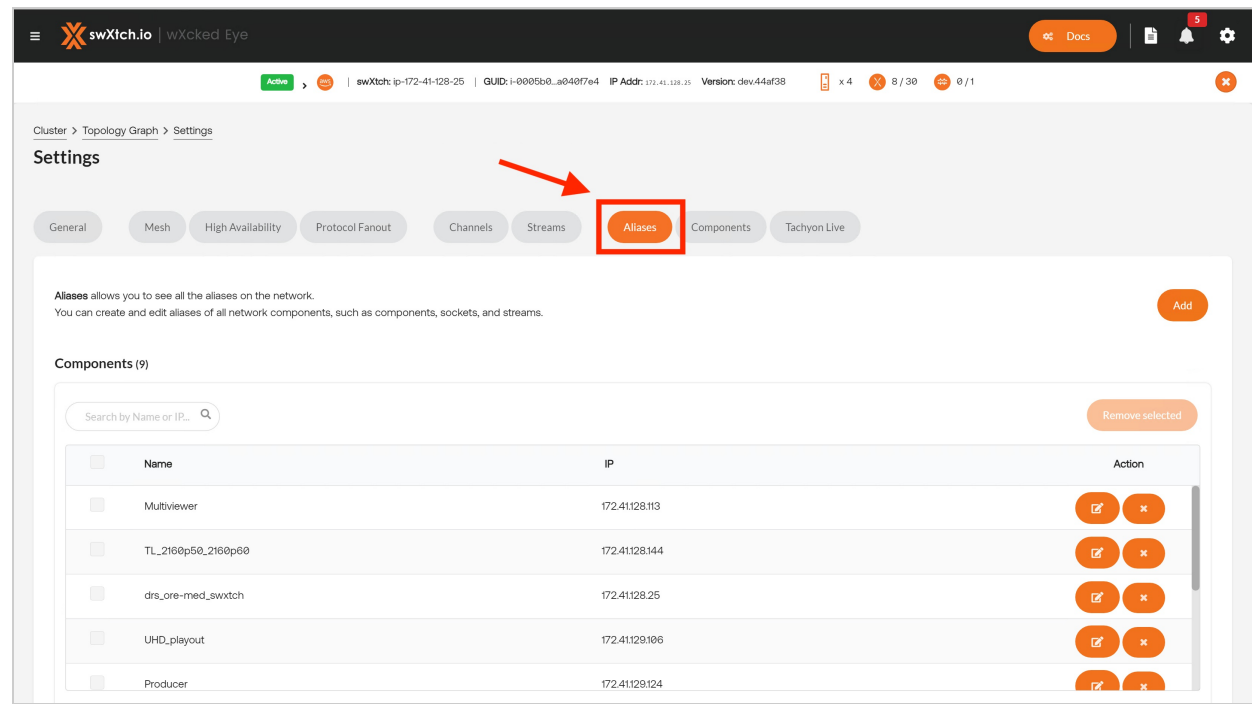


Adding Converters

Users can add a converter by clicking the Add button in the Converter section. A new window will open where users can name existing converters in their network. These can be a range of different transformations, such as Tachyon LIVE. **Please note:** wXcked Eye does not create converters. This feature only allows users to name a placeholder to connect streams visually in the UI.



Aliases



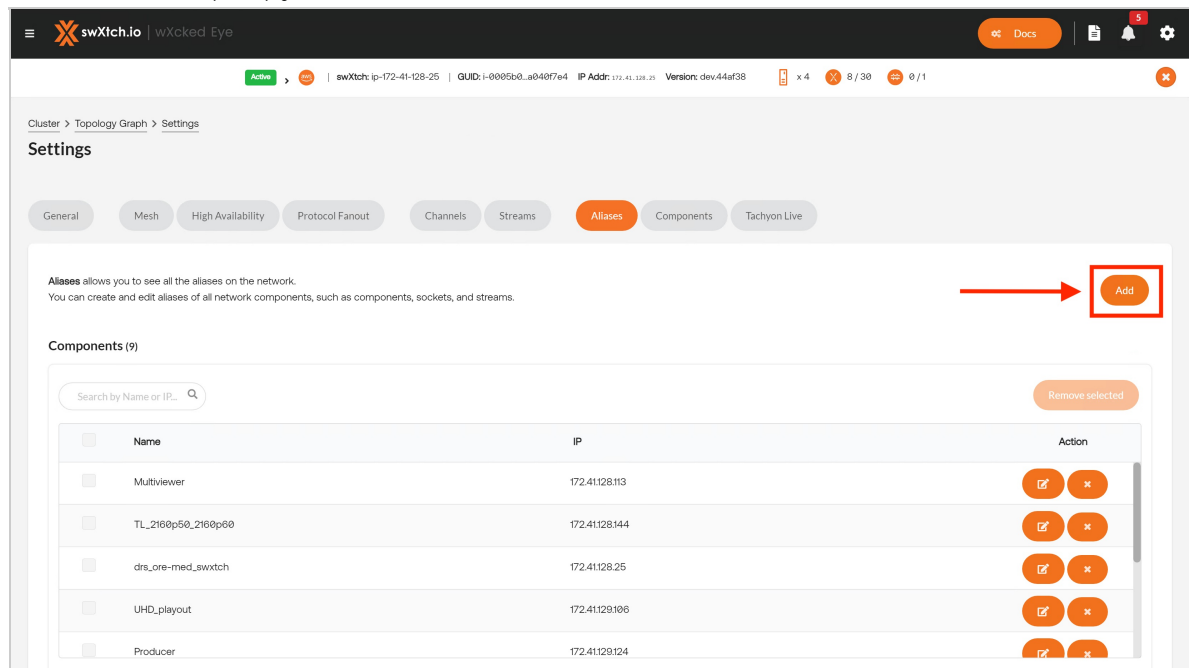
Assigning an Alias

Aliases are assigned user-friendly names for a component, stream, or socket that will be displayed throughout the wXcked Eye UI. While Components (cloudSwXtches, xNICs and cloudSwXtch Bridges) already display a name assigned for the VM during installation, a user can assign an alias for that component. This new name will appear in the Aliases tab.

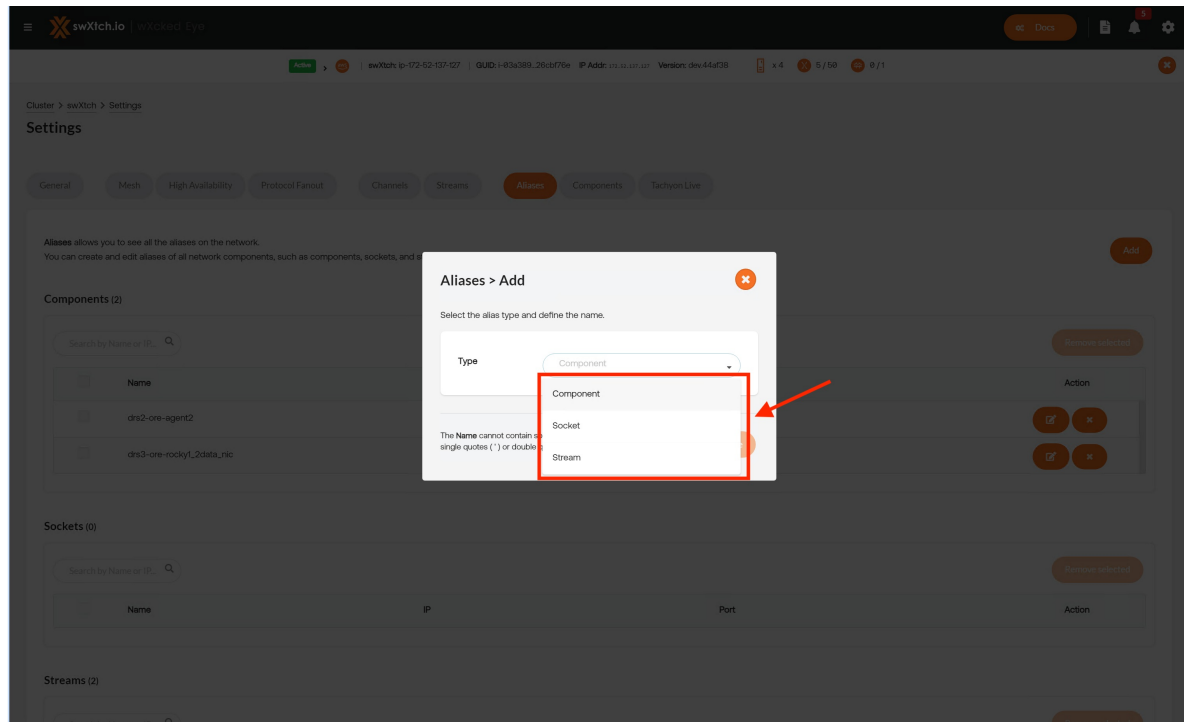
The same goes for Streams and Sockets. If a user changes the name of either, the new name will populate in their respective section in the Aliases tab.

Assigning an Alias to a Component, Socket, or Stream

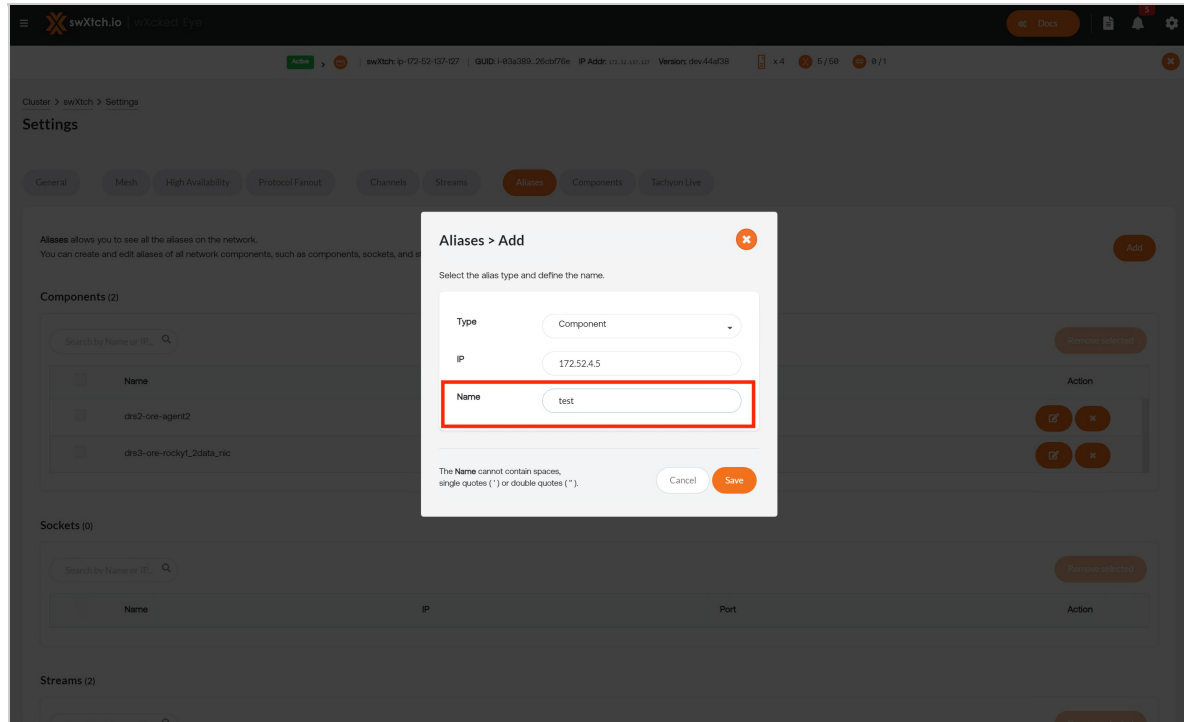
1. Go to the **Aliases** tab on the **wXcked Eye Settings** page.
2. Click on the **Add** button at the top of the page.



3. Select either **Component**, **Socket**, or **Stream** from the dropdown menu.



4. Enter a new **Alias** into the respective object's **Name** field. Be mindful that each Alias type will require different information.

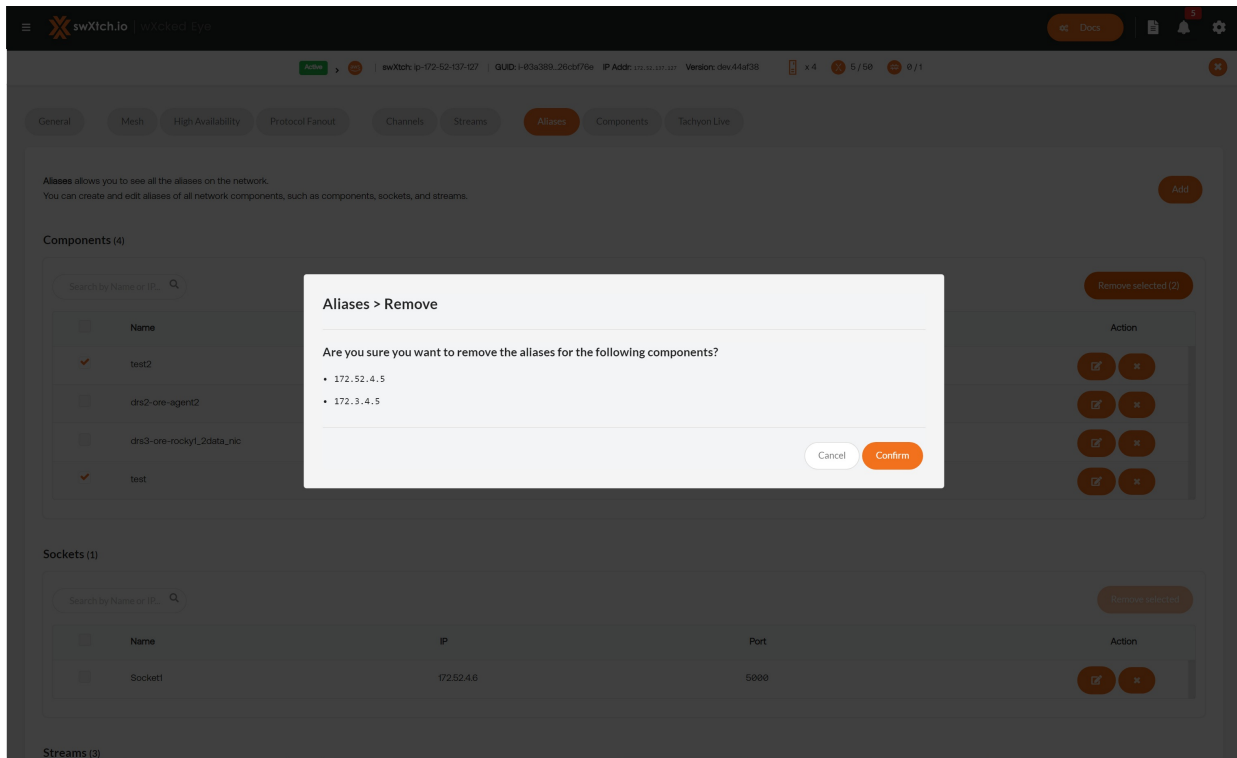


5. Select **Save** at the bottom of the window.

A new **Alias** should appear in the type's matching section. **Please note**: When assigning an alias to a component or a stream, the object will appear in their respective Settings tab.

Deleting an Alias

Deleting an **Alias** will not remove the component, socket or stream from the wXcked Eye UI. It will only remove the assigned name and the listing on the Aliases tab. **To delete**, select the X button next to the item or use the checkbox to select multiple. A prompt will appear, confirming deletion.



Alternate Ways to Assign an Alias to a Component

In addition to the Aliases tab, there are two additional ways a user can assign an alias for a Component: the **Topology Graph** and the **Settings' Component tab**. Altering the name in either of these pages will result in the addition of the component in the Aliases tab.

Alternative #1: Topology Graph

To assign an alias to a component in the Topology Graph:

1. Go to the Topology Graph.
2. Select one of the nodes in the Topology Graph. This will open the Component Information panel for that node.
3. Click the notepad button by the Component's current name. A new window will open.
4. Enter an Alias in the Name field.
5. Click Save.

The new name will appear in the Topology Graph for that node. It will also now list under Components in the Aliases tab in the wXcked Eye Settings page.

For more information, see the [wXcked Eye Topology Graph](#) article.

Alternative #2: Components Tab

In the Components tab under wXcked Eye's Settings page, swXtch.io related products (cloudSwXtches, xNICs, and cloudSwXtch Bridges) connected to the primary cloudSwXtch you're viewing will automatically populate the Components list. This forms the cloudSwXtch network displayed in the Topology Graph. Each of these existing components will have names assigned during VM creation.

Next to each component in the Components tab, **there is an edit button**. Selecting it will open a new window where a user can assign an Alias for their component. This will change the name of the component in the Topology Graph and add the name in the Aliases tab.

For more information, see [Components](#) under Configure cloudSwXtch with wXcked Eye.

Alternate Ways to Assign an Alias to a Stream

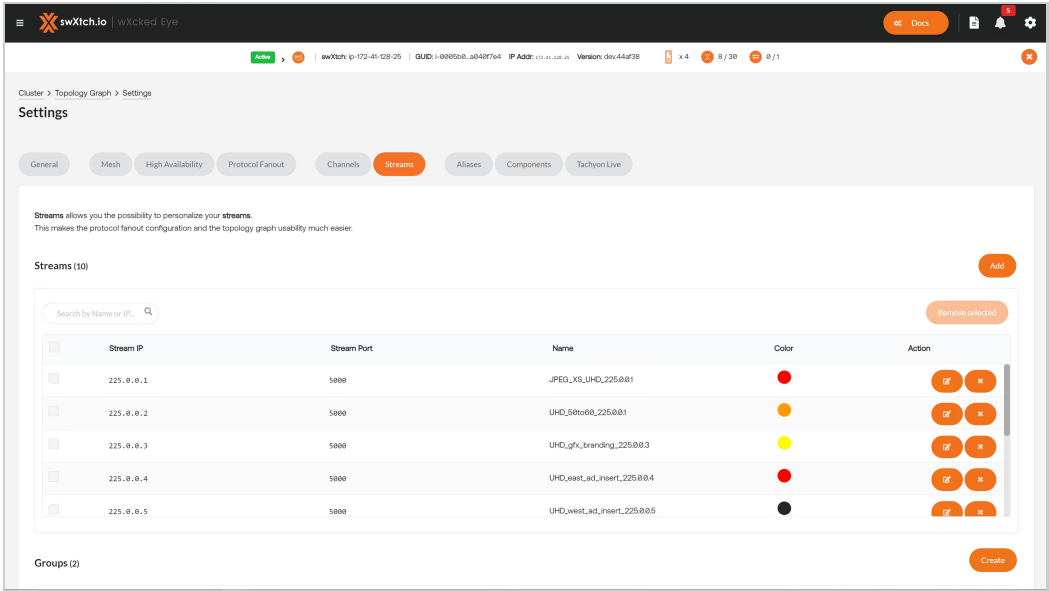
In addition to the Aliases tab, a user can also edit a stream's name in the Streams tab on the wXcked Eye Settings page. Any changes to a stream's name or the creation of a stream name in the Streams tab will result in its appearance in the Aliases tab.

Streams

WHAT TO EXPECT

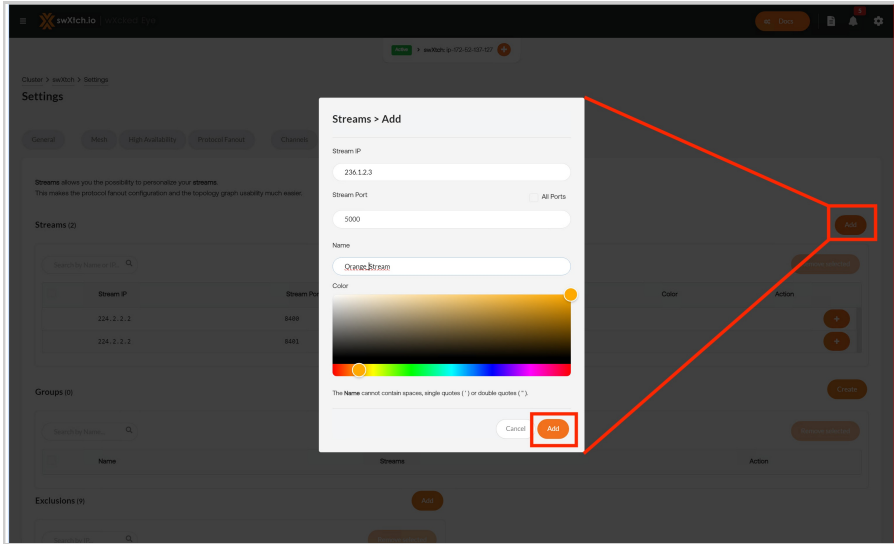
In this article, users will learn how to use the Streams tab under Settings to configure and personalize streams in the wXcked Eye UI.

Navigating the Streams Tab



The Streams tab allows users to name their Multicast IP addresses and assign them a color. This enables users to readily distinguish between different streams in the Topology Graph. In addition, this creates a shortcut for users in dropdown menus throughout the Protocol Fanout tab, allowing them to easily differentiate between multiple streams.

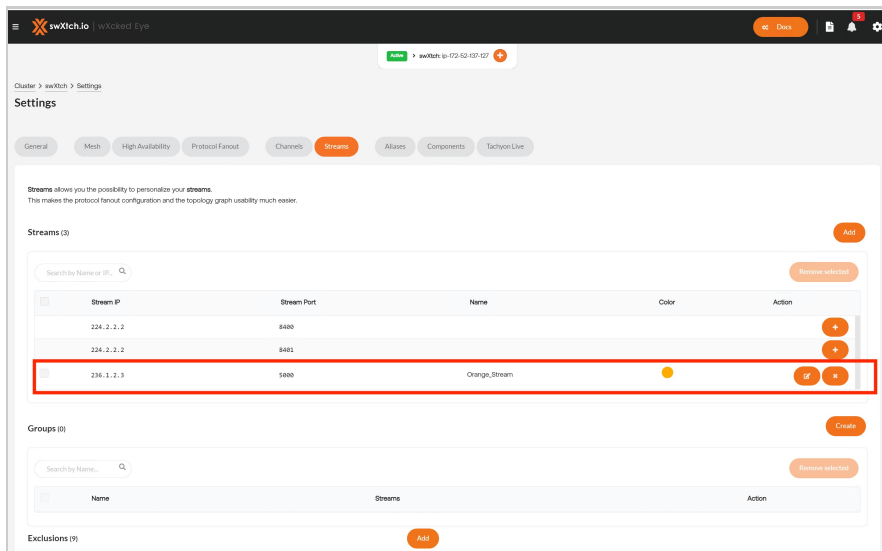
Adding a Stream



To add a stream:

1. Click the Add button at the top of the Streams section. A new window will open.
2. Enter the Stream IP and Stream Port.
3. Assign the stream a name and color.
4. Select Save.

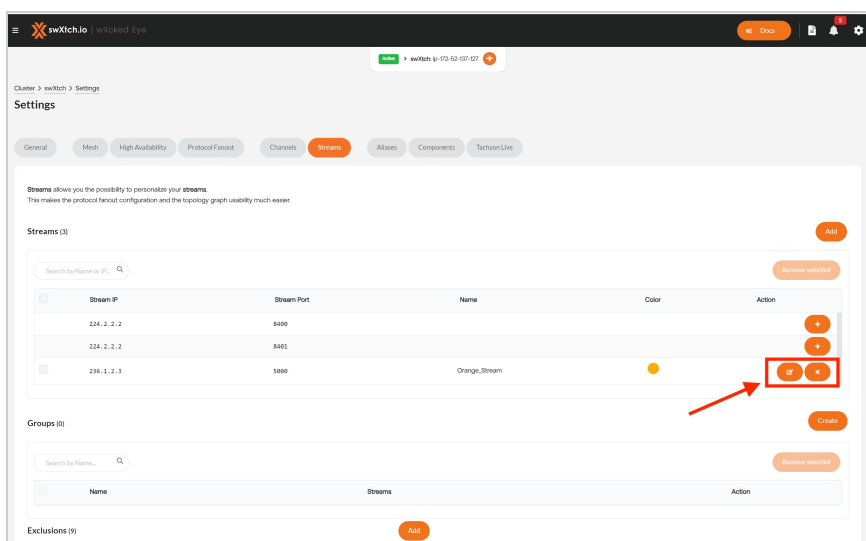
A new stream will now be listed.



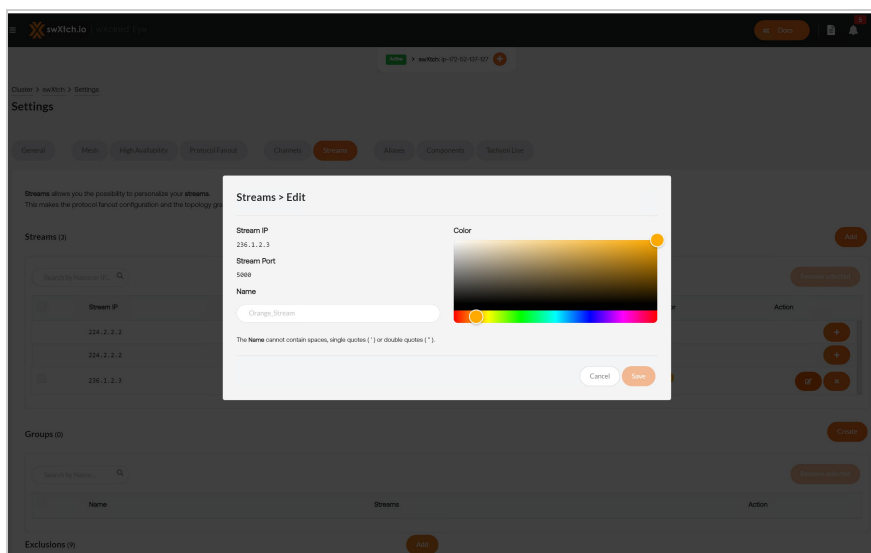
Please note: While the new stream is listed, it will not appear in the Topology Graph unless the cloudSwtch detects it. However, once the cloudSwtch connects with the stream, Wicked Eye will assign it the user-designated name and color and display it in the Topology Graph. For example, in the image above, streams 224.2.2.2:5400 and 224.2.2.2:5401 were detected by the cloudSwtch but are not yet assigned a name or color.

Editing and Deleting A Stream Name

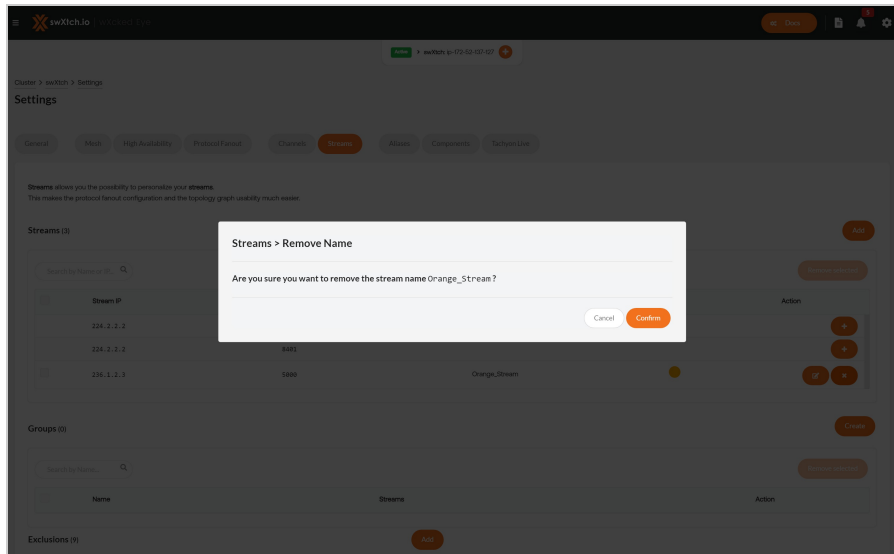
Under the Action column, users can edit or delete their stream's name from their Streams list.



Editing the stream will open a new window where the stream's name and color can be altered. You cannot edit Stream IP or Stream Port.



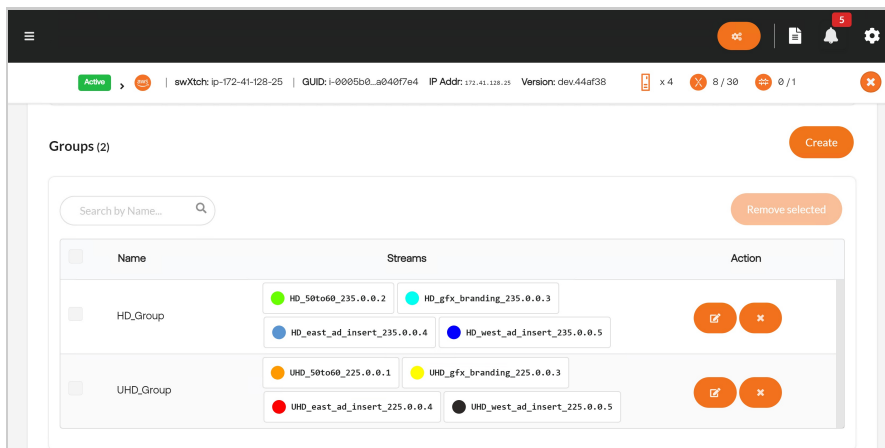
he **Delete** button will open a prompt, confirming the stream's name deletion.



Please note: This will only delete the Stream's name and color designation from the wXcked Eye UI. If the cloudSwXtch still detects the stream, it will still be listed under Streams.

Groups

Further down the page under Groups, users can assign a number of streams a group designation so that they appear highlighted together in the Topology Graph. In the example below, the user created two groups, UHD and HD.



This would help them distinguish between different "types" of streams in the Topology Graph, highlighting ones only belonging to their assigned group.

Creating a Group

Streams > Groups > Create

Create your **Streams Group** with the name and streams you want.
You can select a stream from the Available Streams table or create a new one.
NOTE: Available Streams includes streams involved in the swXtch network and also names defined by you.

Test_Group

Added Streams (3)

224.2.2.2:8400 ✕ 224.2.2.2:8401 ✕ 236.1.2.3:5000 ✕

Available Streams (3) ☐ Show excluded streams

	Stream	Name
✓	224.2.2.2:8400	
✓	224.2.2.2:8401	
✓	236.1.2.3:5000	Orange_Stream

Cancel

To create a Group:

1. Click the **Create** button at the top of the **Groups** section on the page. A new window will open.
2. Assign the streams group a name.
3. Select the streams you will like to add to the Group.
 - a. You can also add a New Stream to the Available Streams list directly in the window. You will follow the same prompts as the Streams tab.
4. Click Create.

A new group will appear in your Groups list.

Editing and Deleting a Group

Under the Action column, users can edit or delete their groups from their Groups list. Editing a group will open a new window where they can add and remove streams or rename the Streams Group.

If a stream is now in the Available Streams list, the user can do the following to add a New Stream:

1. Select New Stream in the Create or Edit Group window. A new window will open, requesting information.

swXtch.io

Streams > Groups > Edit

Test_Group

Added Streams (3)

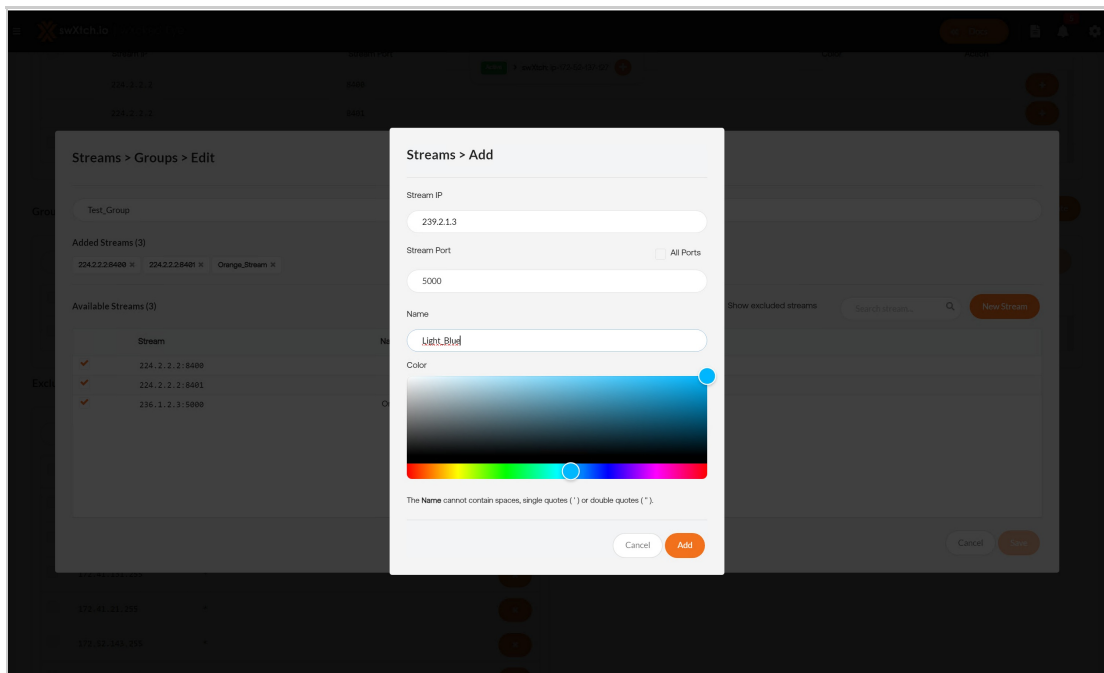
224.2.2.2:8400 ✕ 224.2.2.2:8401 ✕ Orange_Stream ✕

Available Streams (3) ☐ Show excluded streams

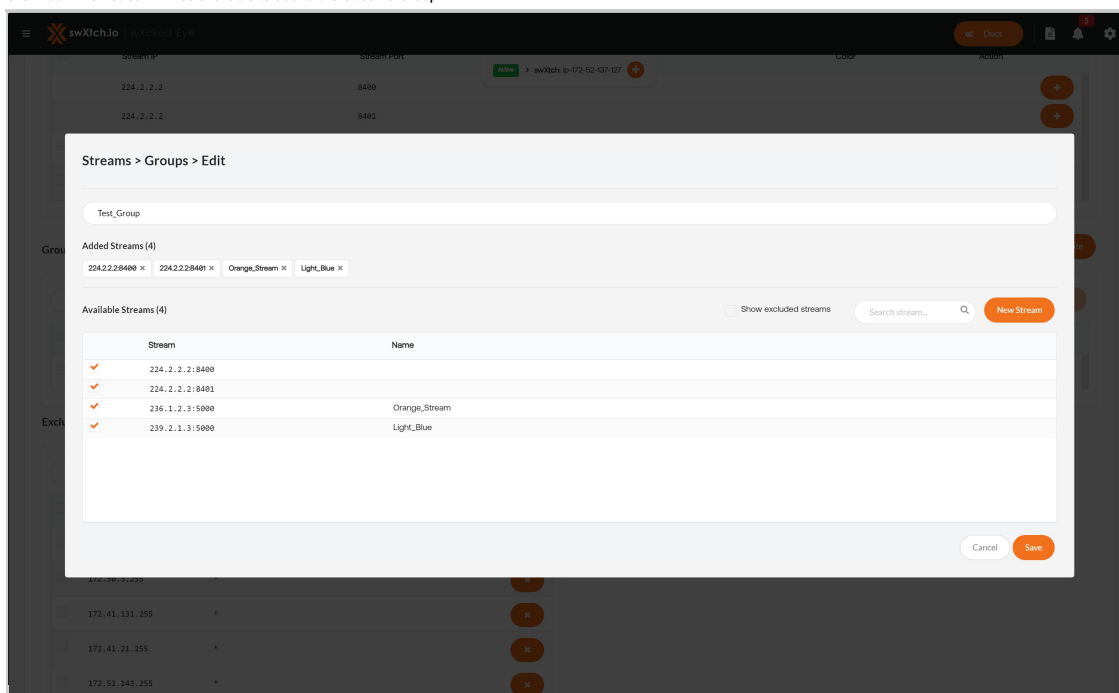
	Stream	Name
✓	224.2.2.2:8400	
✓	224.2.2.2:8401	
✓	236.1.2.3:5000	Orange_Stream

Cancel

2. Add the Stream IP, Stream Port, Name and Color of the stream.



3. Click Add. A new stream will be available to add to the Streams Group.



4. Click Save to confirm edits. The New Stream will now appear in the Streams list.

swXtch.io | wXcked Eye

Docs

Stream IP	Stream Port	Color	Action
224.2.2.2	8480		<div>+</div>
224.2.2.2	8481		<div>+</div>
<input type="checkbox"/> 236.1.2.3	5000	Orange_Stream	<div>+</div> <div>✕</div>
<input type="checkbox"/> 239.2.1.3	5000	Light_Blue	<div>+</div> <div>✕</div>

Groups (1)

Create

Search by Name...

Remove selected

Name	Streams	Action
<input type="checkbox"/> Test_Group	224.2.2.2:8480 224.2.2.2:8481 Orange_Stream <div>Light_Blue</div>	<div>+</div> <div>✕</div>

Exclusions (9)

Add

Search by IP...

Remove selected

Stream IP	Stream Ports	Excepted Ports	Action
<input type="checkbox"/> 172.30.15.255	*		<div>✕</div>
<input type="checkbox"/> 172.30.3.255	*		<div>✕</div>
<input type="checkbox"/> 172.41.131.255	*		<div>✕</div>
<input type="checkbox"/> 172.41.21.255	*		<div>✕</div>
<input type="checkbox"/> 172.52.143.255	*		<div>✕</div>

Exclusions

5

Active > swXtch: ip-172-41-128-25 | GUID: i-0005b0...a040f7e4 IP Addr: 172.41.128.25 Version: dev.44af38 x 4 8 / 30 0 / 1

Exclusions (8)

Add

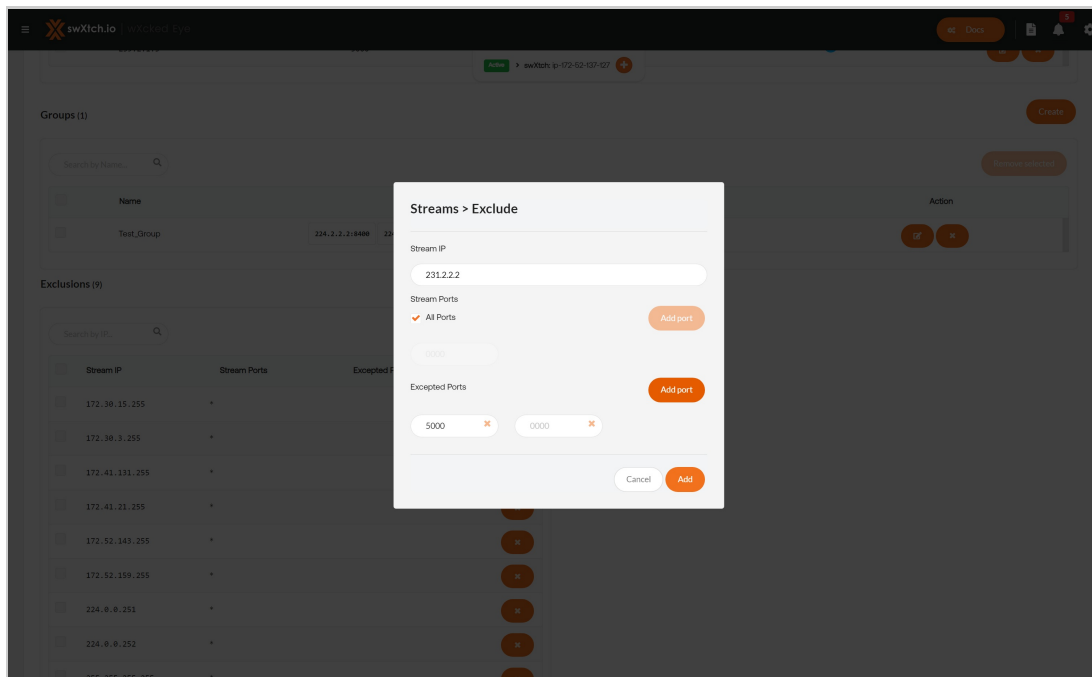
Search by IP...

Remove selected

Stream IP	Stream Ports	Excepted Ports	Action
<input type="checkbox"/> 172.30.3.255	*		<div>✕</div>
<input type="checkbox"/> 172.41.131.255	*		<div>✕</div>
<input type="checkbox"/> 172.41.21.255	*		<div>✕</div>
<input type="checkbox"/> 172.52.143.255	*		<div>✕</div>

The Streams tab includes the functionality to exclude streams from all monitoring related activities in wXcked Eye and swXtch-top. This is especially beneficial for users who use external software to generate additional streams that don't want them included when monitoring packet and data flow through the cloudSwXtch.

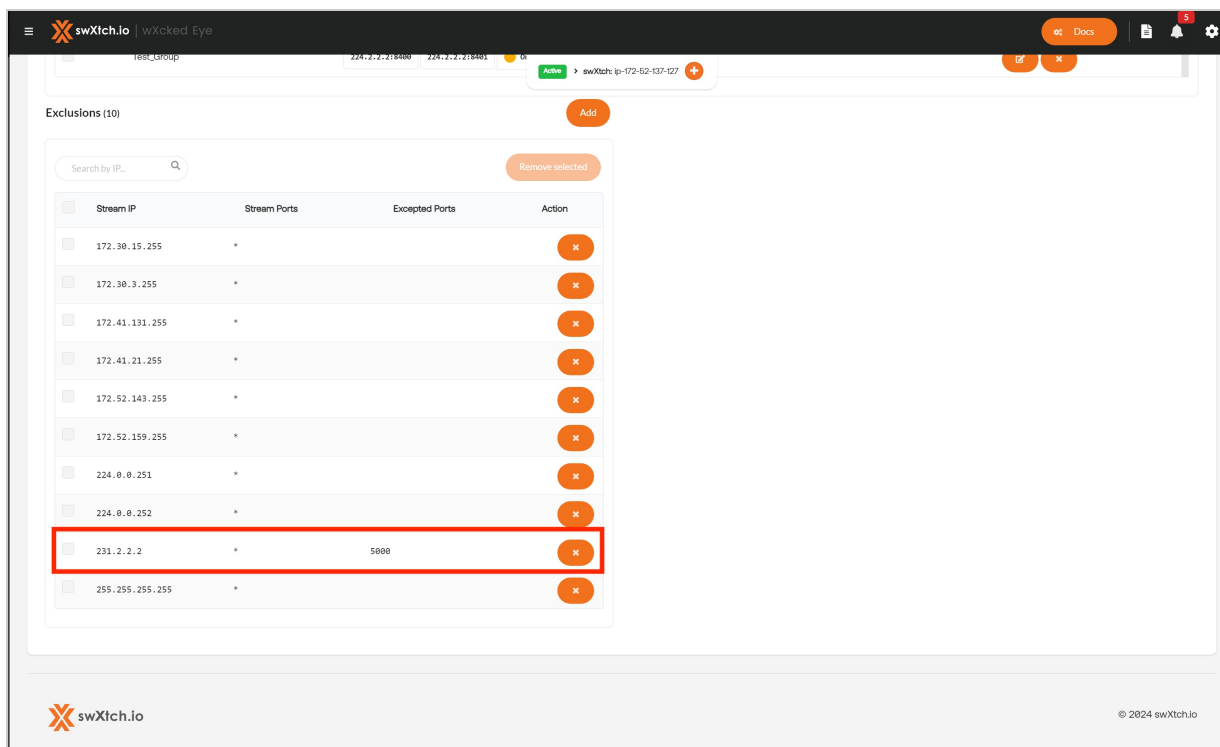
Adding an Exclusion



To add an exclusion:

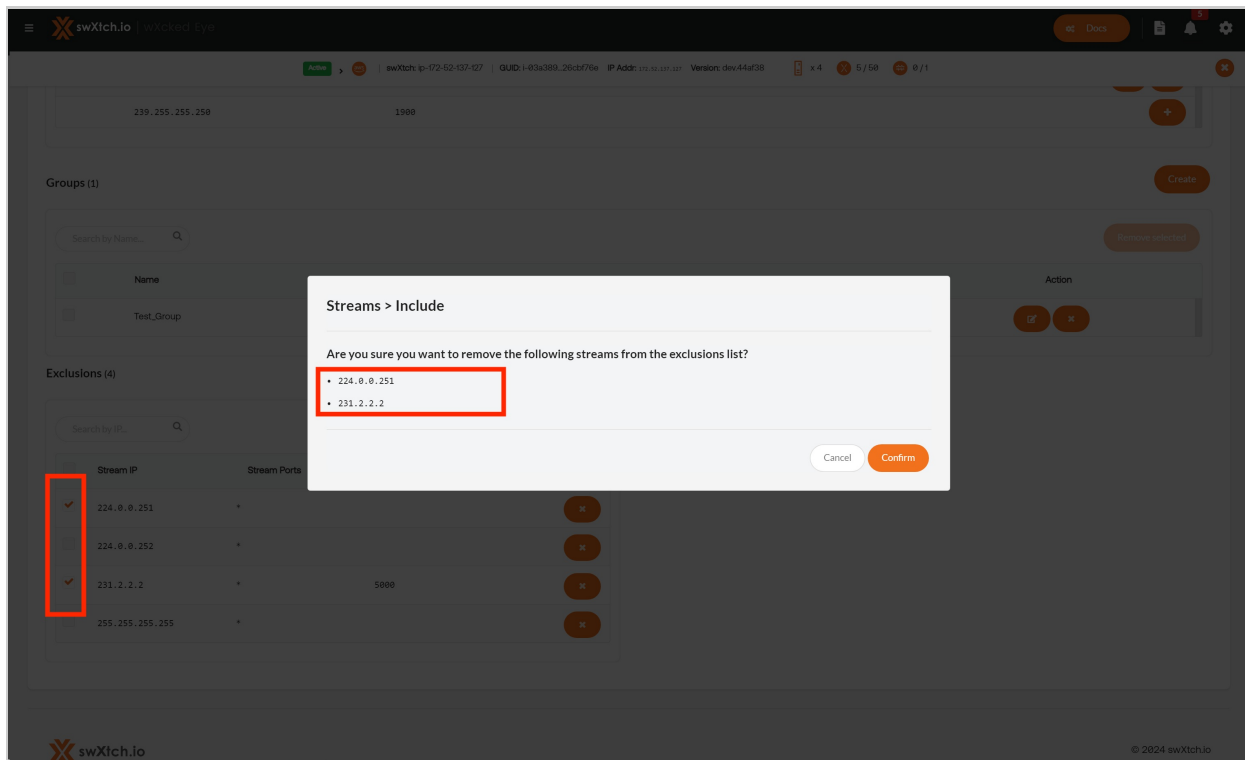
1. Click the Add button at the top of the Exclusions section of the Streams page. A new window will open.
2. Enter the Stream IP. You can also specify all or some ports.
 - a. You can also specify Excepted Ports where streams coming in from those ports are allowed.
3. Click Add.

A new stream should appear in the Exclusions list.



Deleting an Exclusion

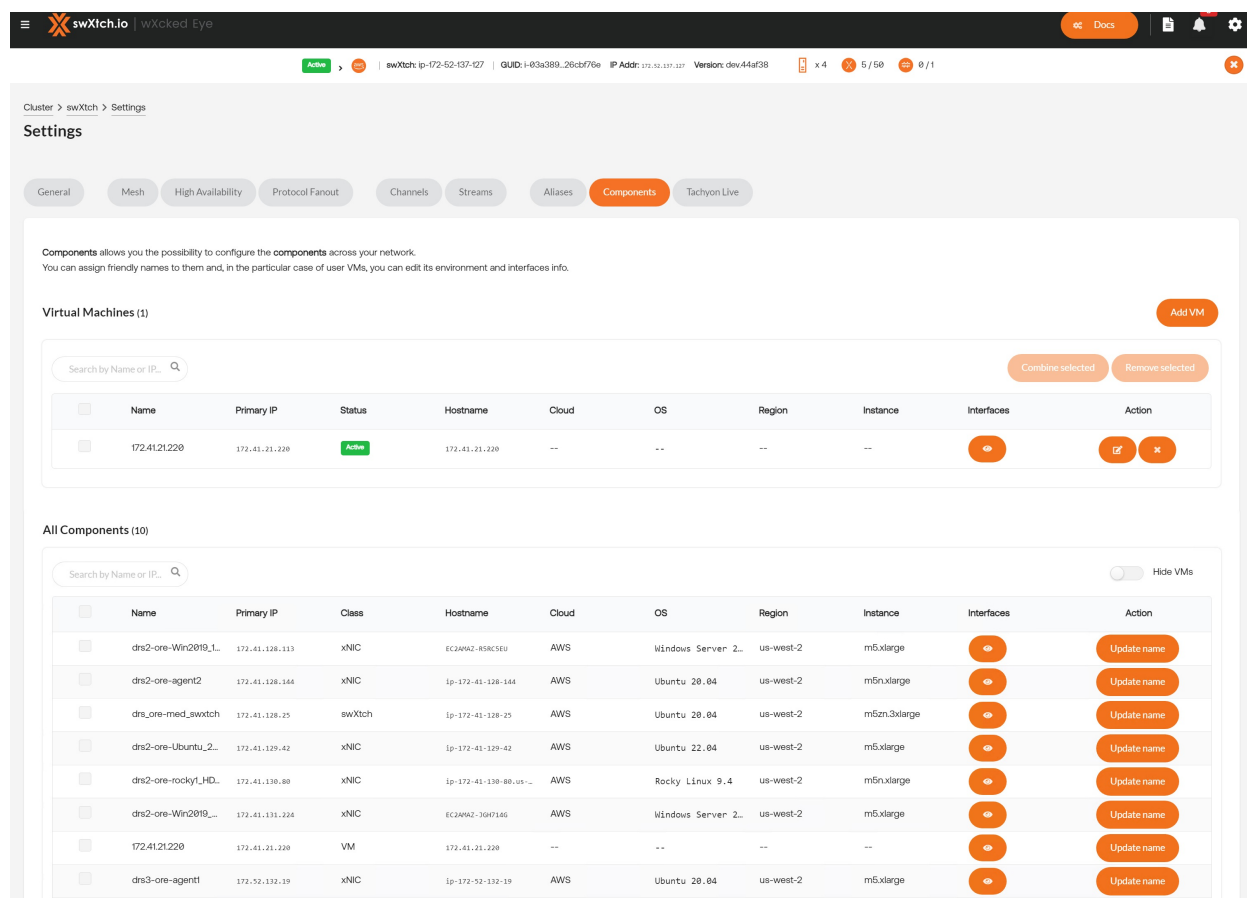
To delete an exclusion, click the X action button to the right of the listed stream OR select multiple streams. A prompt will appear, confirming your request.



Components

WHAT TO EXPECT

In this article, users will learn how the Components tab under Settings allows them to edit the description of nodes in the Topology graph. In addition, users will learn how to merge two VM components, or interfaces, as a single VM.



Introduction

The Components tab under the Settings page in wXcked Eye lists all of the virtual machines displayed in the Topology graph. It is separated into two sections: Virtual Machines and Components.

Virtual Machines

The Virtual Machines section is designated for VMs without an xNIC installed. Here, users can manually populate information for these non-xNIC VMs, such as the name, Primary IP, hostname, cloud provider, the operating system, the region, and the instance type.

Components

The Components section is a inclusive list of all VMs associated with the cloudSwXtch (cloudSwXtches, xNICs and cloudSwXtch Bridges). In addition, active VMs without an xNIC are also listed.

Here, users can manually populate information for virtual machines without an xNIC installed and update the name of components in a cloudSwXtch network (cloudSwXtches, xNICs and cloudSwXtch Bridges.)

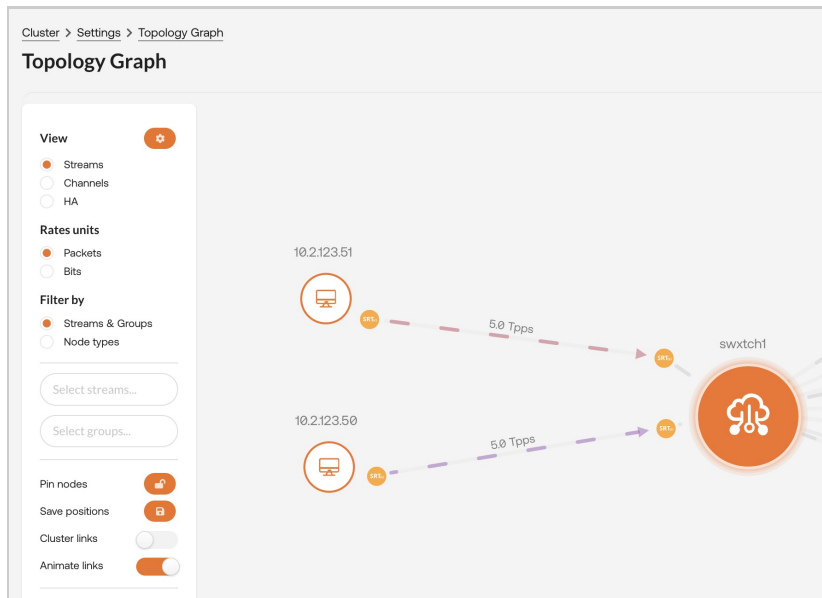
Updating Component Information

To update a component's information, select Update Name underneath the **Action** column. This will open a new page where users can enter information pertaining to the VM's environment. These fields do not automatically populate when adding a VM to a cloudSwXtch network and will need to be filled manually.

Please note: You can also edit the component's information directly in the Topology Graph.

Merging Interfaces into a Single VM

Since wXcked Eye cannot distinguish between interfaces from a single VM, it ends up duplicating the virtual machine in the topology graph. The Component page allows user to select these duplicated VMs and merge them into a single node. This will display traffic as going to and from the VM.



In this example, there are two different virtual machines displayed as separate nodes in the Topology Graph (10.2.123.50 and 10.2.123.51). However, they are actually the same virtual machine, just different interfaces with separate IPs. This is something that only a user will know.

To merge the interfaces into a single VM, users can follow these next steps:

1. Go to the **Components** tab under the **wXcked Eye Settings** page.
2. Search for the IP addresses of the interfaces that you would like to merge.

The screenshot shows the 'Components' tab in the 'wXcked Eye Settings' page. The search bar at the top contains the text '10.2.123.5'. Below the search bar, there is a table titled 'Virtual Machines (10)'. The table has columns: Name, Primary IP, Status, Hostname, Cloud, OS, Region, Instance, Interfaces, and Action. Two rows are visible, corresponding to the IP addresses 10.2.123.50 and 10.2.123.51. Both rows have a status of 'Active' and a 'Combine selected' button next to them.

Name	Primary IP	Status	Hostname	Cloud	OS	Region	Instance	Interfaces	Action
10.2.123.50	10.2.123.50	Active	10.2.123.50	--	--	--	--	Combine selected	Remove selected
10.2.123.51	10.2.123.51	Active	10.2.123.51	--	--	--	--	Combine selected	Remove selected

3. Select the two interfaces and click **Combine selected**.
 - a. This will open a new page where you can set the information for the combined VM.

The screenshot shows the 'Components' tab in the 'wXcked Eye Settings' page. The search bar at the top contains the text '10.2.123.5'. Below the search bar, there is a table titled 'Virtual Machines (10)'. The table has columns: Name, Primary IP, Status, Hostname, Cloud, OS, Region, Instance, Interfaces, and Action. Two rows are visible, corresponding to the IP addresses 10.2.123.50 and 10.2.123.51. Both rows have a status of 'Active' and a 'Combine selected (2)' button next to them. The 'Combine selected (2)' button is highlighted with a red box and an arrow.

Name	Primary IP	Status	Hostname	Cloud	OS	Region	Instance	Interfaces	Action
10.2.123.50	10.2.123.50	Active	10.2.123.50	--	--	--	--	Combine selected (2)	Remove selected (2)
10.2.123.51	10.2.123.51	Active	10.2.123.51	--	--	--	--	Combine selected (2)	Remove selected (2)

4. Under **General**, set the combined VM's **Primary IP address** and assign the VM a name. This will be used in the Topology graph. It is recommended to select a name that explicitly indicates that this VM has two separate interface.
5. Add information regarding the VM's **environment**.
6. Under **Interfaces**, assign a **name** to each of the two selected interfaces.
7. Click **Save**.
 - a. The two VMs should now be merged into one with the primary interface IP set.

Components > Combine VMs

General

Primary IP10.2.123.50

NameCombined_VM

Environment

Hostname10.2.123.50

CloudAWS

OSLinux

RegionUS East

InstanceInstance

Interfaces

Nameeth0

Primary IP10.2.123.50

Nameeth1

IP10.2.123.51

Cancel

Save

8. Confirm the merge by selecting the eye icon under Interfaces.

Cluster > Settings

Settings

General

Mesh

High Availability

Protocol Fanout

Channels

Streams

Components

Apps

Components allows you the possibility to configure the components across your network.

You can assign friendly names to them and, in the particular case of user VMs, you can edit its environment and interfaces info.

Virtual Machines (9)

10.2.123.5

Combine selected

Remove selected

	Name	Primary IP	Status	Hostname	Cloud	OS	Region	Instance	Interfaces	Action
<input type="checkbox"/>	Combined_VM	10.2.123.50	Active	10.2.123.50	AWS	Linux	US East	--		

a. This will open a new window, listing the VM's interfaces.

Cluster > Settings

Settings

General

Mesh

High Availability

Protocol Fanout

Channels

Components allows you the possibility to configure the components across your network.

You can assign friendly names to them and, in the particular case of user VMs, you can edit its

Virtual Machines (9)

10.2.123.5

Combine selected

Remove selected

	Name	Primary IP	Status	Hostname	Cloud	OS	Region	Instance	Interfaces	Action
<input type="checkbox"/>	Combined_VM	10.2.123.50	Active	10.2.123.50	AWS	Linux	US East	--		

Combined_VM

Primary IP10.2.123.50

Interfaces

Nameeth0

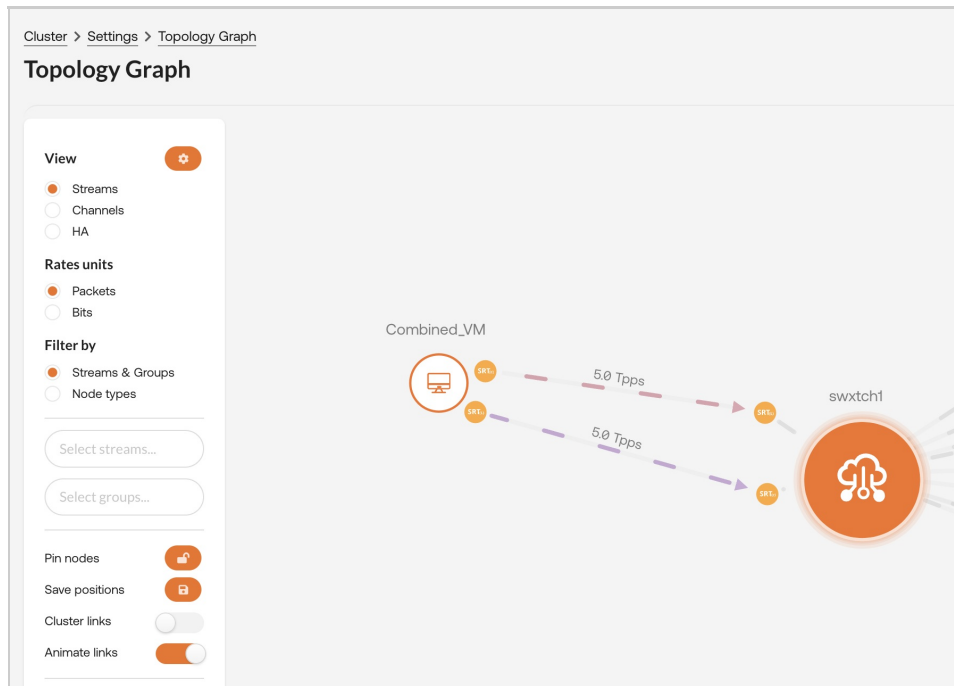
IP10.2.123.50

Nameeth1

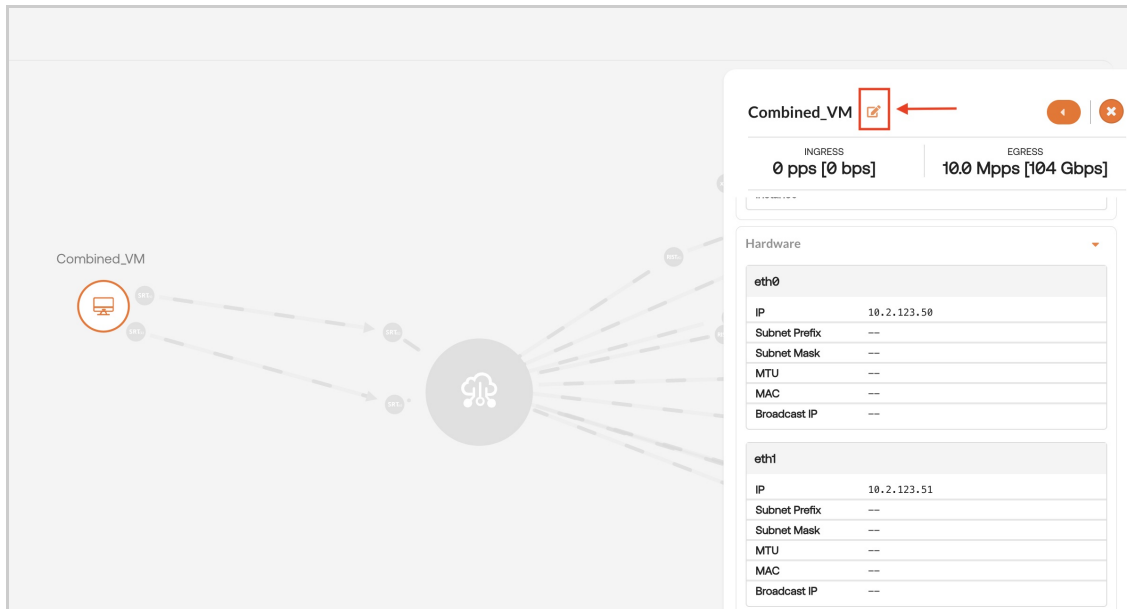
IP10.2.123.51

Close

9. Verify that the new combined node is visible in the Topology Graph.



a. You can also see the component's information by highlighting the node. Editing is also available and will update the node's information across the wXcked Eye UI.



Configuring cloudSwXtch for HA

WHAT TO EXPECT

In this article, users will learn about the specific commands they can use to configure the cloudSwXtch for high availability. Alternatively, users can also configure the cloudSwXtch for high availability via wXcked Eye. **This is the preferred method.**

For more information, see [High Availability with wXcked Eye](#).

Configuring cloudSwXtch for High Availability

For High Availability to work, the cloudSwXtch must be configured. This allows for the system to know the paths through user naming and ultimately enables HA views in swXtch-top. In this section, we will review the various HA commands available to the user for a successful high availability configuration.

Users can also configure High Availability for the cloudSwXtch in wXcked Eye. That is the preferred method. For more information, see [High Availability with wXcked Eye](#).

HA Help

To get a list of the HA commands, use -h or --help as shown below.

Bash	Copy
<pre>PS C:\Users\testadmin> swx ha -h High Availability cluster management tool (create, show, and destroy the HA cluster) Usage: swx ha [command] Available Commands: create Create or Update the HA cluster of swxtches using a config file destroy Destroy the HA cluster show Show information about the HA cluster Flags: -h, --help help for ha -s, --service-host-address string Host swxtch address in the form <host>[:port] -d, --show-error show-error - display additional information for error messages. Use "swx ha [command] --help" for more information about a command.</pre>	

NOTE

The default port in which the cloudSwXtch listens for these swx configuration commands is port 80. You can safely omit the port in the "-s" parameter since 80 will be used. Do not use port 10802 (the one used in the config file), as it is intended for xNIC communications only. It will not work for swx commands.

HA Create

To create or update an HA cluster, a HAconfig.json file must exist. Note that the IP is for the control plane. The following shows the format:

Bash	Copy
<pre>{ "name": "MY High Availability", "paths": [{ "name": "Path 1", "swxtches": ["10.2.128.10"] }, { "name": "Path 2", "swxtches": ["10.5.1.6"] }] }</pre>	

If there are multiple cloudSwXtches in a path then the last cloudSwXtch added will have the IP address in the configuration. The rest of the cloudSwXtches do not need to be listed.

The -h and --help commands will show the syntax for the "swx ha create" command.

Bash	Copy
<pre> PS C:\Users\testadmin> swx ha create -h Create or Update the HA cluster of swxtches using a config file Usage: swx ha create [flags] Flags: -h, --help help for create -i, --input string JSON input file Global Flags: -s, --service-host-address string Host swxtch address in the form <host>[:port] -d, --show-error show-error - display additional information for error messages. </pre>	

Below is the command:

Bash	Copy
<pre> swx ha create -i <path_to_config> -s <cloudSwXtch_IP> </pre>	

HA Destroy

To remove a cloudSwXtch from the High Availability flow, the "ha destroy" command can be used. The -h and --help commands will show the syntax for the "swx ha destroy" command.

Bash	Copy
<pre> PS C:\Users\testadmin> swx ha destroy -h Destroy the HA cluster Usage: swx ha destroy [flags] Flags: -h, --help help for destroy Global Flags: -s, --service-host-address string Host swxtch address in the form <host>[:port] -d, --show-error show-error - display additional information for error messages. </pre>	

Below is the command to leave:

Bash	Copy
<pre> PS C:\Users\testadmin> swx ha destroy -s <swxtch name or control data ip of a cloudSwXtch in the HA configuration> </pre>	

Example:

Bash	Copy
<pre> PS C:\Users\testadmin> swx ha destroy -s cloudswxtch101 Validating cluster deletion. Successfully deleted the cluster. </pre>	

Removing a cloudSwXtch from an HA configuration

To remove a cloudSwXtch from a user's HA configuration, they will need to delete and recreate their HA cluster without that cloudSwXtch.

HA Show

To get a list of cloudSwXtches part of the HA flow, the "ha show" command can be used. The -h and --help commands will show the syntax for the "swx ha show" command.

Bash	Copy
<pre> PS C:\Users\testadmin> swx ha show -h Show information about the HA cluster Usage: swx ha show [flags] Flags: -h, --help help for show Global Flags: -s, --service-host-address string Host swxtch address in the form <host>[:port] -d, --show-error show-error - display additional information for error messages. </pre>	

Below is the command to list:

Bash

Copy

```
swx ha show -s <swxtch name or control data IP of a cloudSwXtch in the HA configuration>
```

Example below:

Bash

Copy

```
PS C:\Users\testadmin> swx ha show -s cloudswxtch101
{
  "clusterConfig": {
    "uid": "D20E820C-91DE-A571-4C7C-B60C1695973D",
    "name": "dsd-100-200-HA",
    "paths": [
      {
        "name": "Path1",
        "swxtches": [
          "10.2.128.10"
        ]
      },
      {
        "name": "Path2",
        "swxtches": [
          "10.5.1.6"
        ]
      }
    ]
  }
}
```

swxtch-top has options for HighAvailability. For more information, see the [swxtch-top](#) article.

Configuring xNIC on Endpoints for HA

WHAT TO EXPECT

In this article, users will learn the various ways xNICs at their endpoints can be configured for high availability.

- [Single Multicast Group](#)
- [Multiple Multicast Group](#)
- [Source Specific Multiple Multicast](#)
- [Multi-VPC/Data NIC Support](#)

Note: Reinstalling the xNIC will re-write configuration files to their original state, undoing any of the above configurations.

Configuring xNIC for High Availability

After a user sets up their cloudSwXtch for High Availability, the xNIC will automatically configure itself to receive and/or send HA traffic for a single multicast group. Users can confirm high availability has been configured by viewing the JSON file in the VM where their xNIC resides.

Where to Find the Bridge JSON File

To configure the cloudSwXtch Bridge for high availability streams, the swxtch-bridge.json file needs to be updated. See where to locate them below:

For Linux:

The file can be found in `/var/opt/swxtch/xnic.json`. Currently, only V1 is supported for Linux.

To edit the file, one option is to use nano as shown below:

BashCopy

```
sudo nano /var/opt/swxtch/xnic.json
```

For Windows:

The file can be found in `C:\Program Files\SwXtch.io\Swxtch-xNIC\xnic.json`

Single Multicast Group

An example of the `xnic.json` file is shown below. Note the `ha` section has been added.

BashCopy

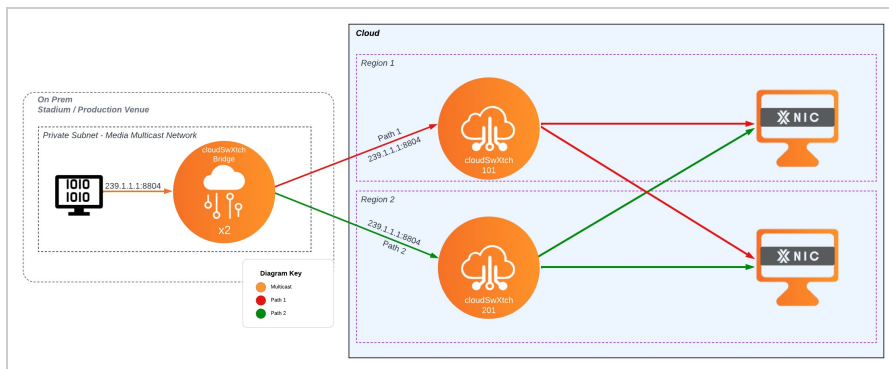
```
{
  "swxtch": "10.2.128.10",
  "controlInterface": "Ethernet 2",
  "dataInterface": "Ethernet",
  "dataPort": 9999,
  "xnicType": 2,
  "dataPlaneSpecs": {
    "verbosity": 0,
    "virtualInterface": {
      "name": "swxtch-tun0",
      "ip": "172.30.0.0",
      "subnet": "255.255.0.0",
      "mtu": 4096
    },
    "bpfPrograms": [
    ]
  },
  "ha": {
    "enable": true,
    "protocol": "rtp",
    "deduplication": {
      "maxTimeToBufferPacketsMs": 50,
      "bufferSizeInPackets": 131072
    }
  },
  "overrideSrcIp": false,
  "statsReportWait": 60,
  "subscriptionsPollingIntervalMs": 100
}
```


ha Section Explained

The ha section exposes variables that can alter the behavior of the hitless switching code. The values for `MaxTimeToBufferPackets_ms` and `BufferSizeInPackets` in the example are good, suggested values; however, they can be tweaked to meet desired high availability requirements.

- **enable**: If set to true, the xNIC will join the HA cluster when configured. The data and control traffic will start flowing to/from cloudSwXtches in the HA cluster. If set to false, the xNIC will ignore the HA cluster and continue operating only with its primary cloudSwXtch. The default is true as it is the most common use case.
- **Protocol**- how to parse the packet. The available options are `swtch` or `rtp`.
 - **swtch** = This can be used when the xNIC is duplicating or deduplicating the multicast. The xNIC will reconstruct based on the sequence count inside the cloudSwXtch packet header.
 - **rtp** = This should be used when processing RTP packets sent from a non-xNIC source. The xNIC will reconstruct based on the RTP timestamp information for Real-time Transport Protocol.
- **deduplication**: This is only valid if "enable" is set to true. When this field is populated with the following values, the data plane will turn on HA and de-duplicate traffic from the HA cluster. If set to null or non-existent, it will receive all traffic from the HA cluster, allowing users to deduplicate using their own application.
 - **MaxTimeToBufferPackets_ms** - how long to buffer packets before declaring it as lost.
 - **BufferSizeInPackets**- the max number of packets that can be buffered.
- **subscriptionsPollingIntervalMs**: Time in milliseconds to buffer and wait for reordered/delayed packets. *Default: ~100ms*

Below is an example of what a single multicast group configuration might look like. In this example, a user is sending the same multicast traffic (239.1.1.1:8804) via two paths with the xNIC consuming both and deduplicating at the end point.

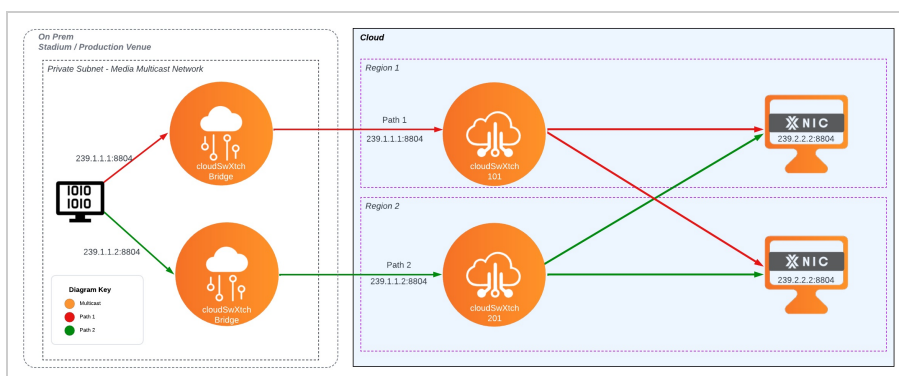


Multiple Multicast for xNIC

Important Rules

- Protocol for Multiple Multicast (MMC) must be set to `rtp`.
- For MMC on a producer, the xNIC must be Type 1.
- If the producer is set to true, then it must be set to Type 1 even if you're just consuming.
- Multiple NICs are currently not supported for Ingress Windows Type 2.

If a user wants to set up for Multiple Multicast groups, they will need to manually configure the `xnic.json` file. Below is an example of what a multiple multicast group configuration might look like:



In this example, you have two paths with the same multicast traffic with different IP addresses. Path 1 is 239.1.1.1 while Path 2 is 239.1.1.2. The application at the end point is listening to 239.2.2.2, which is grouping together Path 1 and Path 2. The xNIC at the end point is tasked with deduplication.

A sample `xnic.json` file of the diagram is shown below with a "streamSpecs" section added.

Note that the following rules apply to this json file:

- The `xnicType` is set to 2 but could also be 1. It has to be set to 1 if the VM is for a producer.
- In the HA section, the protocol is set to `rtp`.

```
{
  "swxtch": "10.2.128.10",
  "controlInterface": "Ethernet 2",
  "dataInterface": "Ethernet",
  "dataPort": 9999,
  "xnicType": 2,
  "dataPlaneSpecs": {
    "verbosity": 0,
    "virtualInterface": {
      "name": "swxtch-tun0",
      "ip": "172.30.0.0",
      "subnet": "255.255.0.0",
      "mtu": 4096
    },
    "bpfPrograms": [
    ]
  },
  "ha": {
    "enable": true,
    "protocol": "rtp",
    "deduplication": {
      "maxTimeToBufferPacketsMs": 50,
      "bufferSizeInPackets": 131072
    }
  },
  "streamSpecs": {
    "mmcProducerEnable": false,
    "multipleMulticastGroups": {
      "239.2.2.2:8804": {
        "pathStreams": [
          {
            "stream": "239.1.1.1:8804"
          },
          {
            "stream": "239.1.1.2:8804"
          }
        ]
      }
    }
  },
  "overrideSrcIp": false,
  "statsReportWait": 60,
  "subscriptionsPollingIntervalMs": 100
}
```

Here, the user is grouping together 2 multicast IPs (239.1.1.1 and 239.1.1.2) and assigning it a multicast group IP address (239.2.2.2). The application at the endpoint is listening for 239.2.2.2, which the xNIC will deduplicate into a stream from 239.1.1.1 and 239.1.1.2. This was illustrated in the diagram above.

Please note: At this time, the ports for the multicast group and the path streams must be the same.

How to update xnic.json file for Multiple Multicast Groups

The user will have to make the following changes to their `xnic.json` file found in the [single multicast group configuration](#) to match the example above. These alterations to the `xnic.json` file should happen after [Configuring the cloudSwXtch for High Availability](#).

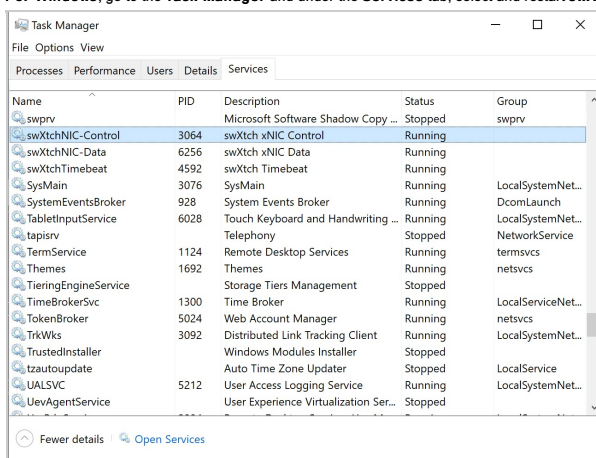
1. The default xNIC Type is set to 2. Change the xNIC type to 1 if following conditions are true:
 - a. if the VM is a producer
 - b. OR if the VM is a Windows machine with multiple data NICs. For more information about multiple data NICs, see [here](#).
2. Change the protocol under ha from "swxtch" to "rtp" including the quotation marks.
3. **For each multicast group**, add the following "streamSpecs" section as shown below with your stream data. If the VM is a producer, set mmcProducerEnable to true. **Note:** A user can enter multiple multicast groups.

```
Bash Copy
},
  "streamSpecs": {
    "mmcProducerEnable": false,
    "multipleMulticastGroups": {
      "239.2.2.2:8804": {
        "pathStreams": [
          {
            "stream": "239.1.1.1:8804"
          },
          {
            "stream": "239.1.1.2:8804"
          }
        ]
      }
    }
  }
},
```

3. Save the **xnic.json** file.

4. Restart the **swXtch-NIC Control** service.

- For Windows, go to the **Task Manager** and under the **Services** tab, select and restart **swXtchNIC-Control**.



- For Linux, use the following command:

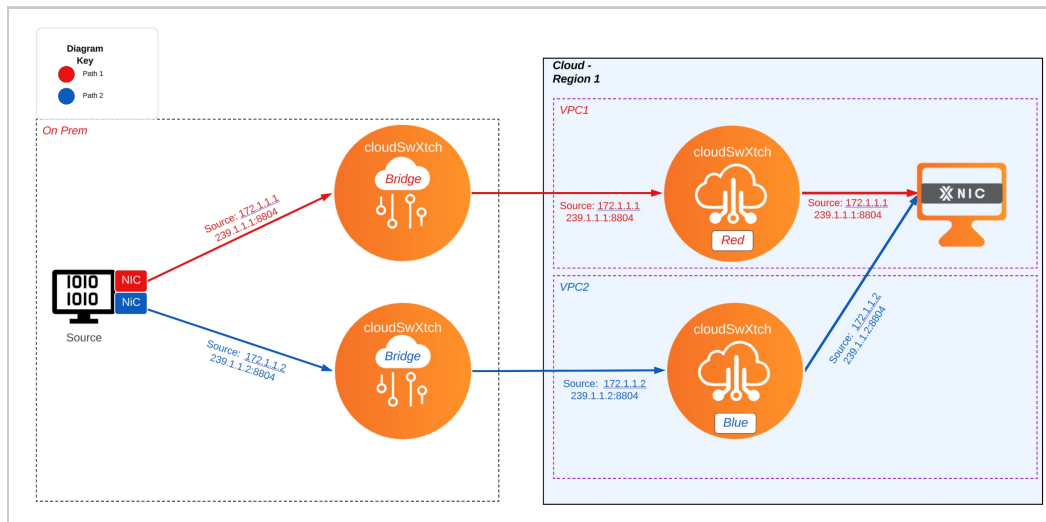
```
Bash Copy
sudo systemctl restart swxtch-xnic-control
```

Source Specific Multiple Multicast

Important Rules

- Protocol for Source Specific Multiple Multicast must be set to **rtp**.
- For Source Specific Multiple Multicast on a producer, the **xNIC** must be Type 1.
- If the producer is set to true, then it must be set to Type 1 even if you're just consuming.

If a user wants to set up Source Specific Multiple Multicast groups, they will need to manually configure the **xnic.json** file. Below is an example of what a source specific multiple multicast group configuration might look like:



In this example, you have two paths with the same multicast traffic with different multicast IP addresses along with their sources. Path 1 is 239.1.1.1 with a source of 172.1.1.1 while Path 2 is 239.1.1.2 with a source of 172.1.1.2. The application at the end point is listening to 239.2.2.2, which is grouping together Path 1 and Path 2. The xNIC at the end point is tasked with deduplication.

Producer

A sample producer `xnic.json` file of the diagram is shown below with a "streamSpecs" section added. **Note that the following rules apply to this producer json file:**

- The **xnicType** is set to 1.
- In the HA section, the **protocol** is set to **rtp**.
- In the streamSpecs section, there is no source specified since it is a producer.

```
{
  "swxtch": "172.41.128.25",
  "controlInterface": "ens5",
  "dataInterface": "ens6",
  "dataPort": 9999,
  "xnicType": 1,
  "xnicRpcPort": 10002,
  "dataPlaneSpecs": {
    "verbosity": 0,
    "virtualInterface": {
      "type": "tun",
      "name": "swxtch-tun0",
      "ip": "172.30.0.0",
      "subnet": "255.255.0.0",
      "mtu": 4096
    },
    "bpfPrograms": [
      {
        "name": "tc-ingress",
        "interface": "ens6",
        "attachPoint": "BPF_TC_INGRESS"
      },
      {
        "name": "tc-egress",
        "interface": "ens6",
        "attachPoint": "BPF_TC_EGRESS"
      },
      {
        "name": "tc-forwarder",
        "interface": "ens5",
        "attachPoint": "BPF_TC_EGRESS"
      },
      {
        "name": "tc-forwarder",
        "interface": "ens7",
        "attachPoint": "BPF_TC_EGRESS"
      }
    ]
  },
  "ha": {
    "enable": true,
    "protocol": "rtp",
    "deduplication": {
      "maxTimeToBufferPacketsMs": 50,
      "bufferSizeInPackets": 131072
    }
  },
  "streamSpecs": {
    "mmcProducerEnable": true,
    "multipleMulticastGroups": {
      "224.2.2.2:8400": {
        "pathStreams": [
          {
            "stream": "224.2.2.3:8400"
          },
          {
            "stream": "224.2.2.4:8400"
          }
        ]
      }
    }
  },
  "overrideSrcIp": false,
  "statsReportWait": 60,
  "subscriptionsPollingIntervalMs": 100
}
```

Consumer

A sample [consumer xnic.json](#) file of the diagram is shown below with a "streamSpecs" section added.

Note that the following rules apply to this consumer json file:

- The xnicType is set to 2 but should be 1 if the VM is a Windows machine with multiple data NICs. For more information about multiple data NICs, see [here](#).
- In the HA section, the protocol is set to rtp.
- In the streamSpecs section, there is a source specified since it is a consumer.

```
{
  "swtch": "172.41.128.25",
  "controlInterface": "Ethernet 3",
  "dataInterface": "Ethernet 4",
  "dataPort": 9999,
  "xnicType": 2,
  "xnicRpcPort": 10002,
  "dataPlaneSpecs": {
    "verbosity": 0,
    "virtualInterface": {
      "type": "tun",
      "name": "swtch-tun0",
      "ip": "172.30.0.0",
      "subnet": "255.255.0.0",
      "mtu": 4096
    }
  },
  "ha": {
    "enable": true,
    "protocol": "rtp",
    "deduplication": {
      "maxTimeToBufferPacketsMs": 50,
      "bufferSizeInPackets": 131072
    }
  },
  "streamSpecs": {
    "mmcProducerEnable": false,
    "multipleMulticastGroups": {
      "239.2.2.2:8400": {
        "overrideSourceIp": "172.1.1.1",
        "pathStreams": [
          {
            "stream": "239.1.1.1:8400",
            "ssm": {
              "filter": "include",
              "srcList": [
                "172.1.1.1"
              ]
            }
          },
          {
            "stream": "239.1.1.2:8400",
            "ssm": {
              "filter": "include",
              "srcList": [
                "172.1.1.2"
              ]
            }
          }
        ]
      }
    }
  },
  "overrideSrcIp": false,
  "statsReportWait": 60,
  "subscriptionsPollingIntervalMs": 100
}
```

Here, you have two paths with the same multicast traffic with different multicast IP addresses along with their sources. Path 1 is 239.1.1.1 with a source of 172.1.1.1 while Path 2 is 239.1.1.2 with a source of 172.1.1.2. The application at the end point is listening to 239.2.2.2, which is grouping together Path 1 and Path 2. The xNIC at the end point is tasked with deduplication.

Please note: At this time, the ports for the multicast group and the path streams must be the same.

How to update xnic.json file for Source Specific Multiple Multicast Groups

The user will have to make the following changes to their `xnic.json` file found in the [single multicast group configuration](#) to match the example above. These alterations to the `xnic.json` file should happen after [Configuring the cloudSwXtch for High Availability](#).

1. Change the xNIC type to 1 if the VM is a producer OR if is a Windows machine with multiple data NICs. For more information about multiple data NICs, see [here](#).
2. Change the protocol under ha from "swtch" to "rtp" including the quotation marks.
3. **Insert the following after the HA section.** Then, for each additional multiple multicast group, add a new multiple multicast set. If the VM is a producer, set `mmcProducerEnable` to true.

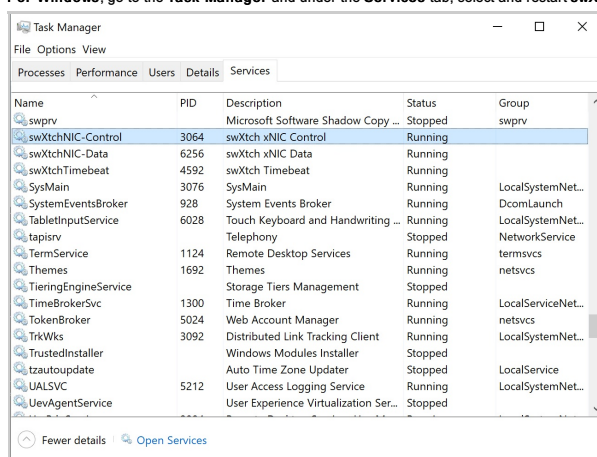
```
Bash Copy

"streamSpecs": {
  "mmcProducerEnable": false,
  "multipleMulticastGroups": {
    "239.2.2.2:8400": {
      "overrideSourceIp": "172.1.1.1",
      "pathStreams": [
        {
          "stream": "239.1.1.1:8400",
          "ssm": {
            "filter": "include",
            "srcList": [
              "172.1.1.1"
            ]
          }
        },
        {
          "stream": "239.1.1.2:8400",
          "ssm": {
            "filter": "include",
            "srcList": [
              "172.1.1.2"
            ]
          }
        }
      ]
    }
  }
},
}
```

3. Save the `xnic.json` file.

4. Restart the `swXtch-NIC Control` service.

- For Windows, go to the **Task Manager** and under the **Services** tab, select and restart `swXtchNIC-Control`.



- For Linux, use the following command:

```
Bash Copy

sudo systemctl restart swxtch-xnic-control
```

How to update `xnic.json` file to Disable Deduplication at the Consumer

Some users may want to disable deduplication by the xNIC in favor of their own application. To do this, navigate to the `xnic.json` file and add "null" after the deduplication flag, removing the bracketed `maxTimeToBufferPacketsMs` and `bufferSizeInPackets`, as seen in the example below:

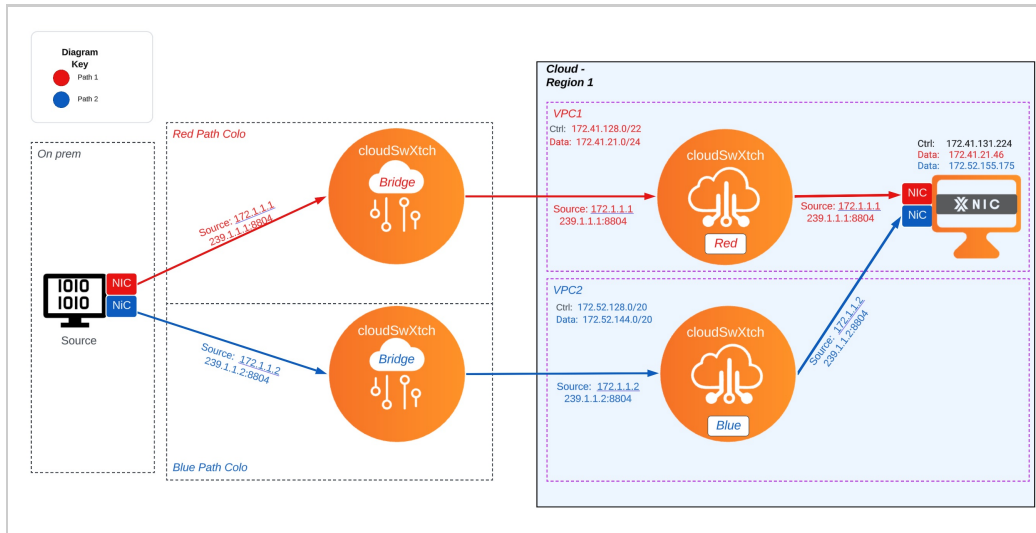
```
PowerShell Copy

"ha": {
  "enable": true,
  "protocol": "rtp",
  "deduplication": null
},
```

This will inform the xNIC at the consumer VM, receiving traffic from the HA cluster, to not deduplicate the streams. The user can now apply their own application to deduplicate the traffic if they so wish.

Multiple VPC/Data NIC Support

swXtch.io supports multiple VPC/Data NICs for network isolation across two or more paths. In the example below, you have two data NICs for red and blue paths.



The NICs can belong to either the same VPC or entirely different ones. For better data isolation, it is recommended to have each path flow through different VPCs so that if one complete VPC goes down, traffic will still flow via the other. Another benefit is that since the NIC is part of path/region, no peering is required between VPCs.

AWS Multi-VPC ENI Attachments Explained

For a detailed description of multi-VPC ENI attachments on AWS, see their documentation here: <https://aws.amazon.com/about-aws/whats-new/2023/10/multi-vpc-eni-attachments/>

Multiple VPC/Data NIC Rules

The following rules apply for Multiple VPC/Data NIC on xNIC:

1. For Multiple VPC/Data NICs on a producer, the xNIC must be Type 1.
2. If the xNIC is the producer, then single multicast group is supported.
3. Windows consumer must be set to Type 1.

For rules regarding a specific high availability configuration, see the corresponding section:

- [Multiple Multicast](#)
- [Source Specific Multiple Multicast](#)

Configuring cloudSwXtch Bridge for HA

WHAT TO EXPECT

In this article, users will learn how to configure Bridge 2 and 3 for High Availability.

Configuring cloudSwXtch Bridge Type 2 and 3 for High Availability

After a user sets up their cloudSwXtch for High Availability, the cloudSwXtch Bridge will automatically configure itself to receive and/or send HA traffic for a single multicast group. Users can confirm high availability has been configured by viewing the swxtch-bridge.json file in the VM where their cloudSwXtch Bridge resides.

Reminder: A cloudSwXtch Bridge can only exist on a Linux machine.

Where to Find the Bridge JSON File

To configure the cloudSwXtch Bridge for high availability streams, the swxtch-bridge.json file needs to be updated. See where to locate them below:

Linux:

The file can be found in /var/opt/swxtch/swxtch-bridge.json. To edit the file, use nano as shown below:

BashCopy

```
sudo nano /var/opt/swxtch/swxtch-bridge.json
```

Single Multicast Group

An example of the swxtch-bridge.json file is shown below. Note the ha section has been added. While it may look similar to the xnic.json file, there are a number of differences. One noteworthy addition is the "producer" and "consumer" fields in the ha section.

BashCopy

```
{
  "swxtch": "10.2.128.10",
  "controlInterface": "Ethernet 2",
  "dataInterface": "Ethernet",
  "dataPort": 9999,
  "xnicType": 2,
  "dataPlaneSpecs": {
    "verbosity": 0,
    "virtualInterface": {
      "name": "swxtch-tun0",
      "ip": "172.30.0.0",
      "subnet": "255.255.0.0",
      "mtu": 4096
    },
  },
  "ha": {
    "producer": false,
    "consumer": false,
    "maxTimeToBufferPacketsMs": 50,
    "bufferSizeInPackets": 131072,
    "protocol": "swxtch"
  },
  "statsReportWait": 60
}
```

It is important to imagine configuration from the perspective of the cloudSwXtch Bridge. At default, both the producer and consumer is set to false. Switching either to true will change the direction of the high availability.

- If the producer is set to true, the cloudSwXtch Bridge is producing duplicate streams taken from on-prem applications and sending to cloudSwXtches in the cloud.
- If the consumer is set to true, the cloudSwXtch Bridge is receiving duplicate streams from the cloud. However, unlike the xNIC, it does not do the deduplication and will send both streams to the on-prem applications.

Whether it is producer or consumer, one of them has to be set to true in order to activate high availability via the cloudSwXtch Bridge.

Once a selection has been made, a user must restart the cloudSwXtch Bridge control service by running the following command:

Bridge Type 3:

BashCopy

```
sudo systemctl restart swxtch-bridge3-ctrl
```

Bridge Type 2:

BashCopy

```
sudo systemctl restart swxtch-bridge2.service
```

Multiple Multicast for cloudSwXtch Bridge

If a user wants to set up Multiple Multicast groups for cloudSwXtch Bridge, they will need to manually configure the `swxtch-bridge.json` file by changing the protocol under `ha` to `"rtp"` and adding the `"streamSpecs"` section. This process is similar to configuring the [xNIC for multiple multicast groups](#).

How to update swxtch-bridge.json file for Multiple Multicast Groups:

The user will have to make the following changes to their `swxtch-bridge.json` file found in the [single multicast group configuration](#) to match the example above. These alterations to the `swxtch-bridge.json` file should happen after [Configuring the cloudSwXtch for High Availability](#).

- 1. Change the protocol under `ha` from `"swxtch"` to `"rtp"` including the quotation marks.
- 2. For each multicast group, add the following `"streamSpecs"` section as shown below with your `parent` and `children` groups listed. **Note:** A user can enter multiple multicast groups.

BashCopy

```
    },
    "streamSpecs": {
      "mmcProducerEnable": false,
      "multipleMulticastGroups": [
        {
          "parent": "239.2.2.2:8804",
          "children": [
            "239.1.1.1:8804",
            "239.1.1.2:8804"
          ]
        },
        {
          "parent": "239.3.3.3:8804",
          "children": [
            "239.1.1.3:8804",
            "239.1.1.4:8804"
          ]
        }
      ]
    }
  ],
},
```

- 3. Save the `swxtch-bridge.json` file.
- 4. Restart the cloudSwXtch Bridge control service by running the following command in Linux:

Bridge 3

BashCopy

```
sudo systemctl restart swxtch-bridge3-ctrl
```

Bridge 2

BashCopy

```
sudo systemctl restart swxtch-bridge2.service
```

Bridge Type 2 and Type 3

Configuring cloudSwXtch Bridge Type 2 and 3 Interfaces

By default, cloudSwXtch Bridge Type 2 and Type 3 installation will attempt to resolve the interface that is routable to the cloudSwXtch. However, if a user would like to do this manually, the cloudSwXtch Bridge Type 2 can be configured in one of two ways:

- For hairpin forwarding on a **single** interface
- For bridge in the middle redirection between **two** interfaces

This section will go into the changes a user would have to make to the cloudSwXtch Bridge Type 2 and Type 3 JSON configuration file to apply the above methods.

Where to Find cloudSwXtch Bridge Configuration Files

The location of the cloudSwXtch Bridge Type 2 JSON file is `/var/opt/swxtch/swxtch-bridge.json`.

The location of the cloudSwXtch Bridge Type 3 JSON file is `/var/opt/swxtch/swxtch-bridge3-cfg.json`.

cloudSwXtch Bridge Configuration JSON Example

PowerShellCopy

```
{
  "bridgeConfig": {
    "ctrlInterfaceName": "eth0",
    "dataInterfaceName": "e636:00:02.0",
    "userInterfaceName": "e636:00:02.0",
    "nicsConfig": null,
    "swxtchCtrlIp": "10.2.128.10",
    "swxtchCtrlPort": 80,
    "swxtchDataIp": "10.2.192.116",
    "swxtchDataPort": 9999,
    "overwriteSenderId": null,
    "dataGatewayIp": null,
    "groundToCloudSubscriptions": null,
    "cloudToGroundSubscriptions": [],
    "pollingIntervalMilliseconds": 1000,
    "subscriptionsPollingIntervalMilliseconds": 100,
    "mtuSize": 1500,
    "adaptorsConfig": {},
    "overrideSrcIp": false,
    "slp": {
      "InFlightPacketCount": null,
      "PacketDeliveryRetries": null,
      "ConnectionTimeoutRetries": null,
      "DisableAutoUpdateEstimateFirstNack": null,
      "MinTimeToFirstRetransmissionRequestNs": null,
      "IncomingRetransmissionDelayNs": null,
      "AcknowledgementIntervalNs": null,
      "ChannelBondingStartPort": null,
      "NumberOfChannelBondingPorts": null,
      "ChannelRebondingStartPort": null,
      "NumberOfChannelRebondingPorts": null,
      "MinValueAvgWindow": null,
      "EnableAutoIncoming": null,
      "LatencyFactor": 2
    },
    "xdpModeData": null,
    "xdpModeUser": null,
    "cloudTunIp": null,
    "rpcPort": 10002
  },
  "ha": {
    "producer": false,
    "consumer": false,
    "bufferSizeInPackets": 131072,
    "maxTimeToBufferPacketsMs": 50,
    "protocol": "swxtch"
  },
  "streamSpecs": null
}
```

Fields in Common Explained

Below are deeper explanations for certain fields in the cloudSwXtch Bridge Type 2 and 3 config file:

- **"ctrlInterfaceName"**: NIC used for control-plane communication with cloudSwXtch
- **"dataInterfaceName"**: NIC used for the data-plane communication with cloudSwXtch. For Bridge 2, the config file uses the ethernet name. For Bridge 3, the config file uses the PCI address.
- **"userInterfaceName"**: NIC used for multicast ground traffic. For Bridge 2, the config file uses the ethernet name. For Bridge 3, the config file uses the PCI address.
- **"pathId"**: Please set this to zero.
- **"groundToCloudSubscriptions"**: Please leave blank as it is no longer necessary since ground to cloud is done dynamically via IGMP joins from the cloud client.
- **"cloudToGroundSubscriptions"**: Traffic coming into the cloudSwXtch with these addresses will be forwarded to bridge and then to the userInterface.
- **"pollingIntervalMilliseconds"**: Polling consists on a sync with the cloudSwXtch to exchange MC groups information.

Fields Only in cloudSwXtch Bridge Type 3

The main difference between Type 2 and Type 3 is the addition of an SLP (swXtch Lossless UDP) section. The default value for each SLP-related argument is "null." It is recommended to contact support@swxtch.io if you wish to override.

Using the status API call to get swxtch-bridge JSON file

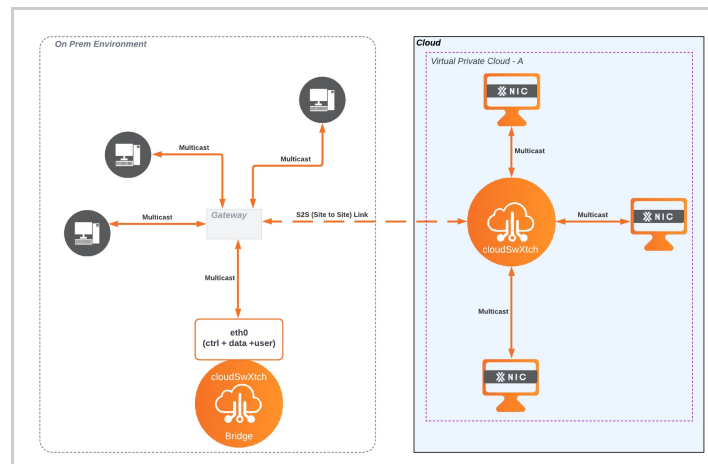
Alternatively, users can use a /status API call to get the swxtch-bridge.json file. To do this, use the following command replacing the <bridge-ctrl-ip> with the IP address of the bridge:

Bash

```
curl -x GET http://<bridge-ctrl-ip>/status
```

Copy

For a single interface



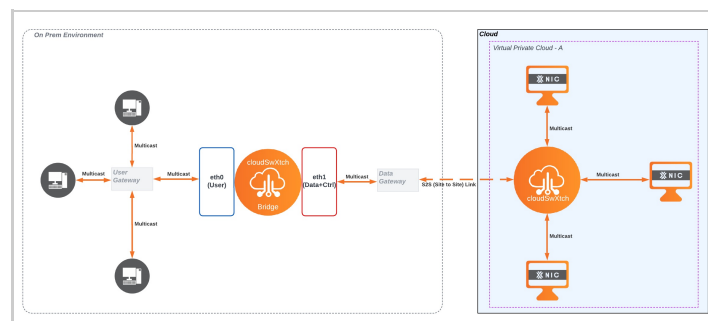
To accomplish a single interface configuration for your cloudSwXtch Bridge Type 2, users will need to specify the same InterfaceName for Ctrl, Data and User in the swxtch-bridge.json file. In the example, each are assigned to eth0.

Bash

```
"ctrlInterfaceName": "eth0",  
"dataInterfaceName": "eth0",  
"userInterfaceName": "eth0",
```

Copy

For bridge in the middle of two interfaces



Alternatively to the single interface approach, a user can decide to place the cloudSwXtch Bridge between two interfaces in order to redirect traffic from the user interface to the data interface. In the example below, the ctrlInterfaceName and the dataInterfaceName are the same (eth1) while the userInterfaceName differs (eth0).

Bash	Copy
<pre>"ctrlInterfaceName": "eth1", "dataInterfaceName": "eth1", "userInterfaceName": "eth0",</pre>	

Using Bridge Type 2 and 3 cloud to ground API to Join/Leave

Bridge Type 2 has the capability to do join and leaves from ground to cloud via an HTTP endpoint on the bridge. This will enable the forwarding of multicast traffic from cloud to ground.

To Join:

Bash	Copy
<pre>curl -X POST http://<BRIDGE_CTRL_IP>/addCloudToGround -d '{"MulticastGroups":["239.239.239.99:9000"],"UpdateConfigFile":false}'</pre>	

To Leave:

Bash	Copy
<pre>curl -X POST http://<BRIDGE_CTRL_IP>/removeCloudToGround -d '{"MulticastGroups":["239.239.239.99:9000"],"UpdateConfigFile":false}'</pre>	

Note: A user can set the UpdateConfigFile to **"true"** in order to make their configuration permanent. This means that the changes to cloudSwXtch Bridge Type 2 will persist between restarts.

Configuring Bridge Type 2 and Type 3 Static Subscriptions

The cloud to ground and ground to cloud flows are static based on entry into a json file. In order to do this, modify the bridge JSON configuration file and add the static multicast groups for either groundToCloudSubscriptions or cloudToGroundSubscriptions

The location of the cloudSwXtch Bridge Type 2 configuration file is `/var/opt/swxtch/swxtch-bridge.json`.

Modify the JSON array attribute for **"cloudToGroundSubscriptions"** or **"groundToCloudSubscriptions"** and add the appropriate multicast groups from either option.

Bash	Copy
<pre>{ "bridgeConfig": { "ctrlInterfaceName": "eth0", "dataInterfaceName": "eth1", "userInterfaceName": "eth0", "swxtchCtrlIp": "10.0.0.1", "swxtchCtrlPort": 80, "swxtchDataIp": "10.0.1.1", "swxtchDataPort": 9999, "pathId": 0, "groundToCloudSubscriptions": ["226.0.23.182:13000", "226.0.23.183:13000", "226.0.23.184:13000", "226.0.23.185:13000"], "cloudToGroundSubscriptions": ["225.0.23.182:12000", "225.0.23.183:12000", "225.0.23.184:12000", "225.0.23.185:12000"], "pollingIntervalMilliseconds": 1000 } }</pre>	

After modifying the configuration file, restart the swxtch-bridge2 service with the following command:

Bash	Copy
<pre>sudo systemctl restart swxtch-bridge2.service</pre>	

In the example above, these multicast groups will now be sent from both cloud to ground and ground to cloud at startup for bridge.

Using a specific gateway address for Bridge Type 2

By default, Bridge Type 2 will resolve the data gateway MAC address by arping the first IP address of the subnet for the data interface. However, if the gateway IP address is not there, then the dataGatewayIP field can be added into the configuration file. This will force the Bridge to resolve the gateway MAC address by using the IP address specified. In the example below, the user inserted their own data gateway IP address.

Bash	Copy
<pre>"dataGatewayIp": "192.168.1.2",</pre>	

Bridge Type 1

Configuring cloudSwXtch Bridge Type 1

When launching the cloudSwXtch Bridge Type 1, the command line arguments for the input must specify the input multicast group IP address, the IP address to use within the multicast network, and the input multicast group port.

The format for the bridge `--input` argument is:

Bash	Copy
<code>--input multicast://<multicast-group-ip>:<nic-ip>:<multicast-group-port></code>	

Multiple multicast groups can be specified by separating them with commas:

PowerShell	Copy
<code>--input multicast://<multicast-group-ip>:<nic-ip>:<multicast-group-port>,multicast://<multicast-group-ip>:<nic-ip>:<multicast-group-port></code>	

The output parameter is the IP and Port of the cloudSwXtch instance where the multicast traffic will be sent to. The format for the bridge `--output` argument is:

PowerShell	Copy
<code>--output udp://<swxtch-data-ip>:9999</code>	

The bridge application needs to know what IP address to use for the source of the multicast packets when those packets are injected into the tunnel network. This is because the network in which the bridge exists is not the same network as the tunnel network used by the application software for sending and receiving multicast traffic. This IP address should be a valid IP address in the 172.30.X.Y range and should be a unique address: i.e., not one used by another VM. This IP address in the 172.30.X.Y range shall be called the bridge source address. The format for the bridge `--override-multicast-sender-ip` argument is:

PowerShell	Copy
<code>--override-multicast-sender-ip <bridge-source-address> --last-leg</code>	

Example: Bridge Type 2 from two multicast groups to a cloudSwXtch

In this example, the system is configured such that:

- cloudSwXtch is at 10.2.192.7 (data plane) and this IP address is reachable from the machine running the swxtch-bridge application.
- NIC IP address which the swxtch-bridge will use for receiving multicast traffic is at 169.192.0.4
- The multicast groups are 239.1.1.4:8804 and 239.1.1.1:8801
- The bridge source address was chosen to be 172.30.1.1

Example command to run the bridge:

PowerShell	Copy
<code>swxtch-bridge --input multicast://239.1.1.4:169.192.0.4:8804,multicast://239.1.1.1:169.192.0.4:8801 --output udp://10.2.192.7:9999 --override-multicast-sender-ip 172.30.1.1 --last-leg</code>	

NOTE:
The bridge application assigns itself to use CPU cores 1 and 2 by default. This can be changed using the following command line parameters:

PowerShell	Copy
<code>--core 1 --core-count 2</code>	

Where core sets the starting core index (from 0) and core-count sets the number of cores to use.

Protocol Conversion and Fanout

Configuring Protocol Conversion and Fanout

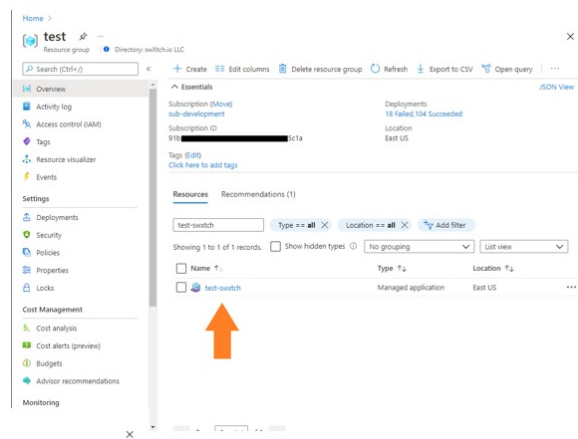
There are two options for configuring Protocol Conversion and Fanout: via wXcked Eye or via the API. For more information, please see the following articles:

- [Protocol Conversion and Fanout with wXcked Eye](#)
- [Configuration API](#)

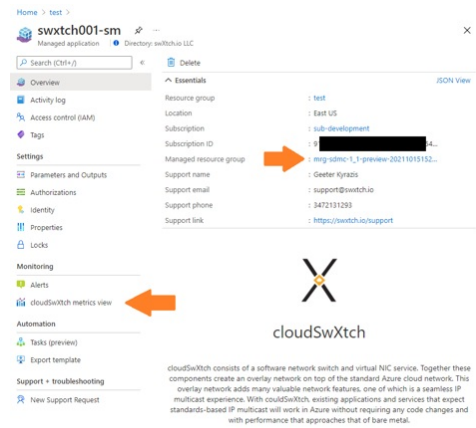
Azure Monitoring

cloudSwXtch instances will show up in your Azure Resource Groups as "Managed applications" with the name given during creation. For example, the below image shows a cloudSwXtch instance with the name "test-switch" in the resource group "test".

When you click on a cloudSwXtch instance in a resource group, you are taken to the cloudSwXtch information page for that instance. From this page you can view properties and other standard Azure component screens.



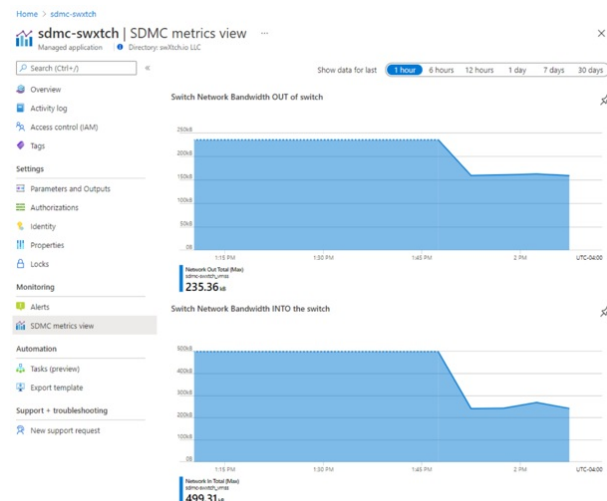
In addition to the standard Azure component sections, this screen has two sections that are unique to the cloudSwXtch managed application: metrics view and managed application resource group.



cloudSwXtch metrics view

The metrics view shows two simple graphs of the network activity of the cloudSwXtch instance. The metrics available are the total bandwidth into and out of the instance. The bandwidth units change based on the timescale chosen.

NOTE:
due to Azure idiosyncrasies, the metrics view will first show up around 15 minutes or so after a cloudSwXtch instance is first created. The swtch-top application can be used immediately.



Managed resource group

The cloudSwXtch product is delivered as a "managed application". This means that a cloudSwXtch instance lives within the customer's subscription and is made up of Azure resources (VMs, etc.) that are instantiated within the same subscription. These resources are directly billed to the subscription owner.

PRO TIP:

When a cloudSwXtch instance is created, it is assigned to the resource group selected by the creator and to an auto-generated resource group that holds the low-level components needed to compose the managed application. The creator of the instance has full access to the resource group that holds the instance and partial access to the auto-generated managed application resource group. The partial access allows the creator to see the various components and view their properties and metrics. It does not, however, allow the creator access to the internal VM instances that make up the managed application. The creator cannot directly control these resources from the portal, except to start/stop the VM.

For more details see:
[Azure managed applications overview](#)

Figure 2 - SDMC metrics view

Changing xNIC configuration settings

All xNIC configuration values are normally set by the xNIC installation script. If manual changes are made to the configuration values, the xNIC service must be restarted:

```
sudo systemctl restart swxtch-xnic
```

The configuration settings for the xNIC are located at:

- Linux: `/var/opt/swxtch/swxtch-xnic.conf`
- Windows: `<ctdb>`

The configuration file is a simple text file in `*.ini` format. The following values are available:

Key Name	Default value	Description and notes
SvcAddr	<ip-of-instance>	IPv4 address of the cloudSwXtch instance.
SvcPort	10802	Control port on cloudSwXtch instance.
VirtualInterfaceName	"swxtch-tun"	Base name of the virtual network interface. Must be < 15 characters.
VirtualInterfaceIpAddr	"172.30.0.0"	IPv4 subnet of the virtual network interface as seen from the host applications
VirtualInterfaceSubnet	"255.255.0.0"	IPv4 subnet mask
CtrlInterface	"eth0"	Network interface to use for control plane traffic.
DataInterface	"eth1"	Network interface to use for data plane traffic.
CtrlPort	10800	Local port used for control traffic <i>from</i> the SDMC switch
DataPort	9999	Local port used for data traffic <i>from</i> the SDMC switch

Prometheus Monitoring

WHAT TO EXPECT

In this article, users will learn how to integrate Prometheus and Grafana as an additional way to monitor their cloudSwXtch environment.

PREREQUISITES

The following process assumes that you already have Prometheus and Grafana installed in a docker container.

STEP ONE: Validate cloudSwXtch can create Prometheus data

On the cloudSwXtch VM, run the following command:

Plaintext	Copy
curl http://localhost/prometheus/metrics	

The output will list information about each metric with example output data. Metrics starting with `swx_core` are from the cloudSwXtch while metrics starting with `swx_xnic` are from xNICs. Since this example has only one cloudSwXtch but many VMs with xNICs, the xNIC data has multiple sample rows. Note that for brevity some of the xNIC rows returned have been deleted.

PowerShell	Copy	
<pre>swxtchadmin@dsd-core-100:~\$ curl http://localhost/prometheus/metrics # HELP swx_core_droppedPacketCountByByteLimit Bytes dropped in the swXtch # TYPE swx_core_droppedPacketCountByByteLimit counter swx_core_droppedPacketCountByByteLimit{category="swxtch_rep1",host="10.2.192.23"} 0 # HELP swx_core_droppedPacketCountByPacketLimit Packets dropped in the swXtch # TYPE swx_core_droppedPacketCountByPacketLimit counter swx_core_droppedPacketCountByPacketLimit{category="swxtch_rep1",host="10.2.192.23"} 0 # HELP swx_core_rxBridgeByteCount Bridge bytes received into the swXtch # TYPE swx_core_rxBridgeByteCount counter swx_core_rxBridgeByteCount{category="swxtch_rep1",host="10.2.192.23"} 0 # HELP swx_core_rxBridgePacketCount Bridge packets received into the swXtch # TYPE swx_core_rxBridgePacketCount counter swx_core_rxBridgePacketCount{category="swxtch_rep1",host="10.2.192.23"} 0 # HELP swx_core_rxByteCount Bytes received into the swXtch # TYPE swx_core_rxByteCount counter swx_core_rxByteCount{category="swxtch_rep1",host="10.2.192.23"} 8.797308e+07 # HELP swx_core_rxMeshByteCount Mesh bytes received into the swXtch # TYPE swx_core_rxMeshByteCount counter swx_core_rxMeshByteCount{category="swxtch_rep1",host="10.2.192.23"} 0 # HELP swx_core_rxMeshPacketCount Mesh packets received into the swXtch # TYPE swx_core_rxMeshPacketCount counter swx_core_rxMeshPacketCount{category="swxtch_rep1",host="10.2.192.23"} 0 # HELP swx_core_rxPacketCount Packets received into the swXtch # TYPE swx_core_rxPacketCount counter swx_core_rxPacketCount{category="swxtch_rep1",host="10.2.192.23"} 505406 # HELP swx_core_rxUnicastByteCount Unicast bytes received into the swXtch # TYPE swx_core_rxUnicastByteCount counter swx_core_rxUnicastByteCount{category="swxtch_rep1",host="10.2.192.23"} 0 # HELP swx_core_rxUnicastPacketCount Unicast packets received into the swXtch # TYPE swx_core_rxUnicastPacketCount counter swx_core_rxUnicastPacketCount{category="swxtch_rep1",host="10.2.192.23"} 0 # HELP swx_core_sequence swXtch sequence number # TYPE swx_core_sequence counter swx_core_sequence{category="swxtch_rep1",host="10.2.192.23"} 3480 # HELP swx_core_txBridgeByteCount Bridge bytes sent from the swXtch # TYPE swx_core_txBridgeByteCount counter swx_core_txBridgeByteCount{category="swxtch_rep1",host="10.2.192.23"} 0 # HELP swx_core_txBridgePacketCount Bridge packets sent from the swXtch # TYPE swx_core_txBridgePacketCount counter swx_core_txBridgePacketCount{category="swxtch_rep1",host="10.2.192.23"} 0 # HELP swx_core_txByteCount Bytes sent from the swXtch # TYPE swx_core_txByteCount counter swx_core_txByteCount{category="swxtch_rep1",host="10.2.192.23"} 1.71498368e+08 # HELP swx_core_txMeshByteCount Mesh bytes sent from the swXtch # TYPE swx_core_txMeshByteCount counter swx_core_txMeshByteCount{category="swxtch_rep1",host="10.2.192.23"} 0 # HELP swx_core_txMeshPacketCount Mesh packets sent from the swXtch # TYPE swx_core_txMeshPacketCount counter swx_core_txMeshPacketCount{category="swxtch_rep1",host="10.2.192.23"} 0 # HELP swx_core_txPacketCount Packets sent from the swXtch # TYPE swx_core_txPacketCount counter swx_core_txPacketCount{category="swxtch_rep1",host="10.2.192.23"} 985630 # HELP swx_core_txUnicastByteCount Unicast bytes sent from the swXtch # TYPE swx_core_txUnicastByteCount counter swx_core_txUnicastByteCount{category="swxtch_rep1",host="10.2.192.23"} 0 # HELP swx_core_txUnicastPacketCount Unicast packets sent from the swXtch # TYPE swx_core_txUnicastPacketCount counter</pre>		

```

swx_core_txUnicastPacketCount{category="swxtch_repl",host="10.2.192.23"} 0
# HELP swx_maxClientCount Maximum number of clients by license
# TYPE swx_maxClientCount gauge
swx_maxClientCount{category="swxtch"} 50
# HELP swx_numClientsConnected Number of client currently connected
# TYPE swx_numClientsConnected gauge
swx_numClientsConnected{category="swxtch"} 6
# HELP swx_xnic_byteCounters_rxMulticastCount Multicast bytes received from the swXtch into the xNIC
# TYPE swx_xnic_byteCounters_rxMulticastCount counter
swx_xnic_byteCounters_rxMulticastCount{category="swxtch_xnic",host="DSd-agent-101"} 7200
swx_xnic_byteCounters_rxMulticastCount{category="swxtch_xnic",host="DSd-agent-102"} 7.0259222e+07
swx_xnic_byteCounters_rxMulticastCount{category="swxtch_xnic",host="DSd-agent-104"} 6.9676152e+07
swx_xnic_byteCounters_rxMulticastCount{category="swxtch_xnic",host="DSd-agent-105"} 7686
swx_xnic_byteCounters_rxMulticastCount{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000018"} 7200
swx_xnic_byteCounters_rxMulticastCount{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000019"} 7200
# HELP swx_xnic_byteCounters_rxTotalCount Total bytes received from the swXtch into the xNIC
# TYPE swx_xnic_byteCounters_rxTotalCount counter
swx_xnic_byteCounters_rxTotalCount{category="swxtch_xnic",host="DSd-agent-101"} 8864
swx_xnic_byteCounters_rxTotalCount{category="swxtch_xnic",host="DSd-agent-102"} 8.6092278e+07
swx_xnic_byteCounters_rxTotalCount{category="swxtch_xnic",host="DSd-agent-104"} 8.5377816e+07
swx_xnic_byteCounters_rxTotalCount{category="swxtch_xnic",host="DSd-agent-105"} 9414
swx_xnic_byteCounters_rxTotalCount{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000018"} 8864
swx_xnic_byteCounters_rxTotalCount{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000019"} 8864
# HELP swx_xnic_byteCounters_txMulticastCount Multicast bytes sent from the xNIC into the swXtch
# TYPE swx_xnic_byteCounters_txMulticastCount counter
swx_xnic_byteCounters_txMulticastCount{category="swxtch_xnic",host="DSd-agent-101"} 0
swx_xnic_byteCounters_txMulticastCount{category="swxtch_xnic",host="DSd-agent-102"} 0
swx_xnic_byteCounters_txMulticastCount{category="swxtch_xnic",host="DSd-agent-104"} 96222
swx_xnic_byteCounters_txMulticastCount{category="swxtch_xnic",host="DSd-agent-105"} 7686
swx_xnic_byteCounters_txMulticastCount{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000018"} 7.1697362e+07
swx_xnic_byteCounters_txMulticastCount{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000019"} 0
# HELP swx_xnic_byteCounters_txTotalCount Total bytes sent from the xNIC into the swXtch
# TYPE swx_xnic_byteCounters_txTotalCount counter
swx_xnic_byteCounters_txTotalCount{category="swxtch_xnic",host="DSd-agent-101"} 0
swx_xnic_byteCounters_txTotalCount{category="swxtch_xnic",host="DSd-agent-102"} 0
swx_xnic_byteCounters_txTotalCount{category="swxtch_xnic",host="DSd-agent-104"} 111006
swx_xnic_byteCounters_txTotalCount{category="swxtch_xnic",host="DSd-agent-105"} 9414
swx_xnic_byteCounters_txTotalCount{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000018"} 8.7854514e+07
swx_xnic_byteCounters_txTotalCount{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000019"} 0
# HELP swx_xnic_latencies_count xNIC latency count
# TYPE swx_xnic_latencies_count gauge
swx_xnic_latencies_count{category="swxtch_xnic",host="DSd-agent-101"} 0
swx_xnic_latencies_count{category="swxtch_xnic",host="DSd-agent-102"} 0
swx_xnic_latencies_count{category="swxtch_xnic",host="DSd-agent-104"} 0
swx_xnic_latencies_count{category="swxtch_xnic",host="DSd-agent-105"} 0
swx_xnic_latencies_count{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000018"} 0
swx_xnic_latencies_count{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000019"} 0
# HELP swx_xnic_latencies_sum xNIC latency sum
# TYPE swx_xnic_latencies_sum gauge
swx_xnic_latencies_sum{category="swxtch_xnic",host="DSd-agent-101"} 0
swx_xnic_latencies_sum{category="swxtch_xnic",host="DSd-agent-102"} 0
swx_xnic_latencies_sum{category="swxtch_xnic",host="DSd-agent-104"} 0
swx_xnic_latencies_sum{category="swxtch_xnic",host="DSd-agent-105"} 0
swx_xnic_latencies_sum{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000018"} 0
swx_xnic_latencies_sum{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000019"} 0
# HELP swx_xnic_packetCounters_rxDroppedCount Lost packets received from the swXtch into the xNIC
# TYPE swx_xnic_packetCounters_rxDroppedCount counter
swx_xnic_packetCounters_rxDroppedCount{category="swxtch_xnic",host="DSd-agent-101"} 0
swx_xnic_packetCounters_rxDroppedCount{category="swxtch_xnic",host="DSd-agent-102"} 0
swx_xnic_packetCounters_rxDroppedCount{category="swxtch_xnic",host="DSd-agent-104"} 0
swx_xnic_packetCounters_rxDroppedCount{category="swxtch_xnic",host="DSd-agent-105"} 0
swx_xnic_packetCounters_rxDroppedCount{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000018"} 0
swx_xnic_packetCounters_rxDroppedCount{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000019"} 0
# HELP swx_xnic_packetCounters_rxMulticastCount Multicast packets received from the swXtch into the xNIC
# TYPE swx_xnic_packetCounters_rxMulticastCount counter
swx_xnic_packetCounters_rxMulticastCount{category="swxtch_xnic",host="DSd-agent-101"} 52
swx_xnic_packetCounters_rxMulticastCount{category="swxtch_xnic",host="DSd-agent-102"} 494783
swx_xnic_packetCounters_rxMulticastCount{category="swxtch_xnic",host="DSd-agent-104"} 490677
swx_xnic_packetCounters_rxMulticastCount{category="swxtch_xnic",host="DSd-agent-105"} 54
swx_xnic_packetCounters_rxMulticastCount{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000018"} 52
swx_xnic_packetCounters_rxMulticastCount{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000019"} 52
# HELP swx_xnic_packetCounters_rxTotalCount Total packets received from the swXtch into the xNIC
# TYPE swx_xnic_packetCounters_rxTotalCount counter
swx_xnic_packetCounters_rxTotalCount{category="swxtch_xnic",host="DSd-agent-101"} 52
swx_xnic_packetCounters_rxTotalCount{category="swxtch_xnic",host="DSd-agent-102"} 494783
swx_xnic_packetCounters_rxTotalCount{category="swxtch_xnic",host="DSd-agent-104"} 490677
swx_xnic_packetCounters_rxTotalCount{category="swxtch_xnic",host="DSd-agent-105"} 54
swx_xnic_packetCounters_rxTotalCount{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000018"} 52
swx_xnic_packetCounters_rxTotalCount{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000019"} 52
# HELP swx_xnic_packetCounters_txDroppedCount Lost packets sent from the xNIC into the swXtch
# TYPE swx_xnic_packetCounters_txDroppedCount counter
swx_xnic_packetCounters_txDroppedCount{category="swxtch_xnic",host="DSd-agent-101"} 0

```

```

swx_nic_packetCounters_txDroppedCount{category="swxtch_xnic",host="DSd-agent-102"} 0
swx_nic_packetCounters_txDroppedCount{category="swxtch_xnic",host="DSd-agent-104"} 0
swx_nic_packetCounters_txDroppedCount{category="swxtch_xnic",host="DSd-agent-105"} 0
swx_nic_packetCounters_txDroppedCount{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000018"} 0
swx_nic_packetCounters_txDroppedCount{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000019"} 0
# HELP swx_nic_packetCounters_txIgmpCount IGMP packets sent from the xNIC into the swXtch
# TYPE swx_nic_packetCounters_txIgmpCount counter
swx_nic_packetCounters_txIgmpCount{category="swxtch_xnic",host="DSd-agent-101"} 0
swx_nic_packetCounters_txIgmpCount{category="swxtch_xnic",host="DSd-agent-102"} 0
swx_nic_packetCounters_txIgmpCount{category="swxtch_xnic",host="DSd-agent-104"} 0
swx_nic_packetCounters_txIgmpCount{category="swxtch_xnic",host="DSd-agent-105"} 0
swx_nic_packetCounters_txIgmpCount{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000018"} 0
swx_nic_packetCounters_txIgmpCount{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000019"} 0
# HELP swx_nic_packetCounters_txMulticastCount Multicast packets sent from the xNIC into the swXtch
# TYPE swx_nic_packetCounters_txMulticastCount counter
swx_nic_packetCounters_txMulticastCount{category="swxtch_xnic",host="DSd-agent-101"} 0
swx_nic_packetCounters_txMulticastCount{category="swxtch_xnic",host="DSd-agent-102"} 0
swx_nic_packetCounters_txMulticastCount{category="swxtch_xnic",host="DSd-agent-104"} 462
swx_nic_packetCounters_txMulticastCount{category="swxtch_xnic",host="DSd-agent-105"} 54
swx_nic_packetCounters_txMulticastCount{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000018"} 504911
swx_nic_packetCounters_txMulticastCount{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000019"} 0
# HELP swx_nic_packetCounters_txTotalCount Total packets sent from the xNIC into the swXtch
# TYPE swx_nic_packetCounters_txTotalCount counter
swx_nic_packetCounters_txTotalCount{category="swxtch_xnic",host="DSd-agent-101"} 0
swx_nic_packetCounters_txTotalCount{category="swxtch_xnic",host="DSd-agent-102"} 0
swx_nic_packetCounters_txTotalCount{category="swxtch_xnic",host="DSd-agent-104"} 462
swx_nic_packetCounters_txTotalCount{category="swxtch_xnic",host="DSd-agent-105"} 54
swx_nic_packetCounters_txTotalCount{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000018"} 504911
swx_nic_packetCounters_txTotalCount{category="swxtch_xnic",host="aks-nodepool1-23164585-vmss000019"} 0
# HELP swx_nic_rxMulticastGroups_byteCount Multicast group traffic sent from the Swxtch into the xNIC - Multicast group byte count
# TYPE swx_nic_rxMulticastGroups_byteCount counter
swx_nic_rxMulticastGroups_byteCount{category="swxtch_xnic",cloudSwxtchVersion="dev.cloudswxtch.2.1.1.10",groupId="10.2.131.255",host="DSd-agent-101",hostAddress="10.2.128.29",index="3",osDistribution="Ubuntu 22.04",xNicType="t2"} 1116
swx_nic_rxMulticastGroups_byteCount{category="swxtch_xnic",cloudSwxtchVersion="dev.cloudswxtch.2.1.1.10",groupId="10.2.131.255",host="DSd-agent-101",hostAddress="10.2.128.29",index="5",osDistribution="Ubuntu 22.04",xNicType="t2"} 1116
swx_nic_rxMulticastGroups_byteCount{category="swxtch_xnic",cloudSwxtchVersion="dev.cloudswxtch.2.1.1.10",groupId="10.2.131.255",host="DSd-agent-101",hostAddress="10.2.128.29",index="6",osDistribution="Ubuntu 22.04",xNicType="t2"} 1100
swx_nic_rxMulticastGroups_byteCount{category="swxtch_xnic",cloudSwxtchVersion="dev.cloudswxtch.2.1.1.10",groupId="10.2.131.255",host="DSd-agent-101",hostAddress="10.2.128.29",index="7",osDistribution="Ubuntu 22.04",xNicType="t2"} 1100
# HELP swx_nic_rxMulticastGroups_packetCount Multicast group traffic sent from the Swxtch into the xNIC - Multicast group packet count
# TYPE swx_nic_rxMulticastGroups_packetCount counter
swx_nic_rxMulticastGroups_packetCount{category="swxtch_xnic",cloudSwxtchVersion="dev.cloudswxtch.2.1.1.10",groupId="10.2.131.255",host="DSd-agent-101",hostAddress="10.2.128.29",index="3",osDistribution="Ubuntu 22.04",xNicType="t2"} 9
swx_nic_rxMulticastGroups_packetCount{category="swxtch_xnic",cloudSwxtchVersion="dev.cloudswxtch.2.1.1.10",groupId="10.2.131.255",host="DSd-agent-101",hostAddress="10.2.128.29",index="5",osDistribution="Ubuntu 22.04",xNicType="t2"} 9
swx_nic_rxMulticastGroups_packetCount{category="swxtch_xnic",cloudSwxtchVersion="dev.cloudswxtch.2.1.1.10",groupId="10.2.131.255",host="DSd-agent-101",hostAddress="10.2.128.29",index="6",osDistribution="Ubuntu 22.04",xNicType="t2"} 4
swx_nic_rxMulticastGroups_packetCount{category="swxtch_xnic",cloudSwxtchVersion="dev.cloudswxtch.2.1.1.10",groupId="10.2.131.255",host="DSd-agent-101",hostAddress="10.2.128.29",index="7",osDistribution="Ubuntu 22.04",xNicType="t2"} 4
# HELP swx_nic_txMulticastGroups_byteCount Multicast group traffic sent from the xNIC into the Swxtch - Multicast group byte count
# TYPE swx_nic_txMulticastGroups_byteCount counter
swx_nic_txMulticastGroups_byteCount{category="swxtch_xnic",cloudSwxtchVersion="dev.cloudswxtch.2.1.1.10",groupId="10.2.131.255",host="DSd-agent-104",hostAddress="10.2.128.75",index="100",osDistribution="Windows Server 2019 Datacenter - Microsoft Windows [Version 10.0.17763.5329]",xNicType="t2"} 2988
swx_nic_txMulticastGroups_byteCount{category="swxtch_xnic",cloudSwxtchVersion="dev.cloudswxtch.2.1.1.10",groupId="10.2.131.255",host="DSd-agent-104",hostAddress="10.2.128.75",index="27",osDistribution="Windows Server 2019 Datacenter - Microsoft Windows [Version 10.0.17763.5329]",xNicType="t2"} 2853
# HELP swx_nic_txMulticastGroups_packetCount Multicast group traffic sent from the xNIC into the Swxtch - Multicast group packet count
# TYPE swx_nic_txMulticastGroups_packetCount counter
swx_nic_txMulticastGroups_packetCount{category="swxtch_xnic",cloudSwxtchVersion="dev.cloudswxtch.2.1.1.10",groupId="10.2.131.255",host="DSd-agent-104",hostAddress="10.2.128.75",index="100",osDistribution="Windows Server 2019 Datacenter - Microsoft Windows [Version 10.0.17763.5329]",xNicType="t2"} 18
swx_nic_txMulticastGroups_packetCount{category="swxtch_xnic",cloudSwxtchVersion="dev.cloudswxtch.2.1.1.10",groupId="10.2.131.255",host="DSd-agent-104",hostAddress="10.2.128.75",index="27",osDistribution="Windows Server 2019 Datacenter - Microsoft Windows [Version 10.0.17763.5329]",xNicType="t2"} 9

```

If successful (there is an output), continue on to updating your Prometheus Directory for cloudSwXtch.

STEP TWO: Update Prometheus Directory for cloudSwXtch

Attached is an example prometheus.yaml file with an cloudSwXtch job name configuration.

prometheus	732 Byte 📄
------------	----------------------------

1. Open the example **prometheus.yml** file and copy lines 26-36.
2. Paste those lines into your existing **prometheus.yml** file.
3. Update the cloudSwXtch **targets** line that has " 127.0.0.1:80" and put in the IP address of the cloudSwXtch in place of the localhost IP.

- a. **Please note:** If Prometheus and cloudSwXtch are on the same VM, then the localhost IP (127.0.0.1) will still work.

```

26 - job_name: swxtch
27   honor_timestamps: true
28   scrape_interval: 5s
29   scrape_timeout: 2s
30   metrics_path: /prometheus/metrics
31   scheme: http
32   follow_redirects: true
33   enable_http2: true
34   static_configs:
35     - targets:
36       - 127.0.0.1:80
37

```

4. Run the following docker command to run it in VM. If you decide to run it this way, you will need to run it after every reboot or when you close your window. Please use this method when testing in order to limit the amount of records added to the Prometheus database.

Plaintext	Copy
docker run --network host -v ~/prometheus:/etc/prometheus prom/prometheus	

5. Use this command to run Prometheus automatically upon reboot. (Preferred method for a production environment.)

Plaintext	Copy
docker run --network host --restart always -v ~/prometheus:/etc/prometheus prom/prometheus	

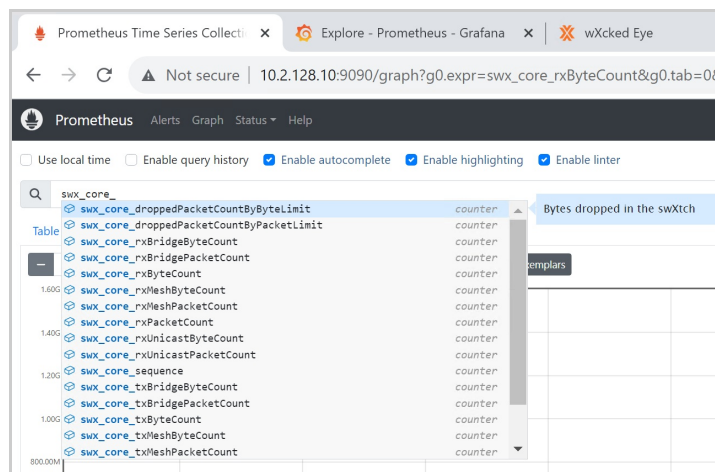
STEP THREE: Access Prometheus UI

In order to access the Prometheus UI, users should open a browser on their Windows machine in the same VNET and enter the following URL:

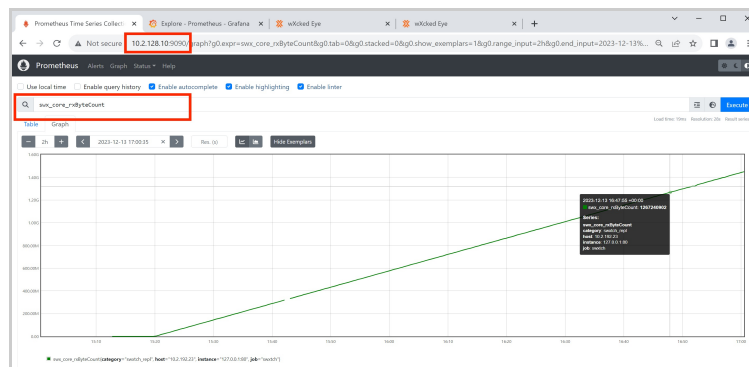
Plaintext	Copy
http://<prometheus-IP>:9090/	

Note: Replace the <prometheus-IP> with the IP address of your Prometheus instance.

Users can enter the prefix "swx" into the search field to see a list of data fields related to the cloudSwXtch (swx_core) and its xNICs (swx_xnic).



In the example below, the user has chosen **swx_core_rxByteCount** as their data field.



By selecting **Execute**, users will be able to populate the table with the desired data. **Note:** Producers and consumers must be running in order to see data. In the example above, a user can select **Execute** multiple times and notice that the number in the orange box grow in size.

For a list of available data fields, view the dropdown section here:

prometheus/metrics

URL

Plaintext

Copy

http://<core>/prometheus/metrics

Parameters

None

Request

None

Response

200 - Success

Data Fields

Metric	Description	Type
swx_core_droppedPacketCountByByteLimit	Bytes dropped in the swXtch	counter
swx_core_droppedPacketCountByPacketLimit	Packets dropped in the swXtch	counter
swx_core_rxBridgeByteCount	Bridge bytes received into the swXtch	counter
swx_core_rxBridgePacketCount	Bridge packets received into the swXtch	counter
swx_core_rxByteCount	Bytes received into the swXtch	counter
swx_core_rxMeshByteCount	Mesh bytes received into the swXtch	counter
swx_core_rxMeshPacketCount	Mesh packets received into the swXtch	counter
swx_core_rxPacketCount	Packets received into the swXtch	counter
swx_core_rxUnicastByteCount	Unicast bytes received into the swXtch	counter
swx_core_rxUnicastPacketCount	Unicast packets received into the swXtch	counter
swx_core_sequence	swXtch sequence number	counter
swx_core_txBridgeByteCount	Bridge bytes sent from the swXtch	counter
swx_core_txBridgePacketCount	Bridge packets sent from the swXtch	counter
swx_core_txByteCount	Bytes sent from the swXtch	counter
swx_core_txMeshByteCount	Mesh bytes sent from the swXtch	counter
swx_core_txMeshPacketCount	Mesh packets sent from the swXtch	counter
swx_core_txPacketCount	Packets sent from the swXtch	counter
swx_core_txUnicastByteCount	Unicast bytes sent from the swXtch	counter
swx_core_txUnicastPacketCount	Unicast packets sent from the swXtch	counter
swx_xnic_activeConnectionCount	xNIC active connections	gauge
swx_xnic_byteCount		
swx_xnic_byteCounters_rxMulticastCount	Multicast bytes received from the swXtch into the xNIC	counter
swx_xnic_byteCounters_rxTotalCount	Total bytes received from the swXtch into the xNIC	counter
swx_xnic_byteCounters_txMulticastCount	Multicast bytes sent from the xNIC into the swXtch	counter
swx_xnic_byteCounters_txTotalCount	Total bytes sent from the xNIC into the swXtch	counter
swx_xnic_latencies_count	xNIC latency count	gauge
swx_xnic_latencies_sum	xNIC latency sum	guage
swx_xnic_maxActiveConnections	xNIC max number of active connections	guage

swx_xnic_packetCounters_rxDroppedCount	Lost packets received from the swXtch into the xNIC	counter
swx_xnic_packetCounters_rxMulticastCount	Multicast packets received from the swXtch into the xNIC	counter
swx_xnic_packetCounters_rxTotalCount	Total packets received from the swXtch into the xNIC	counter
swx_xnic_packetCounters_txDroppedCount	Lost packets sent from the xNIC into the swXtch	counter
swx_xnic_packetCounters_txIcmpCount	IGMP packets sent from the xNIC into the swXtch	counter
swx_xnic_packetCounters_txMulticastCount	Multicast packets sent from the xNIC into the swXtch	counter
swx_xnic_packetCounters_txTotalCount	Total packets sent from the xNIC into the swXtch	counter
swx_xnic_rxHaCounters_egressByteCount	HA bytes sent	counter
swx_xnic_rxHaCounters_egressPacketCount	HA packets sent	counter
swx_xnic_rxHaCounters_enqueueFailureCount	HA enqueue failure count	counter
swx_xnic_rxHaCounters_outputStreamLossCount	HA output stream lose packets	counter
swx_xnic_rxHaCounters_paths_ingressByteCount	HA Bytes received	counter
swx_xnic_rxHaCounters_paths_ingressPacketCount	HA Packets received	counter
swx_xnic_rxHaCounters_paths_missingPacketCount	HA Packets missing	counter
swx_xnic_rxHaCounters_paths_usedPacketCount	HA Packets used	counter
swx_xnic_rxMulticastGroups_byteCount	Multicast Groups bytes sent from the SwXtch into the xNIC	counter
swx_xnic_rxMulticastGroups_packetCount	Multicast Groups packets sent from the SwXtch into the xNIC	counter
swx_xnic_timestamp	timestamp	counter
swx_xnic_txMulticastGroups_byteCount	Multicast Groups bytes sent from the xNIC into the SwXtch	counter
swx_xnic_txMulticastGroups_packetCount	Multicast Groups packets sent from the xNIC into the SwXtch	counter
swx_xnic_usedPacketCount	Packets	counter

STEP FOUR: Access Grafana UI

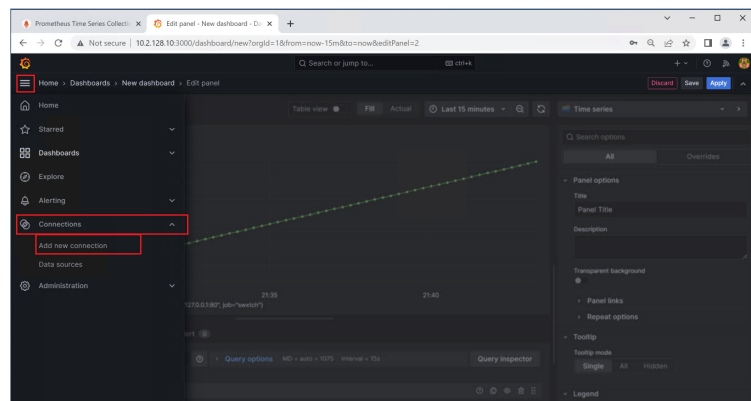
Alternatively, users can also use Grafana as another method for viewing cloudSwXtch metrics.

1. In a Windows machine on the same VNET, open a browser and enter the following URL:

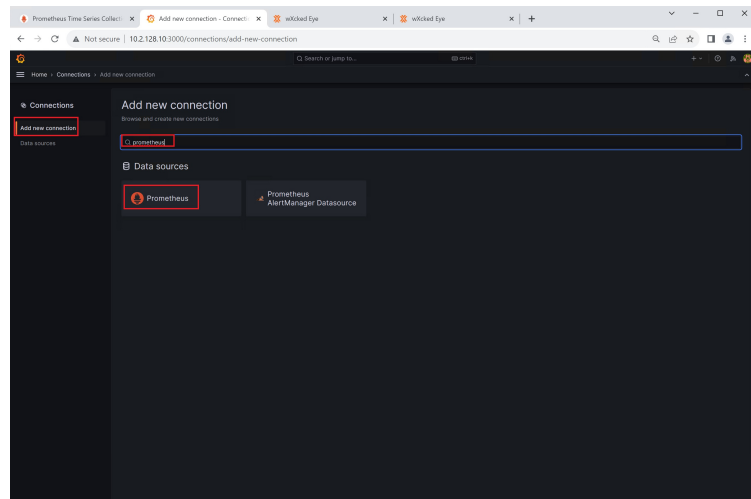
Plaintext	Copy
http://<Grafana-IP>:3000/	

Note: Replace the <Grafana-IP> with the IP address of your Grafana instance.

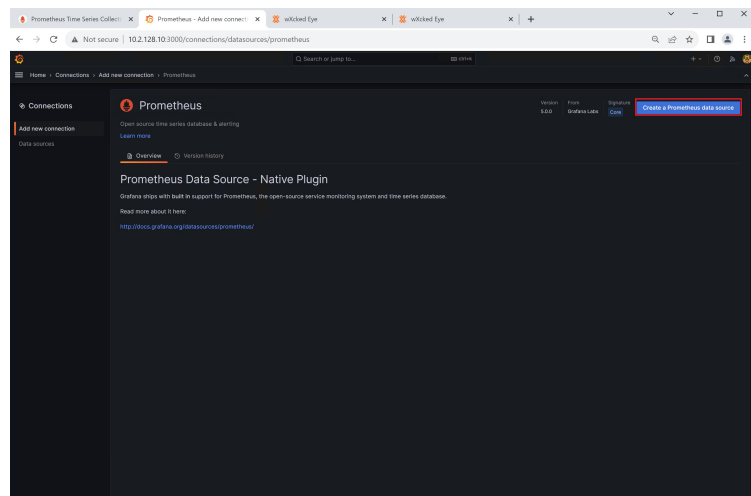
2. Sign in as a user **admin** and the password **admin**.
3. Click on the three horizontal lines next to **Home** to get additional options.
4. Select **Connection**.
5. Click **Add new connection**.



6. Search **Prometheus** on the **Add new connection** page and select it from the options.

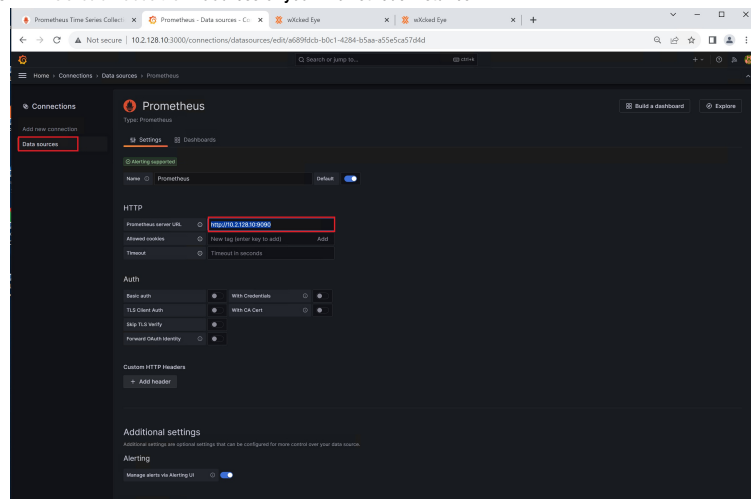


7. Click **Create a Prometheus data source**.

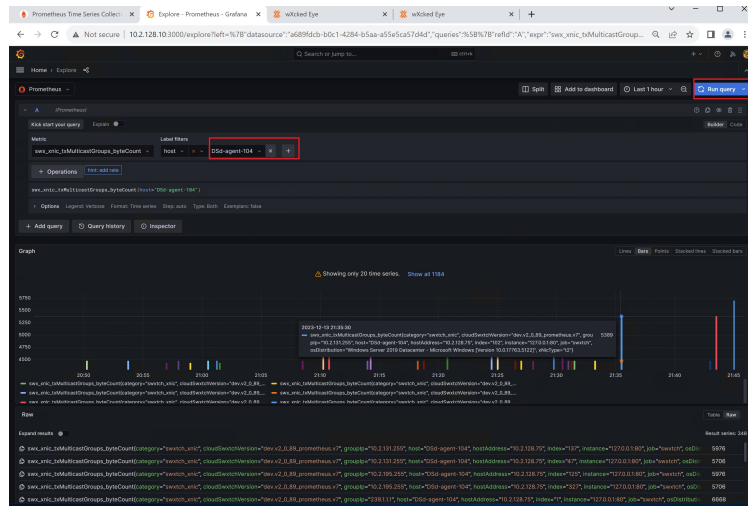


8. Go to **Data Sources** and select the **Prometheus** data source created.

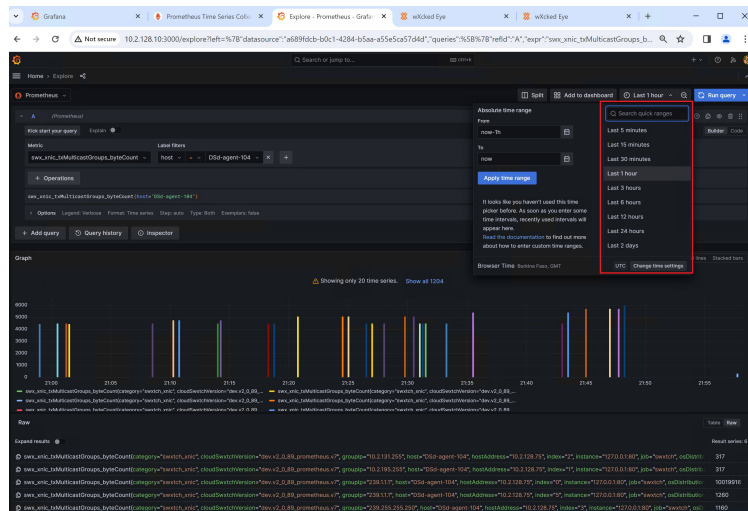
9. Enter the **Prometheus server URL**. This should include the **IP address** of your Prometheus instance.



10. Click **Explore**.



If desired, a user can also change the amount of time in the top of the window.



Testing cloudSwXtch

WHAT TO EXPECT

In this article, users will learn about the various tools that are available to them test their cloudSwXtch multicast network.

cloudSwXtch Testing Tools

Included within the xNIC installation are four utilities that can be used to verify both the functionality and performance of their cloudSwXtch network.

- **swxtch-perf**: Simulates traffic movement throughout the cloudSwXtch network assigning machines with the xNIC installed as producers and consumers
- **swxtch-top**: Displays detailed system statistics from the console of any VM that has the xNIC software installed. This includes data regarding the xNIC, the cloudSwXtch Bridge, streams, mesh, and high availability.
- **swxtch-tcpdump**: As our version of tcpdump, this tool helps capture multicast packets sent to and from the cloudSwXtch with logic to decode our own head and display the original MC payload.
- **swxtch-where**: Allows users to call for hardware information regarding their cloudSwXtch VM

Each of the utilities above can be run from a VM which has the xNIC software installed. Detailed information can be found for each by entering in the **--help** command-line argument.

wXcked Eye for Monitoring

In addition to viewing cloudSwXtch statistics in swxtch-top, users can also use wXcked Eye from the web browser of a VM in the cloudSwXtch network. For more information, see [Using wXcked Eye for cloudSwXtch](#).

Universal Third-Party Test Tools

There are a number of universal third-party tools available to test the fidelity of your cloudSwXtch network. For more information on these alternative tools, see our [Universal Third-Party Test Tools](#) article.

swxtch-perf

Overview

To simulate traffic movement throughout the cloudswitch overlay network you can use swxtch-perf to create producer and consumers on machines with the xNIC installed.

swxtch-perf producer has multiple parameters that can be configured to generate different traffic flows. There can be multiple instances of swxtch-perf generating traffic on a single machine.

None	Copy
swxtch-perf producer --sendto <MC_ADDRESS:DEST_PORT> --nic <NETWORK_INTERFACE>	

swxtch-perf consumer will pick up the traffic generated by the producer(s) in the network.

None	Copy
swxtch-perf consumer --recvfrom <MC_ADDRESS:DEST_PORT> --nic <NETWORK_INTERFACE>	

NOTE

<MC_ADDRESS> = Multicast Address
<DEST_PORT> = Destination Port
<NETWORK_INTERFACE> = Network Interface where xNIC concted to. The network interface does not have to be specified in xNic V1, but must be specified in xNic V2. (See [xNIC Linux Installation](#) for V1 and V2 differences.

swxtch-perf

For a quick view at the functionality and usage of swxtch-perf use -h or -help .

None	Copy
swxtch-perf -h Usage: swxtch-perf [options] command	
Positional arguments: command [producer consumer] suported commands	
Optional arguments: -h --help shows help message and exits [default: false] -v --version prints version information and exits [default: false] --nic name of NIC to use this is Mandatory for swxtch-perf to work. --recvfrom IP:Port The IP and Port where packets come from [default: "239.5.69.2:10000"] --sendto IP:Port The IP and Port where packets are sent to [default: "239.5.69.2:10000"] --ssm_include (consumer command only) List of SSM addresses to include (i.e. 192.168.2.1 193.168.2.4) [nargs: 1 or more] --ssm_exclude (consumer command only) List of SSM addresses to exclude (i.e. 192.168.2.1 193.168.2.4) [nargs: 1 or more] --payload_length (producer command only) number of bytes for the multicast udp payload [default: 100] --total_pkts Total packets to send/receive. To run without this limit use 0 [default: 0] --pps (producer command only) packet-rate or packet per seconds [default: 1] --seconds Number of seconds to run the application. To run without this limit use 0 [default: 0] --loopback Receives packets from --recvfrom and sends packets to --sendto [default: false] --generic (consumer command only) to consume generic packets [default: false] --latency Enables timestamp propagation and measurement of latency [default: false] --broadcast Enables broadcast packets in NIC, this overrides IP argument [default: false] --generic-broadcast Sends broadcast packets to 255.255.255.255, valid only with --broadcast argument [default: false] --broadcast-port Port for broadcast traffic, valid only with --broadcast argument [default: 10000] --rtt-latency Enables timestamp propagation and measurement of RTT/2 where RTT = round trip time [default: false] --one-way-latency Enables timestamp propagation and measurement of one way latency [default: false] --latency-buckets Enables histogram of latency. Use with --latency [default: false] --dbg Enables more information in the logs [default: false] --show-full-packet-bps Shows the bps with all headers included	

Parameters

Argument	Description	Default Value	Valid Range	Machine Type	Operating System
h	Shows commands that are available.				All
v	Shows version.			Both	All
nic	Specify which network interface xNIC will listen to this command is Mandatory.		--	Both	All
recvfrom	Specify the multicast group and port to listen for packets IPv4 addresses are valid; Ports: 1024 <= x <= 65535. Mandatory for Consumer Mode and Multicast.			Consumer	All
sendto	Specify the multicast group and port to send packets, mandatory for producer if using multicast.	All	IPv4 addresses are valid; Ports: 1024 <= x <= 65535 Mandatory for Producer Mode and Multicast.	Producer	All
ssm_include	List of SSM addresses to include (i.e. 192.168.2.1 193.168.2.4)		1 or more	Consumer	All
ssm_exclude	List of SSM addresses to exclude (i.e. 192.168.2.1 193.168.2.4)		1 or more	Consumer	All
payload_length	Number of bytes per packet.	100	8 and 65475	Producer	All
total_pkts	Number packets to receive or send before exiting iperf.	0	8 and 3750	Producer	Windows
pps	packet-rate or packets per second.	1	100000	Producer	All
seconds	Number of seconds to run the application, use 0 to run without a limit.	0		Both	Windows
loopback	Receives packets from recvfrom and sends packets to sendto.	false	true:false	Both	All
generic	Consume generic packets.	false	true:false	Consumer	All
latency	Enables timestamp propagation and measurement of latency.	false	true:false	Both	Linux
broadcast	Sets swtch-perf to use normal broadcast mode, when sending it will use the IP of the --nic argument.	false	true:false	Both	All
generic-broadcast	Sets iperf to use broadcast mode using the IP of 255.255.255.255.	false	true:false	Both	All
broadcast-port	Sets port to be used for broadcast, and is only valid with --broadcast and --generic-broadcast argument and is Mandatory for --broadcast --generic-broadcast .		Ports: 1024 <= x <= 65535	Both	All
rtt-latency	Enables timestamp propagation and measurement of RTT/2 where RTT = round trip time	false			Windows
one-way-latency	Enables timestamp propagation and measure of one way latency	false			Windows
latency-buckets	Enables histogram of latency. Use with --latency	false			Windows
dbg	Enables more information in the logs	false			All
show-full-packet-bps	Shows the bps with all headers included			Both	All

Multicast - Example

These examples can be run from one machine or across multiple machines. Parameters for NIC names assume default installation options.

EXAMPLE

Single Producer, Single Consumer, and one multicast group

Run this command on a VM to create a multicast group on the address 230.1.1.1 and port 3490 :

None	Copy
Linux: swtch-perf producer --sendto 239.1.1.1:3490 --pps 1000 --nic swtch-tun0 Windows: swtch-perf producer --sendto 239.1.1.1:3490 --pps 1000 --nic swtch-tun	

Example with results:

None	Copy
swtch-perf producer --sendto 239.1.1.1:3490 --pps 1000 --nic swtch-tun0 Trying to reach a packet-rate of 1000 pps swtch-perf producer threads started... Ctrl+C to exit. ----- ----- TOTALS THIS PERIOD TX PKTS TX BYTES TX DROPS TX-PPS TX-bps TX-DPS ----- ----- ----- ----- ----- ----- 1,283 128KB 0 1.28K 1.0Mbps 0 2,274 227KB 0 991 792Kbps 0 3,267 326KB 0 993 794Kbps 0 4,262 426KB 0 995 796Kbps 0	

Run this command on one of the VMs to listen to traffic on the Multicast Address 230.1.1.1 port 13490 :

None	Copy
Linux: swtch-perf consumer --recvfrom 239.1.1.1:3490 --nic swtch-tun0 Windows: swtch-perf consumer --recvfrom 239.1.1.1:3490 --nic swtch-tun	

Example with results:

None

Copy

testadmin@DSd-agent-102:~\$ swtch-perf consumer --recvfrom 239.1.1.1:3490 --nic swtch-tun0
swtch-perf consumer threads started... Ctrl+C to exit.

TOTALS			THIS PERIOD		
RX PKTS	RX BYTES	RX DROPS	RX-PPS	RX-bps	RX-DPS
0	0B	0	0	0bps	0
0	0B	0	0	0bps	0
0	0B	0	0	0bps	0
330	33.00KB	0	330	264Kbps	0
1,326	132KB	0	996	796Kbps	0
2,328	232KB	0	1.00K	801Kbps	0
3,330	333KB	0	1.00K	801Kbps	0
4,332	433KB	0	1.00K	801Kbps	0
5,328	532KB	0	996	796Kbps	0
6,330	633KB	0	1.00K	801Kbps	0

- To add more consumers you simply run the same swtch-perf command on new VMs.

Broadcast - Example

These examples can be run from one machine or across multiple machines. Parameters for NIC names assume default installation options.

EXAMPLE

Single Producer, Single Consumer, and broadcast

Run this command on a VM to create a broadcast

None	Copy
Linux: swtch-perf producer --broadcast --nic eth1 --pps 1000 --broadcast-port 1234 Windows: swtch-perf producer --broadcast --nic 'Ethernet 2' --pps 1000 --broadcast-port 1234	

Example with results:

None	Copy
<pre>PS C:\Users\testadmin> swtch-perf producer --broadcast --nic 'Ethernet 2' --pps 1000 --broadcast-port 1234 Config: Sending traffic to broadcast address. Ip Address: 10.2.195.255 Port : 10000 Interface IP Address: 45 Running without a total packet counter limit Running the application without a timing limit Sent 972 total packets, throughput: 890.383 pkts/sec Sent 2047 total packets, throughput: 993.128 pkts/sec Sent 3123 total packets, throughput: 991.82 pkts/sec Sent 4198 total packets, throughput: 990.419 pkts/sec</pre>	

Run this command on one of the VMs to listen for broadcast

None	Copy
Linux: swtch-perf consumer --broadcast --nic eth1--pps 1000 Windows: swtch-perf consumer --broadcast --nic 'Ethernet 3' --pps 1000	

swtch-top

WHAT TO EXPECT

swtch-top is one of the utility applications included in xNIC installation. It can be run from the console of any VM that has the xNIC software installed, displaying real-time statistics of an attached cloudSwXtch instance. This includes data regarding mesh, high availability, multicast and PTP.

In this article, you will learn how to navigate through the different pages in swtch-top and get better visibility on how data flows in your cloudSwXtch instance.

Running swtch-top

Depending on your operating system, you can use certain commands to run swtch-top on your VM.

For both Windows and Linux agents (xNICs), users can enter the following into the terminal:

BashCopy

```
swtch-top --swxtch <cloudSwXtch-IP>
```

Example:

BashCopy

```
swtch-top --swxtch 10.5.1.6
```

From the cloudSwXtch, users can enter the following command:

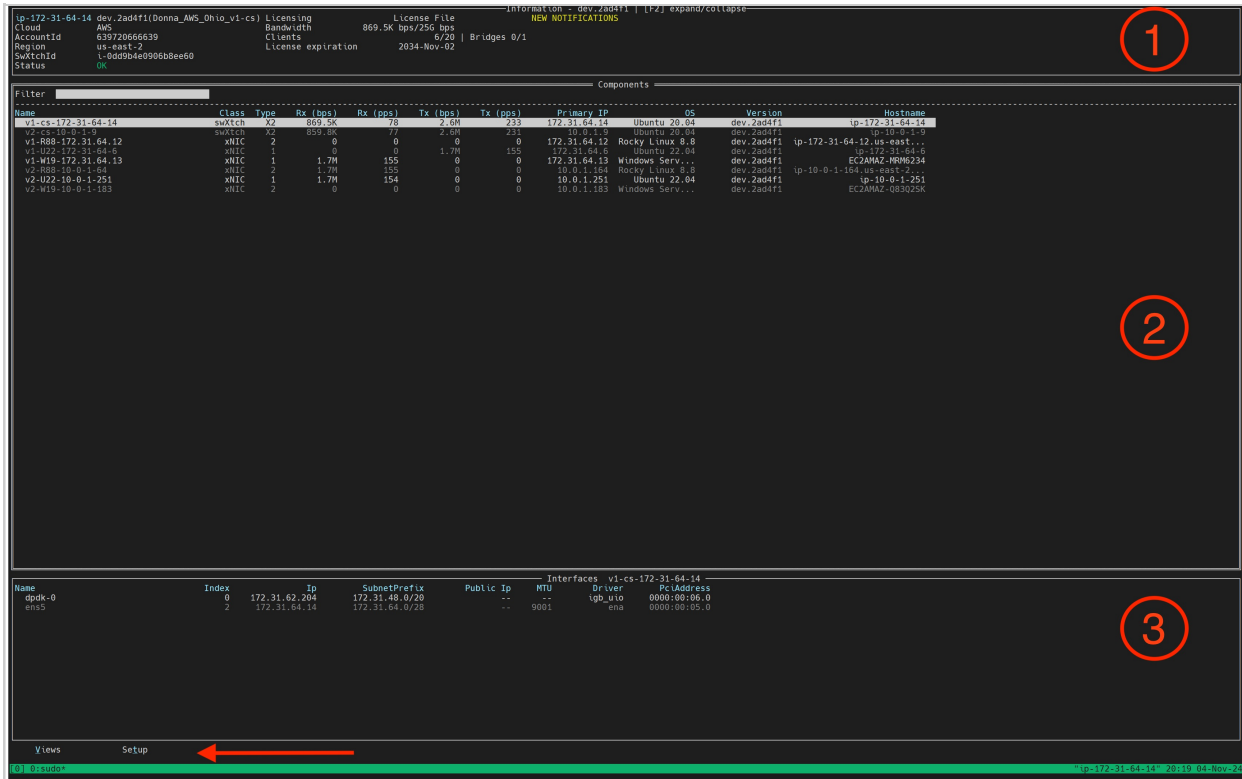
BashCopy

```
swtch-top
```

Updating cloudSwXtch for swtch-top

The following screenshots have been taken on the latest version of cloudSwXtch. To learn how to upgrade your cloudSwXtch, please see the article, [Upgrading cloudSwXtch](#).

Navigating swtch-top Dashboard



The swtch-top dashboard is organized into 3 panels as shown in the screenshot above. While the top panel will remain static, displaying information regarding the cloudSwXtch, the bottom 2 panels will change depending on the selected view. The swtch-top dashboard has 11 different views:

- 1. Components
- 2. Streams
- 3. StreamLinks
- 4. Adaptors
- 5. HA

- 6. Lossless
- 7. Settings
- 8. PTP
- 9. Subscriptions
- 10. Notifications
- 11. Configuration

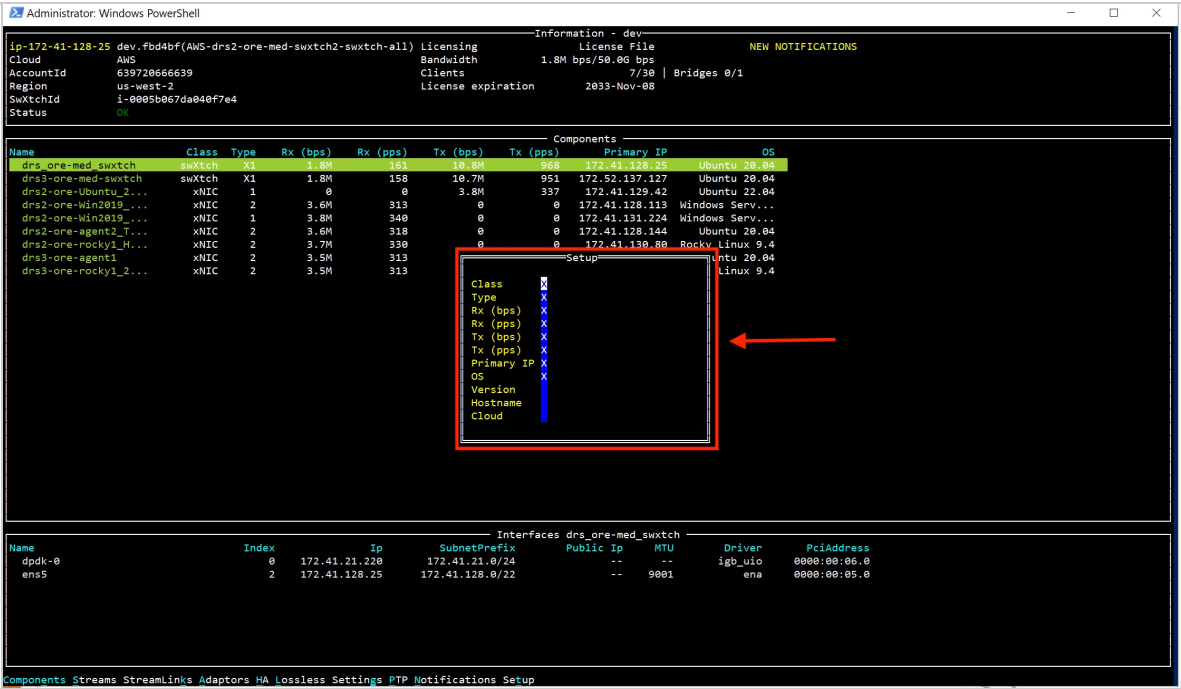
The default view is Components. To switch between views, either click on the view's name in the navigation bar OR press CTRL + the highlighted letter in the navigation bar. For example, to toggle to "StreamLinks," enter CTRL + K.

Using the Setup Function

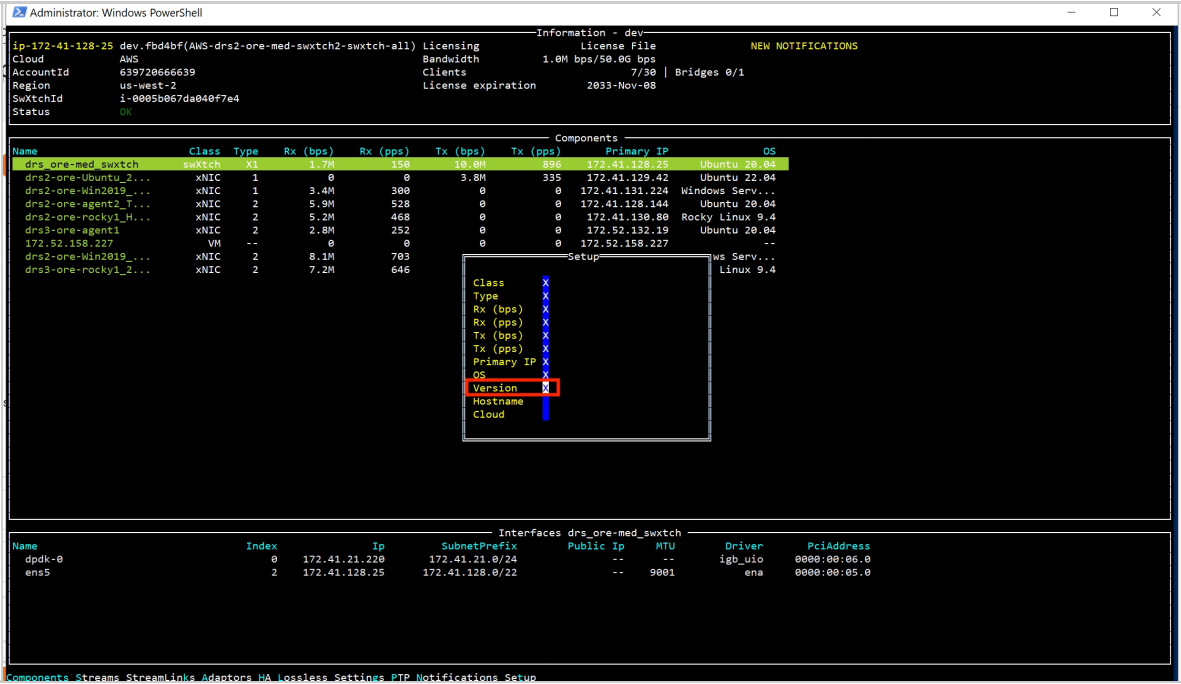
In addition to navigating between different switch-top views, the navigation bar also includes the Setup function. This enables users to modify what columns are visible in each of the views and the width of those columns. **Please note:** These changes will persist between switch-top sessions.

To do this:

- 1. Hit CTRL + T (Setup) on your keyboard to open the Setup menu.



- 2. Use the up and down directional keys to highlight the column you wish to add.
- 3. Hit Enter to select it from the menu. An X will appear.



- 4. Hit OK to confirm your selection.

Administrator: Windows PowerShell

Information - dev

ip-172-41-128-25 dev.fbd4bf(AWS-drs2-ore-med-swxtch2-swxtch-all) Licensing License File NEW NOTIFICATIONS

Cloud AWS Bandwidth 2.4M bps/50.0G bps

AccountId 63972066639 Clients 7/30 | Bridges 0/1

Region us-west-2 License expiration 2033-Nov-08

SwXtchId i-0005b067da040f7e4

Status OK

Name	Class	Type	Rx (bps)	Rx (pps)	Tx (bps)	Tx (pps)	Primary IP	OS
drs2-ore-med-swxtch	swxtch	x1	2.4M	232	4.8M	425	172.41.128.25	Ubuntu 20.04
drs2-ore-Ubuntu_2...	xNIC	1	0	0	0	0	172.41.129.42	Ubuntu 22.04
drs2-ore-Win2019...	xNIC	1	2.6M	230	0	0	172.41.131.224	Windows Serv...
drs2-ore-agent2_T...	xNIC	2	2.6M	230	0	0	172.41.128.144	Ubuntu 20.04
drs2-ore-rocky1_H...	xNIC	2	2.6M	232	0	0	172.41.130.80	Rocky Linux 9.4
drs3-ore-agent1	xNIC	2	2.6M	235	0	0	172.52.132.19	Ubuntu 20.04
172.52.158.227	VM	--	0	0	0	0	172.52.158.227	--
drs2-ore-Win2019...	xNIC	2	2.7M	237	0	0	172.41.128.113	Windows Serv...
drs3-ore-rocky1_2...	xNIC	2	2.6M	232	0	0	172.52.139.237	Rocky Linux 9.4

Setup

Tx (bps) X

Tx (pps) X

Primary IP X

OS X

Version X

Hostname X

Cloud X

Region X

Instance X

OK Cancel

Name	Index	Ip	SubnetPrefix	Public Ip	MTU	Driver	PciAddress
dpdk-0	0	172.41.21.220	172.41.21.0/24	--	--	igb_uio	0000:00:06.0
ens5	2	172.41.128.25	172.41.128.0/22	--	9001	ena	0000:00:05.0

Components Streams StreamLinks Adaptors HA Lossless Settings PTP Notifications Setup

The new column should now appear in the swxtch-top view.

Administrator: Windows PowerShell

Information - dev

ip-172-41-128-25 dev.fbd4bf(AWS-drs2-ore-med-swxtch2-swxtch-all) Licensing License File NEW NOTIFICATIONS

Cloud AWS Bandwidth 1.4M bps/50.0G bps

AccountId 63972066639 Clients 7/30 | Bridges 0/1

Region us-west-2 License expiration 2033-Nov-08

SwXtchId i-0005b067da040f7e4

Status OK

Name	Class	Type	Rx (bps)	Rx (pps)	Tx (bps)	Tx (pps)	Primary IP	OS	Version
drs2-ore-med-swxtch	swxtch	x1	1.3M	116	2.8M	701	172.41.128.25	Ubuntu 20.04	dev.fbd4bf
drs2-ore-Ubuntu_2...	xNIC	1	0	0	2.6M	230	172.41.129.42	Ubuntu 22.04	dev.fbd4bf
drs2-ore-Win2019...	xNIC	1	1.8M	160	0	0	172.41.131.224	Windows Serv...	dev.fbd4bf
drs2-ore-agent2_T...	xNIC	2	2.0M	181	0	0	172.41.128.144	Ubuntu 20.04	dev.fbd4bf
drs2-ore-rocky1_H...	xNIC	2	2.7M	239	0	0	172.41.130.80	Rocky Linux 9.4	dev.fbd4bf
drs3-ore-agent1	xNIC	2	2.6M	235	0	0	172.52.132.19	Ubuntu 20.04	dev.fbd4bf
172.52.158.227	VM	--	0	0	0	0	172.52.158.227	--	--
drs2-ore-Win2019...	xNIC	2	2.1M	180	0	0	172.41.128.113	Windows Serv...	dev.fbd4bf
drs3-ore-rocky1_2...	xNIC	2	2.7M	237	0	0	172.52.139.237	Rocky Linux 9.4	dev.fbd4bf

Name	Index	Ip	SubnetPrefix	Public Ip	MTU	Driver	PciAddress
dpdk-0	0	172.41.21.220	172.41.21.0/24	--	--	igb_uio	0000:00:06.0
ens5	2	172.41.128.25	172.41.128.0/22	--	9001	ena	0000:00:05.0

Components Streams StreamLinks Adaptors HA Lossless Settings PTP Notifications Setup

Panel 1: Information

Information - dev

ip-172-41-128-25 dev.fbd4bf(AWS-drs2-ore-med-swxtch2-swxtch-all) Licensing License File NEW NOTIFICATIONS

Cloud AWS Bandwidth 1.7M bps/50.0G bps

AccountId 63972066639 Clients 7/30 | Bridges 0/1

Region us-west-2 License expiration 2033-Nov-08

SwXtchId i-0005b067da040f7e4

Status OK

The first panel of the swxtch-top dashboard provides users with information regarding their cloudSwXtch as well as their subscription plan. Each cloud provider will have alternative titles for some of the listed items but for the most part, the information is the same.

On the left side of the section:

- cloudSwXtch name
- Cloud (Azure, AWS, GCP, OCI)
- SubscriptionID (Azure), Account ID (AWS), or ID (GCP)
- ResourceGroupName (Azure) or Region (AWS and GCP)
- SwXtchID
- Status

On the right side of the section:

- Licensing Type
- Bandwidth Allotment
- Number of Clients and Bridges
- License Expiration Date

When there are new notifications under the Notifications view, a "NEW NOTIFICATIONS" message will also display in this box.

For more information regarding licensing, please read the [cloudSwXtch System Requirements](#) article.

Panel 2 and 3: Views

Panel 2 and 3 defaults to the Components view and is shown in the picture above. However, the display changes based on the selections at the navigation bar. To change views, key in CTRL + the associated letter for that view. **Please note:** Not all views have a 3rd panel.

Filter Bar

At the top of each view's main panel, there is a white search "Filter" bar. This allows users to filter through the contents in the current view. Simply type the information you would like to filter the list by, followed by an asterisk (*), and hit Enter. For example, **v1*** to see all the fields that have "v1" in in.

Note: The filter bar is case-sensitive.

Administrator: Windows PowerShell

Information -

ip-172-31-64-14

dev.2cid29(Donna_AWS_Ohio_v1-cs)

Licensing

License File

Cloud

AccountID

Region

SwXtchId

Status

AWS

63972066639

us-east-2

i-0dd9b4e906b8ee60

OK

Bandwidth

0.0 bps/25.06 bps

Clients

6/20

Bridges

0/1

License expiration

2034-Nov-02

Filter

Components

Name	Class	Type	Rx (bps)	Rx (pps)	Tx (bps)	Tx (pps)	Primary IP	OS
v1-cs-172-31-64-14	swXtch	X1	1.5M	955.0	0.1M	4.0K	172.31.64.14	Ubuntu 20.04
v2-cs-10-0-1-9	swXtch	X1	1.5M	1.0K	7.7M	5.0K	10.0.1.9	Ubuntu 20.04
v1-R88-172.31.64.12	xNIC	1	1.5M	1.0K	0.0	0.0	172.31.64.12	Rocky Linux 8.8
v1-U22-172.31-64-6	xNIC	1	0.0	0.0	3.1M	2.0K	172.31.64.6	Ubuntu 22.04
v1-W19-172.31.64.13	xNIC	1	1.6M	1.0K	0.0	0.0	172.31.64.13	Windows Serv...
v2-R88-10-0-1-64	xNIC	2	1.5M	1.0K	0.0	0.0	10.0.1.164	Rocky Linux 8.8
v2-U22-10-0-1-251	xNIC	2	3.1M	2.0K	0.0	0.0	10.0.1.251	Ubuntu 22.04
v2-W19-10-0-1-183	xNIC	1	2.0M	1.3K	0.0	0.0	10.0.1.183	Windows Serv...

Interfaces v1-cs-172-31-64-14

Name	Index	Ip	SubnetPrefix	Public Ip	MTU	Driver	PciAddress
dpdk-0	0	172.31.62.204	172.31.48.0/20	--	--	igb_uio	0000:00:06.0
ens5	2	172.31.64.14	172.31.64.0/28	--	9001	ena	0000:00:05.0

Components Streams StreamLinks Adaptors HA Lossless Settings PTP Subscriptions Notifications Configurations Setup

Components

Administrator: Windows PowerShell

Information - dev

ip-172-41-128-25

dev.fbd4bf(AWS-drs2-ore-med-swxtch2-swxtch-all)

Licensing

License File

Cloud

AccountID

Region

SwXtchId

Status

AWS

63972066639

us-west-2

i-0005b067da040f7e4

OK

Bandwidth

1.7M bps/50.06 bps

Clients

7/30

Bridges

0/1

License expiration

2033-Nov-08

Filter

Components

Name	Class	Type	Rx (bps)	Rx (pps)	Tx (bps)	Tx (pps)	Primary IP	OS
drs2-ore-med-swxtch	swXtch	X1	1.8M	159	10.7M	955	172.41.128.25	Ubuntu 20.04
drs3-ore-med-swxtch	swXtch	X1	1.8M	160	10.8M	964	172.52.137.127	Ubuntu 20.04
drs2-ore-Ubuntu_2...	xNIC	1	0	0	3.6M	321	172.41.129.42	Ubuntu 22.04
drs2-ore-Win2019...	xNIC	2	3.9M	336	0	0	172.41.128.113	Windows Serv...
drs2-ore-Win2019...	xNIC	1	3.6M	322	1.6K	0	172.41.131.224	Windows Serv...
drs2-ore-agent2_T...	xNIC	2	3.7M	327	0	0	172.41.128.144	Ubuntu 20.04
drs2-ore-rocky1_H...	xNIC	2	3.7M	330	0	0	172.41.130.80	Rocky Linux 9.4
drs3-ore-agent1	xNIC	2	3.8M	338	0	0	172.52.132.19	Ubuntu 20.04
drs3-ore-rocky1_2...	xNIC	2	3.8M	335	0	0	172.52.139.237	Rocky Linux 9.4

Interfaces drs2-ore-Ubuntu_22_04_2data_nic

Name	Index	Ip	SubnetPrefix	Public Ip	MTU	Driver	PciAddress
ens5	2	172.41.129.42	172.41.128.0/22	--	9001	ena	0000:00:05.0
ens6	3	172.41.21.187	172.41.21.0/24	--	9001	ena	0000:00:06.0
ens7	4	172.52.152.190	172.52.144.0/20	--	9001	ena	0000:00:07.0
swxtch-tun0	5	172.30.1.42	172.30.0.0/22	--	4096	--	--

The Components view gives a detailed list of all the connected nodes in a cloudSwXtch's network. To navigate to this view, press CTRL+E or click on the view's name. To switch between components, use the up and down directional keys on your keyboard.

This view includes:

- **Name** - The name of the component
- **Class** - Either cloudSwXtch, xNIC, or VM (non-xNIC)
- **Type** - X1 or X2 for cloudSwXtch or Type 1 or 2 for xNIC
- **RX bps** - The total ingress bits per second that the component is receiving.
- **RX pps** - The total ingress packets per second that the component is receiving.
- **TX bps** - The total egress bits per second that the component is transmitting.
- **TX pps** - The total egress packets per second that the xNIC is transmitting.
- **Primary IP** - The primary interface IP address
- **OS**: Operating System

Additional options under Setup:

- **Version**
- **Hostname**
- **Cloud**
- **Region**
- **Instance**

An Interface panel displays when in the Component view, detailing the interfaces of a selected component. This panel includes the following information for each interface:

- **Name**
- **Index**
- **IP Address**
- **SubnetPrefix**
- **Public IP**
- **MTU**
- **Driver**
- **PciAddress**

Streams

Streams		Components	Stream	Table	Stats		
			Rx	Pkts	Rx	Bits	Tx Pkts
							Tx Bits
(-) 224.2.2.2:8400							
(-) drs2-ore-Ubuntu_22.04_2data_nic		xNIC	0	0	476	5.4M	
(-) drs3-ore-med-swxtch		swXtch	225	2.5M	903	10.2M	
drs2-ore-Win2019_1Multiviewer		xNIC	225	2.5M	0	0	
drs2-ore-Win2019_2data_nic_2		xNIC	226	2.5M	0	0	
drs3-ore-agent1		xNIC	226	2.5M	0	0	
drs3-ore-rocky1_2data_nic		xNIC	226	2.5M	0	0	
(-) drs_ore-med_swxtch		swXtch	251	2.8M	1.5K	16.9M	
drs2-ore-Win2019_1Multiviewer		xNIC	252	2.8M	0	0	
drs2-ore-Win2019_2data_nic_2		xNIC	251	2.8M	0	0	
drs2-ore-agent2_TL_2160p50_216...		xNIC	251	2.8M	0	0	
drs2-ore-rocky1_HD_playout		xNIC	251	2.8M	0	0	
drs3-ore-agent1		xNIC	251	2.8M	0	0	
drs3-ore-rocky1_2data_nic		xNIC	251	2.8M	0	0	
(+) 224.2.2.2:8401							
(-) HD_59to60_235.0.0.2							
(-) drs2-ore-agent2_TL_2160p50_2160p60		xNIC	0	0	994	1.5M	
(-) drs_ore-med_swxtch		swXtch	994	1.5M	994	1.5M	
drs2-ore-Ubuntu_22.04_2data_nic		xNIC	994	1.5M	0	0	

This tree view shows all the multicast groups that are being received and/or transmitted by the cloudSwXtch and how it interacts with the various endpoints. To navigate to this view, enter CTRL + S or click on the view's name.

This view includes:

- **Stream** - The name of the stream IP and Port. This could also be an Alias set in wXcked Eye. For example, BaseballCamera1 and BaseballCamera2 were Aliases assigned in wXcked Eye.
- **Components**: swxtch, xNIC or VM (Virtual Machine w/o xNIC)
- **RX Pkts**: The total ingress packets per second being received by the component.
- **RX Bits**: The total ingress bits per second that is received by the component.
- **TX Pkts**: The total egress packets per second that is transmitted by the component.
- **TX Bits**: The total egress bits per second that is transmitted by the component.

In the example above, stream 239.1.1.1: 3490 is listed. It is being transmitted by an endpoint (DSd-agent-201) to the cloudSwXtch (dsd-core-200), which is then send it to the other endpoints. That is why DSd-agent-202, DSd-agent-204 and DSd-agent-205 are receiving packets.

Stream Links

Components	Direction	Protocol	Bits (bps)	Pkts (pps)	Frag-Pkts (fpps)	Frgs/Pkts
(-) ds3-ore-med-swxtch						
(-) 224.2.2.2:8481						
ds22-ore-Ubuntu_22.04_2data_nic (...)	Ingress	xMC	216	0	--	--
ds22-ore-Win2019_1Multiviewer (xNIC)	Egress	xMC	216	0	--	--
ds22-ore-Win2019_2data_nic_2 (xNIC)	Egress	xMC	216	0	--	--
ds22-ore-agent2_TL_2160p50_2160p6...	Egress	xMC	216	0	--	--
ds22-ore-rocky1_HD_plyout (xNIC)	Egress	xMC	216	0	--	--
ds22-ore-agent1 (xNIC)	Egress	xMC	216	0	--	--
ds22-ore-rocky1_2data_nic (xNIC)	Egress	xMC	216	0	--	--
(-) 224.2.2.3:8480						
ds22-ore-Ubuntu_22.04_2data_nic (...)	Ingress	xMC	1.7M	151	--	--
ds22-ore-Win2019_1Multiviewer (xNIC)	Egress	xMC	1.7M	151	--	--
ds22-ore-Win2019_2data_nic_2 (xNIC)	Egress	xMC	1.7M	151	--	--
ds22-ore-agent2_TL_2160p50_2160p6...	Egress	xMC	1.7M	151	--	--
ds22-ore-rocky1_HD_plyout (xNIC)	Egress	xMC	1.7M	151	--	--
ds22-ore-agent1 (xNIC)	Egress	xMC	1.7M	151	--	--
ds22-ore-rocky1_2data_nic (xNIC)	Egress	xMC	1.7M	151	--	--
(-) 239.255.255.250:1900						
ds22-ore-Win2019_2data_nic_2 (xNIC)	Ingress	xMC	0	0	--	--
(-) ds33-ore-med-swxtch						
(-) 224.2.2.2:8481						
ds22-ore-Ubuntu_22.04_2data_nic (...)	Ingress	xMC	216	0	--	--
ds22-ore-Win2019_1Multiviewer (xNIC)	Egress	xMC	272	0	--	--
ds22-ore-Win2019_2data_nic_2 (xNIC)	Egress	xMC	216	0	--	--
ds22-ore-agent2_TL_2160p50_2160p6...	Egress	xMC	208	0	--	--
ds22-ore-rocky1_HD_plyout (xNIC)	Egress	xMC	216	0	--	--
ds22-ore-agent1 (xNIC)	Egress	xMC	272	0	--	--
ds22-ore-rocky1_2data_nic (xNIC)	Egress	xMC	272	0	--	--
(-) 224.2.2.4:8480						
ds22-ore-Ubuntu_22.04_2data_nic (...)	Ingress	xMC	1.7M	149	--	--
ds22-ore-Win2019_1Multiviewer (xNIC)	Egress	xMC	1.7M	149	--	--
ds22-ore-Win2019_2data_nic_2 (xNIC)	Egress	xMC	1.7M	149	--	--
ds22-ore-agent2_TL_2160p50_2160p6...	Egress	xMC	1.7M	149	--	--
ds22-ore-rocky1_HD_plyout (xNIC)	Egress	xMC	1.7M	149	--	--
ds22-ore-agent1 (xNIC)	Egress	xMC	1.7M	149	--	--
ds22-ore-rocky1_2data_nic (xNIC)	Egress	xMC	1.7M	149	--	--
(-) 239.255.255.250:1900						
ds22-ore-Win2019_2data_nic_2 (xNIC)	Ingress	xMC	0	0	--	--

The Stream Links view displays all the cloudSwxtch network components (cloudSwxtch or xNIC) and the streams they are linked to. To select this view, press CTRL + K or click on the view's name. To expand a component, simply use the up and down keys on the keyboard and hit ENTER to select.

In this example, the cloudSwxtch (dsd-core-200) is selected with the singular stream 239.1.1.1:3490 expanded. Here, a user can see a list of endpoints receiving/transmitting the stream to/from the cloudSwxtch, dsd-core-200.

- **Components:** cloudSwxtch or endpoints (xNICs)
- **Direction:** Either Ingress or Egress, depending on whether data is being ingested or transmitted by the component
- **Protocol:** Multicast or Broadcast
- **Bits (bps):** Bits per second
- **Pkts (pps):** Packets per second
- **Frag-Pkts (fpps):** Fragmented packets per second
- **Frgs/Pkts:** Total # of Fragmented Packets

Adaptors

Components	Active	Direction	Stream	Node	Listener Port
(-) dsukiennik-top-n-0-core-000(swXtch)					
srt-listener	false	ingress	229.1.2.3:5000	--	5000
udp	false	egress	229.1.2.3:3000	172.0.0.1:7600	--
udp	false	ingress	229.1.2.3:4000	--	4000
(-) dsukiennik-top-n-0-agent-000(Bridge)					
udp	false	ingress	228.1.2.3:5000	--	4000

The Adaptors view displays information regarding Protocol Conversion and Fanout. It includes a detailed list of configured protocols (UDP, SRT Caller/Listener, or RIST Caller/Listener), their direction (ingress/egress), the stream IP or stream name, the node, and Listener Port.

To navigate to this view, enter CTRL + A or click on the view's name.

HA (High Availability)

Components (xNIC)	Path	Ingress (pps)	Path	Ingress (bps)	HA Path Usage (%)	Missing Packets
(-) ds22-ore-Ubuntu_22.04_2data_nic						
Summary						
Reconstructed Path		0		0	-	0
ds22		0		0	0.00	0
ds22		0		0	0.00	0
(-) 172.41.21.255:138						
Reconstructed Path		0		0	-	0
ds22		0		0	0.00	0
ds22		0		0	0.00	0
(-) 172.41.131.255:138						
Reconstructed Path		0		0	-	0
ds22		0		0	0.00	0
ds22		0		0	0.00	0
(-) ds22-ore-Win2019_1Multiviewer						
Summary						
Reconstructed Path		192		2.2M	-	0
ds22		192		2.2M	50.52	0
ds22		192		2.2M	48.96	0
(-) 224.2.2.2:8480						
Reconstructed Path		192		2.2M	-	0
ds22		192		2.2M	50.52	0
ds22		192		2.2M	48.96	0
(-) ds22-ore-Win2019_2data_nic_2						
Summary						
Reconstructed Path		195		2.1M	-	0
ds22		195		2.1M	37.44	0
ds22		195		2.1M	62.56	0
Cluster						
Name	Swxtch/es					
(-) ds22-drs3HA						
ds22	172.41.128.25:80					
ds22	172.52.137.127:80					

The HA view shows additional details for high availability as the perspective of the consumer. It will only show data if High Availability has been configured.

To navigate to this view, enter CTRL + H or click on the view's name. Go to [High Availability](#) for configuration details.

This view is organized into two sections: **Component Consumer** and **packet/bits statistics**.

- Component Consumer:** This is the xNIC component in the cloudSwXtch network that is consuming or producing the HA paths. Expanding the consumer details will show the stream IP split into two paths (red and blue) and a reconstructed path.

The statistics by each line item is further explained below:

- Path Ingress pps** - The total ingress packets per second that is received in the path for the multicast group.
- Path Ingress bps** - The total ingress bits per second that is received in the path for the multicast group.
- Path Usage %** - The percentage that the path is used in the highly available multicast group.
- Missing packets** - The total number of missing packets for the path since the inception of the stream. If you stop the stream or cloudSwXtches, the number will stop increasing but will not reset.

Lossless

ip-18-64-24-45

dev.7f1520(BVOL)

Licensing

License File

Cloud

AMS

Bandwidth

0 bps/100.00 bps

AccountId

639720666639

Clients

1/1000 | Bridges 1/10

Region

us-east-1

License expiration

2026-Sep-14

SwitchId

i-97371d5d12cb4275

Status

ON

NEW NOTIFICATIONS

Information - dev.7f1520

SLP Rx

Components

Orig Pkts

Retr Pkts

Retr Reqs

Dup Pkts

Late Pkts

Avg Wait(ns)

Acks Sent

Acks2 Recv

SLP Tx

Components

Sent Pkts

Nacks Recv

Retr Pkts

RTT(ns)

RTTVar(ns)

RxC F Drops

TxC F Drops

Acks2 Sent

dev1-13

SLP TX

44.217.30.114

7.0G

44.7M

70.7M

17.3M

158.4K

0

0

90.9M

Aliases			Aliases		
Alias	Primary IP	Hostname	Class	Stream	
drs ore-med swtch	172.41.128.25	ip-172-41-128-25	swtch	--	
HD_50to60_235.0.0.2	--	--	--	235.0.0.2:5000	
HD_east_ad_insert_235.0.0.4	--	--	--	235.0.0.4:5000	
HD_gfx_branding_235.0.0.3	--	--	--	235.0.0.3:5000	
HD_west_ad_insert_235.0.0.5	--	--	--	235.0.0.5:5000	
JPEG_XS_UHD_225.0.0.1	--	--	--	225.0.0.1:5000	
Light_Blue	--	--	--	239.2.1.3:5000	
Orange_Stream	--	--	--	236.1.2.3:5000	
Stream1	--	--	--	235.4.6.4:5000	
UHD_50to60_225.0.0.1	--	--	--	225.0.0.2:5000	
UHD_east_ad_insert_225.0.0.4	--	--	--	225.0.0.4:5000	
UHD_gfx_branding_225.0.0.3	--	--	--	225.0.0.3:5000	
UHD_west_ad_insert_225.0.0.5	--	--	--	225.0.0.5:5000	
drs3-ore-med-swtch	172.52.137.127	ip-172-52-137-127	swtch	--	
drs2-ore-Ubuntu22.04_2data_nic	172.41.129.42	ip-172-41-129-42	xNIC	--	
drs2-ore-Win2019_1Multiviewer	172.41.128.113	EC2AMAZ-R5RC5EU	xNIC	--	
drs2-ore-Win2019_2data_nic_2	172.41.131.224	EC2AMAZ-DGH714G	xNIC	--	

Configurations		Value
Section	(-) HS	
	swtches	drs2: {172.41.128.25:80}, drs3: {172.52.137.127:80}
	entitlements	
	BandwidthMbps	50.00 bps
	ClockSync	true
	Fanout	true
	HS	true
	MajorVersionUpdate	true
	Max Bridges	1
	Max Clients	30
	Mesh	true
	TachyonLive	false
	wXckedEye	true
	(-) meshConfig	
	swtches	--

The Settings view is organized into two parts: **Aliases** and **Configurations**. The Aliases section provides users with a detailed list of user-assigned names for components, sockets, and streams. The Configurations section presents an expanded look at the licensing details found in the Information panel. In addition, they can see what cloudSwXtches are connected in their mesh and HA configurations.

To navigate to this view, enter CTRL + G or click on the view's name.

- **Entitlements:** Depending on their license, users will have a set number for their Max Bandwidth, Max Clients and Max bridges. In the example above, the user has a max bandwidth of 50 GBs with 30 clients max and 1 bridges max. This section will also show if a user has the following features enabled: Mesh, HS, Fanout, Clock Sync (PTP), wXckedEye, TachyonLive, and Major Version Update.
- **meshConfig:** This will list the IP addresses of the cloudSwXtches connected to a mesh.
- **HS:** This will list the Paths created for High Availability with each path showing the IP addresses of connected cloudSwXtches.

PTP

Timing Nodes				
Master Node				
=====				
	Name	dsd-core-200	Time Sync Service	phc:/dev/ptp_hyperv
Follower Nodes				
=====				
	Name	Status	Local Offset	Root Offset
	D5d-agent-101	Not Present	--	--
	D5d-agent-102	Not Present	--	--
	D5d-agent-103	Not Present	--	--
	D5d-agent-104	Not Present	--	--
	D5d-agent-105	Not Present	--	--
	D5d-agent-106	Not Present	--	--
	D5d-agent-107	Not Present	--	--
	D5d-agent-201	Present	2.89 μ s	23.12 μ s
	D5d-agent-202	Present	3.54 μ s	27.82 μ s
	D5d-agent-204	Present	3.34 μ s	16.96 μ s
	D5d-agent-205	Present	5.55 μ s	14.28 μ s
	Dsd-Windows-10E	Not Present	--	--
	aks-nodepool11-23164585-vmss00001I	Not Present	--	--
	aks-nodepool11-23164585-vmss00001J	Not Present	--	--

The PTP view displays information regarding the clock sync configuration for the cloudSwXtch. The page in swtch-top will only populate with information if the user has the PTP feature enabled.

To navigate to this view, enter CTRL + P or click on the view's name.

In the example above, the cloudSwXtch (dsd-core-200) is acting as the Master Node.

- **Master Node** - The Master Node is what the PTP configuration sets as the most reliable time source. This will send the true time it receives from the source clock to the Follower Nodes.
 - **Name** - The name of the cloudSwXtch
 - **Time Sync Service** - The source clock
- **Follower Nodes** - The Follower Nodes lists the agents/VMs that subscribe to the Master Node for accurate timing.
 - **Name** - The name of the endpoints
 - **Status** - The status of the endpoints, noting if the node is active in the PTP configuration
 - **Local Offset** - The local offset denotes the offset in time from the cloudSwXtch to the xNIC.
 - **Root Offset** - The root offset denotes the offset in time from the GrandMaster clock to the cloudSwXtch and its follower nodes (xNIC). Note how the root is larger than the local. This is normal behavior since the distance between the follower node and the Grandmaster clock is greater than the offset between a cloudSwXtch and xNIC.

PTP Stabilization

After upgrading your cloudSwXtch system, you may notice that the local and root offset values are much larger than they actually are. It can take up to 30 minutes for the values to stabilize and return back to normal levels.

Subscriptions

Component		Subscriptions		Sources
(-) drs2-ore-Win2019_1_TEST				
(-) Ethernet 4				
224.0.0.251	exclude			--
224.0.0.252	exclude			--
224.2.2.2	include		172.30.1.42-172.41.3.4-172.52.158.227	
224.2.2.3	include			172.30.1.42
(-) drs2-ore-Win2019_2data_nic_2				
(-) Ethernet 4				
224.0.0.251	exclude			--
224.0.0.252	exclude			--
224.2.2.2	include		172.30.1.42-172.41.3.4-172.52.158.227	
224.2.2.3	include			172.30.1.42
(-) drs2-ore-agent2_TL_2160p50_2160p60				
(-) ens6				
224.0.0.251	exclude			--
224.2.2.2	include			172.30.1.42
224.2.2.3	include			172.30.1.42
(-) drs2-ore-rocky1_HD_playout				
(-) eth1				
224.2.2.2	include			172.30.1.42
224.2.2.3	include			172.30.1.42
(-) drs3-ore-agent1				
(-) ens6				
224.0.0.251	exclude			--
224.2.2.2	include			172.30.1.42
224.2.2.3	include			172.30.1.42
(-) drs3-ore-rocky1_2data_nic				
(-) eth2				
224.2.2.2	include			172.30.1.42
224.2.2.3	include			172.30.1.42

The Subscriptions page details the different source list configurations for single source multicast and multiple multicast groups. Here, users will get a detailed lists of included and excluded streams and the source they originate from.

To navigate to this view, enter CTRL + U or click on the view's name.

Notifications

Filter

							Notifications

The Notifications view compiles a list of cloudSwXtch-related notifications.

To navigate to this view, enter CTRL + N or click on the view's name.

A New Notifications alert will appear in the Information panel in swXtch-top as shown below. Visiting the Notifications view will remove this alert.

ip-172-41-128-25		Information - dev		NEW NOTIFICATIONS
Cloud	AWS	Licensing	License File	
AccountId	639720666639	Bandwidth	1.7M bps/50.86 bps	
Region	us-west-2	Clients	7/30 Bridges 0/1	
SwXtchId	i-0005b067da040f7e4	License expiration	2033-Nov-08	
Status	OK			

Configurations

Components		Configurations	Value
(-) drs2-ore-Ubuntu 22.04_2data_nic			
(-) drs2-ore-Win2019_1_TEST			
Primary Swxtch Address			http://172.41.128.25:80
Xnic Type			2
Data Interface			Ethernet 4
Stream Specs			
Mmc Producer Enable			false
Multiple Multicast Groups			
224.2.2.2:8400			
Override Source Ip			172.41.21.40
Path Streams			
[0]			
Ssm			
Filter			include
Src List			[0]
Stream			172.30.1.42
[1]			224.2.2.3:8400
Ssm			
Src List			172.30.1.42
Filter			include
Stream			224.2.2.4:8400
Xnic Rpc Port			10002
Stats Report Wait			60
Data Plane Specs			
Bpf Programs			--
Verbosity			0
Virtual Interface			
Subnet			255.255.252.0
Type			tun
Ip			172.30.0.113
Mtu			4096
Name			swxtch-tun0
Swxtch			172.41.128.25
Control Interface			Ethernet 3
Ha			

The Configurations view gives users a visualization of the xNICs' configuration JSON files.

To navigate to this view, enter CTRL + F or click on the view's name.

Troubleshooting swtch-top

1. If the swtch-top "Status" is showing that there is a "Connection error:"

Check that the cloudSwXtch is started.

Check that you entered in the proper cloudswtch name or IP when running the swtch-top command.

If name does not work when running the swtch-top command then the DNS is not set-up correctly, use the IP address instead.
2. If an xNIC was installed but is not showing up in swtch-top:

Navigate to the swtch-nic.conf file and validate that the "SwxtchSvcAddr" is correct.

Windows can be found at "C:\Program Files\SwXtch.io\Swxtch-xNIC"

Linux can be found at "/var/opt/swtch/swxtch-xnic.conf"

Check that the firewall is open for the following ports:

subnet	protocol	ports	vm
ctrl-subnet	tcp	80	cloudSwXtch
ctrl-subnet	udp	10800-10803	all
data-subnet	udp	9999	all

3. If a multicast group is not showing up then check that they have registered.

In Linux, run this command:

Text

Plaintext	Copy
ip maddress show	

- In Windows, run this command in PowerShell:

Text

Plaintext	Copy
netsh.exe interface ipv4 show joins	

- If the joins are not showing here then the application is not joining the multi-cast group. In this case run swtch-perf for the same IP:Port combination and then re-try in the program.

If the joins are not showing here then the application is not joining the multi-cast group. In this case run swtch-perf for the same IP:Port combination and then re-try in the program.

If using Windows make use of Task Manager and view Performance to know where data is being sent/received.

Validate using TCPdump or Wireshark to identify where traffic is going as it could be going to the wrong network interface, it should be going to the Data Interface if xNIC2 and Swtch-tun0 if xNIC1. An example is below:

Plaintext	Copy
\$ sudo tcpdump udp -X -i <interface>	

NOTE

xNIC1 interface: swxtch-tun0
xNIC2 interface: data nic (usually eth1 for Linux, and "Ethernet 2" for Windows)

- Validate that a firewall is not stopping the multicast and open up the firewall to include port exceptions.

swtch-top on a cloudSwXtch

swtch-top should be run from a virtual machine with an xNIC installed, it should be avoided to run it or anything else directly on a cloudSwXtch. That being said it can be done, but you must run it with sudo. Only run it on the cloudSwXtch if doing advanced troubleshooting.

sudo /swtch/swtch-top dashboard --swtch localhost

Alternatively use 127.0.0.1 or swtch-hostname or swtch-IP in place of localhost

swxtch-tcpdump

WHAT TO EXPECT

Users can use a cloudSwxtch specific version of tcpdump called swxtch-tcpdump. This tool helps with capturing multicast packets sent to and from the cloudSwxtch. It is the same as tcpdump but with logic to decode our own header and display the original MC payload.

In this article, users will learn about the available arguments for swxtch-tcpdump.

Using swxtch-tcpdump

Execute the following command:

Bash	Copy
swxtch-tcpdump	

Note: The default is **swxtch-tun** (Windows) or **swxtch-tun0** (Linux). If their multicast is running on a different interface, then a user will need to specify that interface. To get a list of interfaces for Windows, you can use **ip config**. For Linux, you can use **ip a**. After you get the name of the correct interface, you can use the **-i** argument followed by your desired interface name.

Example:

Bash	Copy
swxtch-tcpdump -i ens6	

Additional arguments

Users can use the **-h** argument as shown below to get a list of available arguments for swxtch-tcpdump.

Bash	Copy
<pre>ubuntu@ip-172-41-128-232:/var/opt\$ swxtch-tcpdump -h swxtch-tcpdump version 5.0.0-PRE-GIT libpcap version 1.9.1 (with TPACKET_V3) OpenSSL 1.1.1f 31 Mar 2020 Usage: swxtch-tcpdump [-AbDefhHIJKlLnNOpqStuUvxX#] [-B size] [-c count] [--count] [-C file_size] [-E algo:secret] [-F file] [-G seconds] [-i interface] [--immediate-mode] [-j tstamptype] [-M secret] [--number] [--print] [--print-sampling nth] [-Q in out inout] [-r file] [-s snaplen] [-T type] [--version] [-V file] [-w file] [-W filecount] [-y datalinktype] [--time-stamp-precision precision] [--micro] [--nano] [-z postrotate-command] [-Z user] [expression]</pre>	

swxtch-where

WHAT TO EXPECT

swxtch-where allows users to call for hardware information regarding their cloudSwXtch VM.

In this article, users will learn about the different arguments they can use with swxtch-where and example outputs they should expect.

swXtch-where Cloud Type

Below are the Linux and Windows commands to call swxtch-where. An empty command (without an argument) like the examples below will only return the cloud type.

Windows

For Windows VM, swxtch-where must be called from the xNIC directory and have the .exe extension.

Bash	Copy
PS C:\Program Files\SwXtch.io\Swxtch-xNIC2> .\swxtch-where.exe	

Linux

For Linux VM, a user would only need to input the following:

Bash	Copy
swxtch-where	

swxtch-where Format

What is probably the most useful option in the swxtch-where argument list is `-f json` or `--format json`, which provides users with a json of hardware-related information regarding the cloudSwXtch VM. This information is similar to the Hardware view in swxtch-top and Hardware panel in wXcked Eye's Settings' General tab. It presents a breakdown of the control and data subnet with information categorized as either metadata or operating system.

Windows

Bash	Copy
PS C:\Program Files\SwXtch.io\Swxtch-xNIC2> .\swxtch-where.exe --format json	

Linux

Bash	Copy
swxtch-where -f json	

Example Output in Windows:

Bash	Copy
<pre>S C:\Program Files\SwXtch.io\Swxtch-xNIC2> .\swxtch-where.exe --format json { "cloudType": "AZURE", "nics": [{ "computed": { "isPreferredControlNic": false, "isPreferredDataNic": true, "isSRIOV": false }, "meta": { "ipAddress": "10.5.1.7", "ipBroadcast": "10.5.1.255", "ipSubnet": "10.5.1.0", "mac": "00:22:48:27:0e:dc", "subnetMask": "255.255.255.0" }, "os": { "driver": "mlx5.sys", "mac": "00:22:48:27:0e:dc", "mtu": 1500, "name": "Ethernet 261", "pciAddress": "65025:00:02.0" } }, { "computed": { "isPreferredControlNic": true, "isPreferredDataNic": false, "isSRIOV": false }, "meta": { "ipAddress": "10.5.2.7", "ipBroadcast": "10.5.2.255", "ipSubnet": "10.5.2.0", "mac": "00:22:48:27:07:14", "subnetMask": "255.255.255.0" }, "os": { "driver": "netvsc.sys", "ipAddress": "10.5.2.7", "ipBroadcast": "10.5.2.255", "ipSubnet": "10.5.2.0", "mac": "00:22:48:27:07:14", "mtu": 1500, "name": "Ethernet", "pciAddress": "", "subnetMask": "255.255.255.0" } }] }</pre>	

swxtch-where Version

Using the `-v` or `--version` argument after the `swxtch-where` command will return the version.

Example in Windows:

Bash	Copy
<pre>PS C:\Program Files\SwXtch.io\Swxtch-xNIC2> .\swxtch-where.exe -v 1.0</pre>	

swxtch-where Help

The `swxtch-where -h` or `--help` argument provides users with a detailed list of available arguments.

Example in Windows:

```
PS C:\Program Files\SwXtch.io\Swxtch-xNIC2> .\swxtch-where.exe - h
Usage: C:\Program Files\SwXtch.io\Swxtch-xNIC2\swxtch-where.exe [options]
```

swxtch-where utility.
Calling with no arguments simply returns the cloud type.

Optional arguments:

```
-h --help      shows help message and exits [default: false]
-v --version   prints version information and exits [default: false]
-f --format    output format (example: "json")
```

Universal Third-Party Testing Tools

WHAT TO EXPECT

While xNIC installation provides you with a number of useful tools to test the functionality of your cloudSwXtch network (swtch-perf, swtch-tcpdump, etc.), there is also a wealth of universal third-party testing tools at your disposal.

In this article, we will take a deeper dive into these alternative options and understand their benefits and pitfalls.

- [VLC](#)
- [ffmpeg/ffplay](#)
- [iPerf \(v2\)](#)
- [sockperf](#)
- [Python/Go](#)

Please Note: These tools should be used for testing purposes only.

VLC

VLC is a free and open-source cross-platform multimedia player and framework that plays most multimedia files, and various streaming protocols. As a highly visual tool, it can be used to demonstrate the delivery and fidelity of video streams from the cloudSwXtch to the xNIC.

Example

In the following example, a producer is streaming a .ts file using the RTP protocol to the multicast address 239.1.1.2 to port 1000 in a indefinite loop. The agent is consuming the stream.

Producer:

PowerShell	Copy
<pre>vlc file:///home/ubuntu/surfing.ts --sout '#rtp{dst=239.1.1.2,port=10000,mux=ts}' --loop</pre>	

Consumer:

PowerShell	Copy
<pre>vlc rtp://@239.1.1.2:10000</pre>	

vlc vs. cvlc

When using Linux, you can use cvlc instead of vlc. cvlc is an alias for "vlc -I dummy". It is VLC without the user interface, lighter and faster to open remotely, and will show a window playing the video only. **Note:** cvlc does not work for Windows.

ffmpeg/ffplay

ffmpeg/ffply are command line tools to demonstrate the delivery of a stream. **ffmpeg** is the streamer, **ffplay** is the command to see the stream. While VLC requires a file to stream from the producer to the consumer, ffmpeg/ffplay can produce video test patterns. In the following example, we will use "testsrc", which will generate a test video pattern. This will display a color pattern with a scrolling gradient and a timestamp. If a stream appears, it means this was a successful delivery. **Please note:** Other test patterns are available for use.

Example

In this example, the user is testing a stream signal with a resolution of 640x480 at 30fps, in MPEG using labfi (Libavfilter input virtual device) for 1000 seconds to the multicast address 239.1.1.1, port 10000 using a packet size of 1316 bytes.

Producer

PowerShell	Copy
<pre>ffmpeg -hide_banner -f lavfi -re -i testsrc=duration=1000:size=640x480:rate=30 -f mpegts "udp://239.1.1.1:10000?pkt_size=1316"</pre>	

Below is an example of **ffmpeg producer output**.

```
ubuntu@FS2A1AD22:~$ ffmpeg -hide_banner -f lavfi -re -i testsrc=duration=1000:size=640x480:rate=30 -f mpegts "udp://239.1.1.1:11000?pkt_size=1316"
Input #0, lavfi, from 'testsrc=duration=1000:size=640x480:rate=30':
  Duration: N/A, start: 0.000000, bitrate: N/A
  Stream #0:0: Video: rawvideo (RGB[24] / 0x18424752), rgb24, 640x480 [SAR 1:1 DAR 4:3], 30 tbr, 30 tbn, 30 tbc
Stream mapping:
  Stream #0:0 -> #0:0 (rawvideo (native) -> mpeg2video (native))
Press [q] to stop, [?] for help
Output #0, mpegts, to 'udp://239.1.1.1:11000?pkt_size=1316':
  Metadata:
    encoder         : Lavf58.76.100
  Stream #0:0: Video: mpeg2video (Main), yuv420p(tv, progressive), 640x480 [SAR 1:1 DAR 4:3], q=2-31, 200 kb/s, 30 fps, 90k tbn
  Metadata:
    encoder         : Lavc58.134.100 mpeg2video
  Side data:
    cpb: bitrate max/min/avg: 0/0/200000 buffer size: 0 vbv_delay: N/A
    frame= 517 fps= 30 q=31.0 size=      909KB time=00:00:17.13 bitrate= 434.8kbits/s speed=0.996x
```

Consumer:

PowerShell	Copy
ffplay -hide_banner "udp://239.1.1.1:11000"	

Below is an example ffplay consumer output.

```
ubuntu@FS2A2AD22:~$ ffplay -hide_banner "udp://239.1.1.1:11000"
[mpeg2video @ 0x7811f0b29e40] Invalid frame dimensions 0x0, f=0/0
Input #0, mpegts, from 'udp://239.1.1.1:11000':KB sq=    0B f=0/0
  Duration: N/A, start: 57.800000, bitrate: N/A
  Program 1
  Metadata:
    service_name     : Service01
    service_provider : ffmpeg
  Stream #0:0[0x100]: Video: mpeg2video (Main) ([2][0][0][0] / 0x0002), yuv420p(tv, progressive), 640x480 [SAR 1:1 DAR 4:3], 30 fps, 30 tbr, 90k tbn, 60 tbc
  Side data:
    cpb: bitrate max/min/avg: 0/0/0 buffer size: 49152 vbv_delay: N/A
    65.46 M-V: 0.866 fd= 179 aq=    0KB vq= 1039KB sq=    0B f=0/0
```

A successful delivery of the stream will display the following stream in the consumer's VM:



iPerf (v2)

Similar to switch-perf, iPerf is a multi-platform tool for network performance measurement and tuning. It is commonly used for bandwidth and latency measurements. However, what differs is that iPerf has additional arguments not found in switch-perf, our streamlined tool. Since iPerf (v3) does not support multicast, it is recommended to use iPerf (v2) for cloudSwitch networking testing.

Example

In this example, the user is creating a UDP stream at multicast address 239.1.1.3, port 1000 for 120 seconds with a buffer length of 1000 bytes. The enhanced report will display stats for every second with enhanced reporting.

Producer:

PowerShell	Copy
<pre>iperf -c 239.1.1.3 -p 10000 -u -t 120 -i 1 -e -l 1000</pre>	

• **Arguments Explained:**

- **-c:** the machine as the producer, specifying the multicast address
- **-p:** the port
- **-u:** the producer will be sending UDP packets
- **-t:** the duration of the stream in seconds
- **-i:** the time interval stats will be reported. In this case, it is every second (1).
- **-e:** enhanced reporting (interval, transfer, bandwidth, write/err, PPS)
- **-l:** length of buffers to read or write (1000)

This is an example of an iPerf producer output.

```
ubuntu@FS2A1AD22:~$ iperf -c 239.1.1.3 -p 10000 -u -t 120 -i 1 -e -l 1000
-----
Client connecting to 239.1.1.3, UDP port 10000 with pid 2535 (1 flows)
TOS set to 0x0 (Nagle on)
Sending 1000 byte datagrams, IPG target: 7629.39 us (kalman adjust)
UDP buffer size: 208 KByte (default)
-----
[ 1] local 172.41.128.64 port 53524 connected with 239.1.1.3 port 10000 (sock=3) on 2024-05-31 15:07:03 (UTC)
[ ID] Interval            Transfer          Bandwidth        Write/Err    PPS
[ 1] 0.0000-1.0000 sec    130 KBytes       1.06 Mbits/sec   0/0          133 pps
[ 1] 1.0000-2.0000 sec    128 KBytes       1.05 Mbits/sec   0/0          131 pps
[ 1] 2.0000-3.0000 sec    128 KBytes       1.05 Mbits/sec   0/0          131 pps
[ 1] 3.0000-4.0000 sec    128 KBytes       1.05 Mbits/sec   0/0          131 pps
[ 1] 4.0000-5.0000 sec    128 KBytes       1.05 Mbits/sec   0/0          131 pps
[ 1] 5.0000-6.0000 sec    128 KBytes       1.05 Mbits/sec   0/0          131 pps
[ 1] 6.0000-7.0000 sec    128 KBytes       1.05 Mbits/sec   0/0          131 pps
[ 1] 7.0000-8.0000 sec    128 KBytes       1.05 Mbits/sec   0/0          131 pps
[ 1] 8.0000-9.0000 sec    128 KBytes       1.05 Mbits/sec   0/0          131 pps
[ 1] 9.0000-10.0000 sec   128 KBytes       1.05 Mbits/sec   0/0          131 pps
[ 1] 10.0000-11.0000 sec  128 KBytes       1.05 Mbits/sec   0/0          131 pps
[ 1] 11.0000-12.0000 sec  128 KBytes       1.05 Mbits/sec   0/0          131 pps
[ 1] 12.0000-13.0000 sec  128 KBytes       1.05 Mbits/sec   0/0          131 pps
[ 1] 13.0000-14.0000 sec  128 KBytes       1.05 Mbits/sec   0/0          131 pps
[ 1] 14.0000-15.0000 sec  129 KBytes       1.06 Mbits/sec   0/0          131 pps
```

Consumer:

PowerShell	Copy
<pre>iperf -s -u -B 239.1.1.3 -p 10000 -i 1</pre>	

• **Arguments Explained:**

- **-s:** server, signifying a consumer
- **-u:** consumer will be taking in UDP packets
- **-B:** bind, specifying the multicast address (Note the capitalization.)
- **-p:** the port
- **-i:** the time interval stats will be reported (1)

This is an example of an iPerf consumer output. Note how each line item coincides with the single second interval.

```
ubuntu@FS2A2AD22:~$ ./iperf -s -u -B 239.1.1.3 -p 10000 -i 1
-----
Server listening on UDP port 10000
Joining multicast group 239.1.1.3
Server set to single client traffic mode (per multicast receive)
UDP buffer size: 208 KByte (default)
-----
[ 1] local 239.1.1.3 port 10000 connected with 172.41.128.64 port 56156
[ ID] Interval            Transfer          Bandwidth        Jitter    Lost/Total Datagrams
[ 1] 0.00-1.00 sec    129 KBytes       1.06 Mbits/sec   0.017 ms  7001/7133 (98%)
[ 1] 1.00-2.00 sec    128 KBytes       1.05 Mbits/sec   0.014 ms  0/131 (0%)
[ 1] 2.00-3.00 sec    128 KBytes       1.05 Mbits/sec   0.010 ms  0/131 (0%)
[ 1] 3.00-4.00 sec    128 KBytes       1.05 Mbits/sec   0.016 ms  0/131 (0%)
[ 1] 4.00-5.00 sec    128 KBytes       1.05 Mbits/sec   0.015 ms  0/131 (0%)
[ 1] 5.00-6.00 sec    129 KBytes       1.06 Mbits/sec   0.017 ms  0/132 (0%)
[ 1] 6.00-7.00 sec    128 KBytes       1.05 Mbits/sec   0.014 ms  0/131 (0%)
[ 1] 7.00-8.00 sec    128 KBytes       1.05 Mbits/sec   0.027 ms  0/131 (0%)
[ 1] 8.00-9.00 sec    128 KBytes       1.05 Mbits/sec   0.015 ms  0/131 (0%)
[ 1] 9.00-10.00 sec   128 KBytes       1.05 Mbits/sec   0.017 ms  0/131 (0%)
```

No Consumer Output

On some older versions of iPerf 2, the consumer will not show any output. This can happen even when configured to do so. However, you can still check `swtch-top` to see if the xNIC interface is getting traffic. To remedy this issue, ensure you have the latest version of iPerf 2.

sockperf

Another alternative to `swtch-perf` is **sockperf**, a Linux-only network benchmarking utility over socket API that is designed for testing the latency and throughput of high-performance systems. While it is similar to both `swtch-perf` and iPerf, it has far less commands than its counterparts. It is recommended to use sockperf when saturating the NIC and using the VM's maximum amount of bandwidth.

Example

In this example, the user will create a stream at multicast address 239.1.1.4, port 10000 using a message size of 1472, sending 2000 messages per second in 30 seconds.

Producer (sockperf throughput):

PowerShell	Copy
<pre>sockperf throughput --ip 239.1.1.4 --msg-size 1472 --port 10000 --mps 2000 --time 30</pre>	

Consumer (sockperf server):

PowerShell	Copy
<pre>sockperf server --ip 239.1.1.4 --Activity 800 --port 10000</pre>	

This will consume the traffic and print the activity for the last 800 messages processed. The consumer will continually wait for the producer until it is shut off.

Python/Go

Users may choose to also build their own tool to test their cloudSwXtch network. Below are two examples using Python or Go programming language to test multicast messaging. What is great about cloudSwXtch is that when we say it requires no code changes, the same applies for testing.

Python

As a pre-installed program on Linux-based machines, using Python is a simple way to test your xNIC.

Producer

The key object here is the socket module and the instruction used to send is sock.sendto. As you can see in the code, no modification to the code is necessary in order to make "compatible" with the xNIC/cloudSwXtch. The following code will work on any multicast-compatible environment.

1. Create a **producer.py** file on the producer VM. One method is to use nano. **Note:** You can name the file anything.

Plaintext	Copy
<pre>nano python.py</pre>	

2. Copy and paste the following script:

Profile	Copy
<pre>import socket import struct import time def send_multicast_message(ip, port, message, delay): # Create the socket multicast_group = (ip, port) sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM) # Set the time-to-live for messages to 1 so they do not go past the local network segment ttl = struct.pack('b', 1) sock.setsockopt(socket.IPPROTO_IP, socket.IP_MULTICAST_TTL, ttl) try: while True: # Send the message print(f'Sending "{message}" to {ip}:{port}') sock.sendto(message.encode('utf-8'), multicast_group) # Delay between messages time.sleep(delay) finally: print('Closing socket') sock.close() # Example usage send_multicast_message('239.1.1.1', 10000, 'Hello, Multicast!', 2)Save the file and close.</pre>	

3. Save and close.
4. Run the script:

Profile	Copy
<pre>python3 producer.py</pre>	

Below is an example of the output:

```
ubuntu@FS2A1AD22:~$ python3 producer.py
Sending "Hello, Multicast!" to 239.1.1.1:10000
Sending "Hello, Multicast!" to 239.1.1.1:10000
Sending "Hello, Multicast!" to 239.1.1.1:10000
█
```

Consumer

1. Create a **receiver.py** file on your consumer VM. One method is to use nano. **Note:** You can name the file anything.

Plaintext	Copy
nano consumer.py	

2. Copy and paste the following script:

Profile	Copy
<pre>import socket import struct def receive_multicast_message(ip, port): # Create the socket sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM) # Bind to the server address sock.bind(('', port)) # Tell the operating system to add the socket to the multicast group group = socket.inet_aton(ip) mreq = struct.pack('4sL', group, socket.INADDR_ANY) sock.setsockopt(socket.IPPROTO_IP, socket.IP_ADD_MEMBERSHIP, mreq) # Receive/respond loop while True: print('Waiting to receive') data, address = sock.recvfrom(1024) print(f'Received "{data.decode("utf-8")}" from {address}') # Example usage receive_multicast_message('239.1.1.1', 10000)Save and close.</pre>	

3. Save and exit.

4. Run the script:

PowerShell	Copy
python3 receiver.py	

Below is an example of the output on the consumer VM:

```
ubuntu@FS2A2AD22:~$ python3 receiver.py
Waiting to receive
Received "Hello, Multicast!" from ('172.41.128.64', 36601)
Waiting to receive
Received "Hello, Multicast!" from ('172.41.128.64', 36601)
Waiting to receive
Received "Hello, Multicast!" from ('172.41.128.64', 36601)
Waiting to receive
```

Go

Go is an alternative programming language. However, unlike Python, it must be installed manually. Use the following example scripts to test.

Producer

The producer uses the command **net.Dial** to send a message using multicast.

GoCopy

```
// producer.go
package main

import (
    "fmt"
    "net"
    "os"
    "time"
)

func main() {
    multicastAddr := "239.1.1.1:10000"

    conn, err := net.Dial("udp", multicastAddr)
    if err != nil {
        fmt.Println("Error connecting:", err)
        os.Exit(1)
    }
    defer conn.Close()

    for {
        message := "Hello, Multicast!"
        _, err = conn.Write([]byte(message))
        if err != nil {
            fmt.Println("Error sending message:", err)
            os.Exit(1)
        }
        fmt.Println("Message sent:", message)
        time.Sleep(2 * time.Second)
    }
}
```

Consumer

The receiver uses the command `net.ListenMulticastUDP`.

GoCopy

```
// receiver.go
package main

import (
    "fmt"
    "net"
    "os"
)

func main() {
    multicastAddr := "239.1.1.1:10000"
    addr, err := net.ResolveUDPAddr("udp", multicastAddr)
    if err != nil {
        fmt.Println("Error resolving address:", err)
        os.Exit(1)
    }

    conn, err := net.ListenMulticastUDP("udp", nil, addr)
    if err != nil {
        fmt.Println("Error listening for multicast:", err)
        os.Exit(1)
    }
    defer conn.Close()

    buf := make([]byte, 1024)
    for {
        n, src, err := conn.ReadFromUDP(buf)
        if err != nil {
            fmt.Println("Error receiving message:", err)
            os.Exit(1)
        }
        message := string(buf[:n])
        fmt.Printf("Received message from %v: %s\n", src, message)
    }
}
```

Other Programming Languages

As with Python or Go, you can create programs in virtually any language with the same logic for sending multicast traffic and let the xNIC + cloudSwXtch take care of the rest. For example, in Javascript, the commands will be `server.send` and `server.on`. In C++, the commands will be `sendto()` and `recvfrom()`.

How to View cloudSwXtch Logs for Troubleshooting

WHAT TO EXPECT

In this article, users will learn about accessing cloudSwXtch logs for the purpose of troubleshooting. We will break down two commands: **swx support** and **sudo journalctl**. Common arguments that can be used when compiling information for the support team at swXtch.io will also be discussed.

Support may also request for you to send xNIC logs. For more information, see [How to Find xNIC Logs](#).

swx support Logs

When troubleshooting, swXtch.io Support will request a report detailing the statistical data stored within the cloudSwXtch during a certain period. This report will include max highmarks, list highmarks, logs, swxtch info, xNIC Logs, License and Config -- all in a compressed tar.gz file.

Accessing the Report

The following command will compile the data stored on the disk between a certain time period by executing all other commands, saving it in a compressed file.

Bash	Copy
<pre>./swx support -f "yyyy-mm-dd" -t "yyyy-mm-dd" -s localhost</pre>	

An example output file would be **swxtch-report-2024-05-22_15-58-55.tar.gz**.

Alternatively, you can export swx support logs using the wXcked Eye UI. To learn more, see the [wXcked Eye Support Page](#) article.

Available swx support Arguments

Exporting only xNIC Logs (xnic-logs)

Users can export xNIC logs by specifying it with the swx support command. In addition to the xnic-logs command, they must use the argument --xnic and specify the xNIC's control IP. In the example below, the xNIC's control IP is 10.5.1.4.

Example:

Bash	Copy
<pre>./swx support xnic-logs -since 2h -s localhost --xnic 10.5.1.4</pre>	

An example output file would be **xnicLogs-10.5.1.4-2024-05-22_20-27-55.tar.gz**.

Date from (-f) and date to (-t)

- Following the -f (date_from), enter the starting date.
- Following the -t (date_to), enter the end date.
- Both dates can also include the time, using the ISO 8601 date format (Example: 2006-01-02T15:04:05Z)

Please note: Both "From" and "to" are optional. If nothing is set, the default is over the past 7 days.

Since hours/days (--since)

Following -since, users can specify number of days (d) or hours (h). For example, if a user would like to get logs since 8 hours ago, they would specify -since 8h. If they would like 4 days ago, they would specify -since 2d.

Example:

Bash	Copy
<div>Hours \$./swx support -s localhost --since 8h</div> <div>Days \$./swx support -s localhost --since 2d</div>	

Core dumps (--dump)

Add -dump if you wish to include the core dumps in the report.

Help (-h) for additional commands

Add a -h to the command if you wish to see an extensive list of available commands. For example, you can specify a certain section of the report to export as a tar.gz file, similar to the xNIC logs example above.

Example:

Bash	Copy
<pre>root@dtd-core-200:/swtch# ./swx support -h Swtch support tool Usage: swx support [flags] swx support [command] Available Commands: bridge-log Bridge log file list-highmarks List highmarks metrics within time range logs Logs max-highmarks Max highmarks metrics swtch-info Swtch Info xnic-logs xNIC logs file Flags: -b, --bridge string bridge address -f, --date-from string Date from -t, --date-to string Date to -d, --dump Dump enable -h, --help help for support -o, --path string Path location -s, --service-host-address string Host swtch address in the form <host>[:port] (default "localhost") -c, --since string Since -x, --xnic string xnic Use "swx support [command] --help" for more information about a command.</pre>	

cloudSwXtch Service Logs

In addition to the swx support logs, swXtch.io Support might request logs for the -ctrl/-repl services. For log requests, the standard command, `sudo journalctl -u`, can be used with either `swtch-ctrl.service` or `swtch-repl.service` to get a detailed breakdown of cloudSwXtch activity.

- `swtch-ctrl.service` - This will display information regarding the cloudSwXtch's control plane.
- `swtch-repl.service` - This will display information regarding the cloudSwXtch's replicator app that is on the data plane.

In addition, the log request command can be used for `swtch-bridge2` (Bridge Type 2) and `swXtch-bridge` (Bridge Type 1) for bridge-related logs.

It is recommended for users to send logs from both services to support@swtch.io. The logs should cover 24 hours worth of time, starting from before the issue to up until now.

Users can use any combination of the arguments below to create their logs.

Accessing and Following Logs (-f)

The following command will begin to display logs for either the `swtch-ctrl` or `swtch-repl` service at the time of the request. The `-f` argument will follow the logs and continually update. Logs prior to the call will not display.

Bash	Copy
<pre>sudo journalctl -u <swtch-ctrl.service OR swtch-repl.service> -f</pre>	

Note: A user will need to choose between `swtch-ctrl` or `swtch-repl`. The `.service` is not necessary and will work with or without.

Listing a Certain Number of Lines in a Log (-n)

The following argument can be used to list a specific number of lines in a log.

Bash	Copy
<pre>sudo journalctl -u <swtch-ctrl OR swtch-repl> -n <number of lines></pre>	

Example:

Bash	Copy
<pre>sudo journalctl -u swtch-repl -n 200</pre>	

Displaying Logs within a Timeframe (--since --until)

The following command will display logs within a set timeframe (between 2 dates).

Bash	Copy
<pre>sudo journalctl - u <swtch-ctrl or swtch-repl> --since <yyyy-mm-dd> --until <yyyy-mm-dd></pre>	

Example:

Bash	Copy
<code>sudo journalctl -u swxtch-repl --since 2023-03-07 --until 2023-03-10</code>	

--since "Yesterday", "today", "now"

You can also use the words "yesterday," "today" or "now" after --since to get logs from that time period.

Bash	Copy
<code>sudo journalctl -u swxtch-repl --since yesterday</code>	

Displaying Logs since Last Boot (-b)

To display logs since last boot, users can use the -b argument.

Bash	Copy
<code>sudo journalctl -u <swxtch-ctrl OR swxtch-repl> -b</code>	

List boots (--list-boot)

The following argument can be used to list all the boots in date/time order.

Bash	Copy
<code>sudo journalctl -u <swxtch-ctrl OR swxtch-repl> --list-boot</code>	

Exporting Logs (>)

The following command will export your logs to a .txt file. Logs should be emailed to support@swtch.io.

Bash	Copy
<code>sudo journalctl -u <swxtch-ctrl or swxtch-repl> > <file-name>.txt</code>	

Example:

Bash	Copy
<code>sudo journalctl -u swxtch-repl > cloudswxtch-test.txt</code>	

You can also combine arguments to export logs from a timeframe or from last boot. It is recommended that logs should cover 24 hours worth of time, starting from before the issue to up until now.

Example:

Bash	Copy
<p>LAST BOOT:</p> <pre>sudo journalctl -u swxtch-ctrl -b > cloudswxtch-test.txt</pre> <p>TIMEFRAME:</p> <pre>sudo journalctl -u swxtch-repl --since 2023-03-07 --until 2023-03-10 > cloudswxtch-test.txt</pre>	

Change Logs to UTC (--utc)

To switch logs from local time to UTC, use the following argument:

Bash	Copy
<code>sudo journalctl --utc</code>	

How to Install xNIC Dependencies in an Air-Gapped Environment

WHAT TO EXPECT

In this article, users will learn how to download and install the appropriate packages necessary to successfully deploy an xNIC in an air-gapped environment. Without the dependencies already on your air-gapped client, xNIC installation will fail.

Please follow the instructions based on your operating system:

- [RHEL-based Distro](#)
- [Ubuntu](#)

Installing Dependencies for RHEL-based Distro

The easiest way to install dependencies on RHEL-based distros is using the built-in tool called [repotrack](#). By using this tool, users can download the necessary packages with their dependencies using a single command. Then, they can copy the rpm files from an **ancillary VM** to the **target agents** (the VMs within the air-gapped environment) and install them there.

List of dependencies

Based on the version of operating system, a user will have different package requirements.

Major Version	Packages list
RHEL 8 or 9	libpcap iproute-tc bpftool jq
Amazon Linux	libpcap iproute-tc bpftool wget jq

Make sure you download the correct packages for your operating system.

Prerequisites: Install yum-utils to get repotrack

If you want to use repotrack, you will need to install yum-utils. To do this, enter the following command and respond yes when asked:

PowerShellCopy

\$ sudo yum install yum-utils

Here is an example output:

Example

PowerShellCopy

```
$ sudo yum install yum-utils
Last metadata expiration check: 0:04:24 ago on Fri 26 Jul 2024 03:35:50 PM UTC.
Dependencies resolved.
=====
Package                Architecture    Version         Repository                               Size
=====
Installing:
yum-utils              noarch         4.0.21-25.el8   rhui-rhel-8-for-x86_64-baseos-rhui-rpms 76 k

Transaction Summary
=====
Install 1 Package

Total download size: 76 k
Installed size: 23 k
Is this ok [y/N]: y
Downloading Packages:
yum-utils-4.0.21-25.el8.noarch.rpm                               1.5 MB/s | 76 kB    00:00
-----
Total                                                             987 kB/s | 76 kB    00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing      : 1/1
  Installing     : yum-utils-4.0.21-25.el8.noarch                1/1
  Running scriptlet: yum-utils-4.0.21-25.el8.noarch                1/1
  Verifying      : yum-utils-4.0.21-25.el8.noarch                1/1

Installed:
yum-utils-4.0.21-25.el8.noarch

Complete!
```

STEP ONE: Download All Packages and Dependencies onto your Ancillary VM

- 1. Select a VM that is connected to a repository or the internet that is similar to the one that will be used as an agent in your air-gapped environment. This will act as your "ancillary VM."
- 2. Execute the following command, replacing <packages-list> with the list associated with the correct version.

Shell

Bash	Copy
<pre>sudo repotrack <packages-list></pre>	

- a. This will download the packages and ALL their dependencies onto your ancillary VM.

STEP TWO: Copy the .rpm files to the Target VM

- 1. Secure copy the files now to the target machine on the air-gapped environment by issuing the following command. Here, we use a SCP command but there are other ways to move files. SCP needs both the ancillary and target VMs be connected, and sometimes that's not an option.

Shell

Bash	Copy
<pre>scp -i <private-key-path%file> *.rpm <target-username>@<target-hostname-or-IP>:<target-path></pre>	

- a. Example with replaced values:

Shell

Bash	Copy
<pre>scp -i ~/.ssh/key.priv *.rpm swxtchadmin@172.41.130.20:/home/swxtchadmin</pre>	

STEP THREE: Install the packages and dependencies on the Target VM

- 1. In the folder containing all the RPM files sent from the ancillary VM, issue the following command on the Target VM:

Shell

Bash	Copy
<pre>sudo dnf install *.rpm --disablerepo=* --allowdowngrading --skip-broken</pre>	

- a. This will analyze the system, skip what's already installed, give a report, and ask for confirmation.

You are now ready to [install the xNIC](#).

Installing Dependencies for Ubuntu distros

List of dependencies

The following packages must be installed on the target VM in order to successfully deploy an xNIC in an air-gapped environment:

Plaintext	Copy
<pre>libpcap-dev binutils iproute2 linux-tools-generic jq</pre>	

Instructions

- 1. Select a VM that is connected to a repository or the internet that is similar to the one that will be used as an agent in your air-gapped environment. This will act as your "ancillary VM."
- 2. Run the following command in the ancillary VM to acquire the updated repositories:

PowerShell

PowerShell	Copy
<pre>sudo apt update</pre>	

- a. Here is an example output:

Example

PowerShellCopy

```
$ sudo apt update
Hit:1 http://us-central1.gce.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-updates InRelease [128 kB]
Get:3 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-backports InRelease [128 kB]
Get:4 http://security.ubuntu.com/ubuntu focal-security InRelease [128 kB]
Get:5 http://us-central1.gce.archive.ubuntu.com/ubuntu focal/universe amd64 Packages [8628 kB]
Get:6 http://us-central1.gce.archive.ubuntu.com/ubuntu focal/universe Translation-en [5124 kB]
Get:7 http://us-central1.gce.archive.ubuntu.com/ubuntu focal/universe amd64 c-n-f Metadata [265 kB]
Get:8 http://us-central1.gce.archive.ubuntu.com/ubuntu focal/multiverse amd64 Packages [144 kB]
Get:9 http://us-central1.gce.archive.ubuntu.com/ubuntu focal/multiverse Translation-en [104 kB]
Get:10 http://us-central1.gce.archive.ubuntu.com/ubuntu focal/multiverse amd64 c-n-f Metadata [9136 B]
Get:11 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [3456 kB]
Get:12 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-updates/main Translation-en [538 kB]
Get:13 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-updates/main amd64 c-n-f Metadata [17.7 kB]
Get:14 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-updates/restricted amd64 Packages [3104 kB]
Get:15 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-updates/restricted Translation-en [434 kB]
Get:16 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-updates/restricted amd64 c-n-f Metadata [540 B]
Get:17 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-updates/universe amd64 Packages [1215 kB]
Get:18 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-updates/universe Translation-en [293 kB]
Get:19 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-updates/universe amd64 c-n-f Metadata [27.5 kB]
Get:20 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-updates/multiverse amd64 Packages [27.1 kB]
Get:21 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-updates/multiverse Translation-en [7936 B]
Get:22 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-updates/multiverse amd64 c-n-f Metadata [616 B]
Get:23 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-backports/main amd64 Packages [45.7 kB]
Get:24 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-backports/main Translation-en [16.3 kB]
Get:25 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-backports/main amd64 c-n-f Metadata [1420 B]
Get:26 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-backports/restricted amd64 c-n-f Metadata [116 B]
Get:27 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-backports/universe amd64 Packages [25.0 kB]
Get:28 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-backports/universe Translation-en [16.3 kB]
Get:29 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-backports/universe amd64 c-n-f Metadata [880 B]
Get:30 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-backports/multiverse amd64 c-n-f Metadata [116 B]
Get:31 http://security.ubuntu.com/ubuntu focal-security/main amd64 Packages [3086 kB]
Get:32 http://security.ubuntu.com/ubuntu focal-security/main Translation-en [459 kB]
Get:33 http://security.ubuntu.com/ubuntu focal-security/main amd64 c-n-f Metadata [14.0 kB]
Get:34 http://security.ubuntu.com/ubuntu focal-security/restricted amd64 Packages [2985 kB]
Get:35 http://security.ubuntu.com/ubuntu focal-security/restricted Translation-en [417 kB]
Get:36 http://security.ubuntu.com/ubuntu focal-security/restricted amd64 c-n-f Metadata [544 B]
Get:37 http://security.ubuntu.com/ubuntu focal-security/universe amd64 Packages [996 kB]
Get:38 http://security.ubuntu.com/ubuntu focal-security/universe Translation-en [211 kB]
Get:39 http://security.ubuntu.com/ubuntu focal-security/universe amd64 c-n-f Metadata [20.9 kB]
Get:40 http://security.ubuntu.com/ubuntu focal-security/multiverse amd64 Packages [24.8 kB]
Get:41 http://security.ubuntu.com/ubuntu focal-security/multiverse Translation-en [5968 B]
Get:42 http://security.ubuntu.com/ubuntu focal-security/multiverse amd64 c-n-f Metadata [540 B]
Fetched 32.1 MB in 5s (6704 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
22 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

PowerShell



3. Create an empty folder and enter that folder. This will house the dependencies.
4. Use the following command to get the list of URLs that will be downloaded:

PowerShellCopy

```
apt-get -y install --print-uris libpcap-dev binutils iproute2 linux-tools-generic jq | cut -d\ ' -f2 | grep http:// > apturls
```

a. This will not generate a visible output.

5. Download all the files with the following command:

PowerShellCopy

```
wget -i apturls
```

a. Here is an example output:

Example

PlaintextCopy

```
$ wget -i apturls
--2024-07-26 15:07:14-- http://us-central1.gce.archive.ubuntu.com/ubuntu/pool/main/n/numactl/libnuma1_2.0.12-1_amd64.deb
Resolving us-central1.gce.archive.ubuntu.com (us-central1.gce.archive.ubuntu.com)... 35.224.11.34, 35.193.225.125, 35.184.25.42,
...
Connecting to us-central1.gce.archive.ubuntu.com (us-central1.gce.archive.ubuntu.com)|35.224.11.34|:80... connected.
HTTP request sent, awaiting response... 200 OK
```

```

Length: 20828 (20K) [application/vnd.debian.binary-package]
Saving to: 'libnuma1_2.0.12-1_amd64.deb'

libnuma1_2.0.12-1_amd64.deb          100%
[=====] 20.34K  --.-KB/s  in 0.001s

2024-07-26 15:07:14 (20.2 MB/s) - 'libnuma1_2.0.12-1_amd64.deb' saved [20828/20828]

--2024-07-26 15:07:14-- http://us-central1.gce.archive.ubuntu.com/ubuntu/pool/main/b/binutils/binutils-common_2.34-6ubuntu1.9_amd64.deb
Reusing existing connection to us-central1.gce.archive.ubuntu.com:80.
HTTP request sent, awaiting response... 200 OK
Length: 207936 (203K) [application/vnd.debian.binary-package]
Saving to: 'binutils-common_2.34-6ubuntu1.9_amd64.deb'

binutils-common_2.34-6ubuntu1.9_amd64.deb  100%
[=====] 203.06K  --.-KB/s  in 0.009s

2024-07-26 15:07:14 (22.5 MB/s) - 'binutils-common_2.34-6ubuntu1.9_amd64.deb' saved [207936/207936]

--2024-07-26 15:07:14-- http://us-central1.gce.archive.ubuntu.com/ubuntu/pool/main/b/binutils/libbinutils_2.34-6ubuntu1.9_amd64.deb
Reusing existing connection to us-central1.gce.archive.ubuntu.com:80.
HTTP request sent, awaiting response... 200 OK
Length: 474932 (464K) [application/vnd.debian.binary-package]
Saving to: 'libbinutils_2.34-6ubuntu1.9_amd64.deb'

libbinutils_2.34-6ubuntu1.9_amd64.deb  100%
[=====] 463.80K  --.-KB/s  in 0.01s

2024-07-26 15:07:14 (32.8 MB/s) - 'libbinutils_2.34-6ubuntu1.9_amd64.deb' saved [474932/474932]

--2024-07-26 15:07:14-- http://us-central1.gce.archive.ubuntu.com/ubuntu/pool/main/b/binutils/libctf-nobfd_2.34-6ubuntu1.9_amd64.deb
Reusing existing connection to us-central1.gce.archive.ubuntu.com:80.
HTTP request sent, awaiting response... 200 OK
Length: 48248 (47K) [application/vnd.debian.binary-package]
Saving to: 'libctf-nobfd_2.34-6ubuntu1.9_amd64.deb'

libctf-nobfd_2.34-6ubuntu1.9_amd64.deb  100%
[=====] 47.12K  --.-KB/s  in 0s

2024-07-26 15:07:14 (145 MB/s) - 'libctf-nobfd_2.34-6ubuntu1.9_amd64.deb' saved [48248/48248]

--2024-07-26 15:07:14-- http://us-central1.gce.archive.ubuntu.com/ubuntu/pool/main/b/binutils/libctf_2.34-6ubuntu1.9_amd64.deb
Reusing existing connection to us-central1.gce.archive.ubuntu.com:80.
HTTP request sent, awaiting response... 200 OK
Length: 46620 (46K) [application/vnd.debian.binary-package]
Saving to: 'libctf_2.34-6ubuntu1.9_amd64.deb'

libctf_2.34-6ubuntu1.9_amd64.deb  100%
[=====] 45.53K  --.-KB/s  in 0.001s

2024-07-26 15:07:14 (41.1 MB/s) - 'libctf_2.34-6ubuntu1.9_amd64.deb' saved [46620/46620]

--2024-07-26 15:07:14-- http://us-central1.gce.archive.ubuntu.com/ubuntu/pool/main/b/binutils/binutils-x86-64-linux-gnu_2.34-6ubuntu1.9_amd64.deb
Reusing existing connection to us-central1.gce.archive.ubuntu.com:80.
HTTP request sent, awaiting response... 200 OK
Length: 1613596 (1.5M) [application/vnd.debian.binary-package]
Saving to: 'binutils-x86-64-linux-gnu_2.34-6ubuntu1.9_amd64.deb'

binutils-x86-64-linux-gnu_2.34-6ubuntu1.9_a 100%
[=====] 1.54M  --.-KB/s  in 0.03s

2024-07-26 15:07:14 (59.6 MB/s) - 'binutils-x86-64-linux-gnu_2.34-6ubuntu1.9_amd64.deb' saved [1613596/1613596]

--2024-07-26 15:07:14-- http://us-central1.gce.archive.ubuntu.com/ubuntu/pool/main/b/binutils/binutils_2.34-6ubuntu1.9_amd64.deb
Reusing existing connection to us-central1.gce.archive.ubuntu.com:80.
HTTP request sent, awaiting response... 200 OK
Length: 3380 (3.3K) [application/vnd.debian.binary-package]
Saving to: 'binutils_2.34-6ubuntu1.9_amd64.deb'

binutils_2.34-6ubuntu1.9_amd64.deb  100%
[=====] 3.30K  --.-KB/s  in 0s

2024-07-26 15:07:14 (532 MB/s) - 'binutils_2.34-6ubuntu1.9_amd64.deb' saved [3380/3380]

--2024-07-26 15:07:14-- http://us-central1.gce.archive.ubuntu.com/ubuntu/pool/universe/libo/libonig/libonig5_6.9.4-1_amd64.deb
Reusing existing connection to us-central1.gce.archive.ubuntu.com:80.
HTTP request sent, awaiting response... 200 OK
Length: 141948 (139K) [application/vnd.debian.binary-package]
Saving to: 'libonig5_6.9.4-1_amd64.deb'

libonig5_6.9.4-1_amd64.deb  100%
[=====] 138.62K  --.-KB/s  in 0.005s

2024-07-26 15:07:14 (25.1 MB/s) - 'libonig5_6.9.4-1_amd64.deb' saved [141948/141948]

--2024-07-26 15:07:14-- http://us-central1.gce.archive.ubuntu.com/ubuntu/pool/universe/j/jq/libjq1_1.6-1ubuntu0.20.04.1_amd64.deb
Reusing existing connection to us-central1.gce.archive.ubuntu.com:80.
HTTP request sent, awaiting response... 200 OK
Length: 120892 (118K) [application/vnd.debian.binary-package]
Saving to: 'libjq1_1.6-1ubuntu0.20.04.1_amd64.deb'

```

```

libjq1_1.6-1ubuntu0.20.04.1_amd64.deb          100%
[=====] 118.06K  ---KB/s  in 0.04s

2024-07-26 15:07:14 (2.64 MB/s) - 'libjq1_1.6-1ubuntu0.20.04.1_amd64.deb' saved [120892/120892]

--2024-07-26 15:07:14-- http://us-central1.gce.archive.ubuntu.com/ubuntu/pool/universe/j/jq/jq_1.6-1ubuntu0.20.04.1_amd64.deb
Reusing existing connection to us-central1.gce.archive.ubuntu.com:80.
HTTP request sent, awaiting response... 200 OK
Length: 50196 (49K) [application/vnd.debian.binary-package]
Saving to: 'jq_1.6-1ubuntu0.20.04.1_amd64.deb'

jq_1.6-1ubuntu0.20.04.1_amd64.deb          100%
[=====] 49.02K  ---KB/s  in 0.002s

2024-07-26 15:07:14 (21.7 MB/s) - 'jq_1.6-1ubuntu0.20.04.1_amd64.deb' saved [50196/50196]

--2024-07-26 15:07:14-- http://us-central1.gce.archive.ubuntu.com/ubuntu/pool/main/g/glibc/libc-dev-bin_2.31-0ubuntu9.16_amd64.deb
Reusing existing connection to us-central1.gce.archive.ubuntu.com:80.
HTTP request sent, awaiting response... 200 OK
Length: 71612 (70K) [application/vnd.debian.binary-package]
Saving to: 'libc-dev-bin_2.31-0ubuntu9.16_amd64.deb'

libc-dev-bin_2.31-0ubuntu9.16_amd64.deb  100%
[=====] 69.93K  ---KB/s  in 0.001s

2024-07-26 15:07:14 (70.3 MB/s) - 'libc-dev-bin_2.31-0ubuntu9.16_amd64.deb' saved [71612/71612]

--2024-07-26 15:07:14-- http://us-central1.gce.archive.ubuntu.com/ubuntu/pool/main/l/linux/linux-libc-dev_5.4.0-190.210_amd64.deb
Reusing existing connection to us-central1.gce.archive.ubuntu.com:80.
HTTP request sent, awaiting response... 200 OK
Length: 1138252 (1.1M) [application/vnd.debian.binary-package]
Saving to: 'linux-libc-dev_5.4.0-190.210_amd64.deb'

linux-libc-dev_5.4.0-190.210_amd64.deb  100%
[=====] 1.08M  ---KB/s  in 0.01s

2024-07-26 15:07:14 (94.7 MB/s) - 'linux-libc-dev_5.4.0-190.210_amd64.deb' saved [1138252/1138252]

--2024-07-26 15:07:14-- http://us-central1.gce.archive.ubuntu.com/ubuntu/pool/main/libx/libxcrypt/libcrypt-dev_4.4.10-10ubuntu4_amd64.deb
Reusing existing connection to us-central1.gce.archive.ubuntu.com:80.
HTTP request sent, awaiting response... 200 OK
Length: 104104 (102K) [application/vnd.debian.binary-package]
Saving to: 'libcrypt-dev_4.4.10-10ubuntu4_amd64.deb'

libcrypt-dev_4.4.10-10ubuntu4_amd64.deb  100%
[=====] 101.66K  ---KB/s  in 0.001s

2024-07-26 15:07:14 (111 MB/s) - 'libcrypt-dev_4.4.10-10ubuntu4_amd64.deb' saved [104104/104104]

--2024-07-26 15:07:14-- http://us-central1.gce.archive.ubuntu.com/ubuntu/pool/main/g/glibc/libc6-dev_2.31-0ubuntu9.16_amd64.deb
Reusing existing connection to us-central1.gce.archive.ubuntu.com:80.
HTTP request sent, awaiting response... 200 OK
Length: 2520144 (2.4M) [application/vnd.debian.binary-package]
Saving to: 'libc6-dev_2.31-0ubuntu9.16_amd64.deb'

libc6-dev_2.31-0ubuntu9.16_amd64.deb  100%
[=====] 2.40M  ---KB/s  in 0.04s

2024-07-26 15:07:14 (56.0 MB/s) - 'libc6-dev_2.31-0ubuntu9.16_amd64.deb' saved [2520144/2520144]

--2024-07-26 15:07:14-- http://us-central1.gce.archive.ubuntu.com/ubuntu/pool/main/e/elfutils/libdw1_0.176-1.1ubuntu0.1_amd64.deb
Reusing existing connection to us-central1.gce.archive.ubuntu.com:80.
HTTP request sent, awaiting response... 200 OK
Length: 225528 (220K) [application/vnd.debian.binary-package]
Saving to: 'libdw1_0.176-1.1ubuntu0.1_amd64.deb'

libdw1_0.176-1.1ubuntu0.1_amd64.deb  100%
[=====] 220.24K  ---KB/s  in 0.005s

2024-07-26 15:07:14 (42.4 MB/s) - 'libdw1_0.176-1.1ubuntu0.1_amd64.deb' saved [225528/225528]

--2024-07-26 15:07:14-- http://us-central1.gce.archive.ubuntu.com/ubuntu/pool/main/libp/libpcap/libpcap0.8-dev_1.9.1-3_amd64.deb
Reusing existing connection to us-central1.gce.archive.ubuntu.com:80.
HTTP request sent, awaiting response... 200 OK
Length: 244140 (238K) [application/vnd.debian.binary-package]
Saving to: 'libpcap0.8-dev_1.9.1-3_amd64.deb'

libpcap0.8-dev_1.9.1-3_amd64.deb  100%
[=====] 238.42K  ---KB/s  in 0.009s

2024-07-26 15:07:14 (27.1 MB/s) - 'libpcap0.8-dev_1.9.1-3_amd64.deb' saved [244140/244140]

--2024-07-26 15:07:14-- http://us-central1.gce.archive.ubuntu.com/ubuntu/pool/main/libp/libpcap/libpcap-dev_1.9.1-3_amd64.deb
Reusing existing connection to us-central1.gce.archive.ubuntu.com:80.
HTTP request sent, awaiting response... 200 OK
Length: 3484 (3.4K) [application/vnd.debian.binary-package]
Saving to: 'libpcap-dev_1.9.1-3_amd64.deb'

libpcap-dev_1.9.1-3_amd64.deb  100%
[=====] 3.40K  ---KB/s  in 0s

2024-07-26 15:07:14 (568 MB/s) - 'libpcap-dev_1.9.1-3_amd64.deb' saved [3484/3484]

```

```
--2024-07-26 15:07:14-- http://us-central1.gce.archive.ubuntu.com/ubuntu/pool/main/l/linux/linux-tools-common_5.4.0-190.210_all.deb
Reusing existing connection to us-central1.gce.archive.ubuntu.com:80.
HTTP request sent, awaiting response... 200 OK
Length: 225144 (220K) [application/vnd.debian.binary-package]
Saving to: 'linux-tools-common_5.4.0-190.210_all.deb'

linux-tools-common_5.4.0-190.210_all.deb 100%
[=====] 219.87K --.-KB/s in 0.007s

2024-07-26 15:07:14 (29.4 MB/s) - 'linux-tools-common_5.4.0-190.210_all.deb' saved [225144/225144]

--2024-07-26 15:07:14-- http://us-central1.gce.archive.ubuntu.com/ubuntu/pool/main/l/linux/linux-tools-5.4.0-190.210_amd64.deb
Reusing existing connection to us-central1.gce.archive.ubuntu.com:80.
HTTP request sent, awaiting response... 200 OK
Length: 5633692 (5.4M) [application/vnd.debian.binary-package]
Saving to: 'linux-tools-5.4.0-190.210_amd64.deb'

linux-tools-5.4.0-190.210_amd64.d 100%
[=====] 5.37M --.-KB/s in 0.1s

2024-07-26 15:07:14 (39.3 MB/s) - 'linux-tools-5.4.0-190.210_amd64.deb' saved [5633692/5633692]

--2024-07-26 15:07:14-- http://us-central1.gce.archive.ubuntu.com/ubuntu/pool/main/l/linux/linux-tools-5.4.0-190-generic_5.4.0-190.210_amd64.deb
Reusing existing connection to us-central1.gce.archive.ubuntu.com:80.
HTTP request sent, awaiting response... 200 OK
Length: 1996 (1.9K) [application/vnd.debian.binary-package]
Saving to: 'linux-tools-5.4.0-190-generic_5.4.0-190.210_amd64.deb'

linux-tools-5.4.0-190-generic_5.4.0-190.210 100%
[=====] 1.95K --.-KB/s in 0s

2024-07-26 15:07:14 (354 MB/s) - 'linux-tools-5.4.0-190-generic_5.4.0-190.210_amd64.deb' saved [1996/1996]

--2024-07-26 15:07:14-- http://us-central1.gce.archive.ubuntu.com/ubuntu/pool/main/l/linux-meta/linux-tools-generic_5.4.0.190.188_amd64.deb
Reusing existing connection to us-central1.gce.archive.ubuntu.com:80.
HTTP request sent, awaiting response... 200 OK
Length: 2480 (2.4K) [application/vnd.debian.binary-package]
Saving to: 'linux-tools-generic_5.4.0.190.188_amd64.deb'

linux-tools-generic_5.4.0.190.188_amd64.deb 100%
[=====] 2.42K --.-KB/s in 0s

2024-07-26 15:07:14 (410 MB/s) - 'linux-tools-generic_5.4.0.190.188_amd64.deb' saved [2480/2480]

--2024-07-26 15:07:14-- http://us-central1.gce.archive.ubuntu.com/ubuntu/pool/main/m/manpages/manpages-dev_5.05-1_all.deb
Reusing existing connection to us-central1.gce.archive.ubuntu.com:80.
HTTP request sent, awaiting response... 200 OK
Length: 2265524 (2.2M) [application/vnd.debian.binary-package]
Saving to: 'manpages-dev_5.05-1_all.deb'

manpages-dev_5.05-1_all.deb 100%
[=====] 2.16M --.-KB/s in 0.05s

2024-07-26 15:07:14 (47.2 MB/s) - 'manpages-dev_5.05-1_all.deb' saved [2265524/2265524]

FINISHED --2024-07-26 15:07:14--
Total wall clock time: 0.5s
Downloaded: 22 files, 14M in 0.4s (40.0 MB/s)
```

6. Compress every .deb file into a .tar.gz file:

PowerShell	Copy
tar -czvf debs.tar.gz *.deb	

a. Here is an example output:

Example

PowerShell	Copy
<pre>\$ tar -czvf debs.tar.gz *.deb binutils-common_2.34-6ubuntu1.9_amd64.deb binutils-x86-64-linux-gnu_2.34-6ubuntu1.9_amd64.deb binutils_2.34-6ubuntu1.9_amd64.deb jq_1.6-1ubuntu0.20.04.1_amd64.deb libbinutils_2.34-6ubuntu1.9_amd64.deb libc-dev-bin_2.31-0ubuntu9.16_amd64.deb libc6-dev_2.31-0ubuntu9.16_amd64.deb libcrypt-dev_4.4.10-10ubuntu4_amd64.deb libctf-nobfd_2.34-6ubuntu1.9_amd64.deb libctf_2.34-6ubuntu1.9_amd64.deb libdw1_0.176-1.1ubuntu0.1_amd64.deb libjq1_1.6-1ubuntu0.20.04.1_amd64.deb libnuma1_2.0.12-1_amd64.deb libonig5_6.9.4-1_amd64.deb libpcap-dev_1.9.1-3_amd64.deb libpcap0.8-dev_1.9.1-3_amd64.deb linux-libc-dev_5.4.0-190.210_amd64.deb linux-tools-5.4.0-190-generic_5.4.0-190.210_amd64.deb linux-tools-5.4.0-190_5.4.0-190.210_amd64.deb linux-tools-common_5.4.0-190.210_all.deb linux-tools-generic_5.4.0-190.188_amd64.deb manpages-dev_5.05-1_all.deb</pre>	

7. Move that **debs.tar.gz** to the target VM in the air-gapped environment. One option is to use the SCP command if both VMs are connected over the network and there's no firewall blocking traffic.
8. Check the file details with the following command:

PowerShell	Copy
<pre>ll debs.tar.gz</pre>	

a. Here is an example output:

Example	
Plaintext	Copy
<pre>\$ ll debs.tar.gz -rw-rw-r-- 1 fsuarez0 fsuarez0 15165956 Jul 26 15:09 debs.tar.gz</pre>	

9. Uncompress the **.tar.gz** in the air-gapped VM:

PowerShell	Copy
<pre>tar -xvzf debs.tar.gz</pre>	

a. Here is an example output:

Example	
Plaintext	Copy
<pre>\$ tar -xvzf debs.tar.gz binutils-common_2.34-6ubuntu1.9_amd64.deb binutils-x86-64-linux-gnu_2.34-6ubuntu1.9_amd64.deb binutils_2.34-6ubuntu1.9_amd64.deb jq_1.6-1ubuntu0.20.04.1_amd64.deb libbinutils_2.34-6ubuntu1.9_amd64.deb libc-dev-bin_2.31-0ubuntu9.16_amd64.deb libc6-dev_2.31-0ubuntu9.16_amd64.deb libcrypt-dev_4.4.10-10ubuntu4_amd64.deb libctf-nobfd_2.34-6ubuntu1.9_amd64.deb libctf_2.34-6ubuntu1.9_amd64.deb libdw1_0.176-1.1ubuntu0.1_amd64.deb libjq1_1.6-1ubuntu0.20.04.1_amd64.deb libnuma1_2.0.12-1_amd64.deb libonig5_6.9.4-1_amd64.deb libpcap-dev_1.9.1-3_amd64.deb libpcap0.8-dev_1.9.1-3_amd64.deb linux-libc-dev_5.4.0-190.210_amd64.deb linux-tools-5.4.0-190-generic_5.4.0-190.210_amd64.deb linux-tools-5.4.0-190_5.4.0-190.210_amd64.deb linux-tools-common_5.4.0-190.210_all.deb linux-tools-generic_5.4.0-190.188_amd64.deb manpages-dev_5.05-1_all.deb</pre>	

10. Install all the **.deb** files with the following command:

PowerShell	Copy
sudo dpkg -i *.deb	

a. Here is an example output:

Example

PowerShell	Copy
<pre>\$ sudo dpkg -i *.deb Selecting previously unselected package binutils-common:amd64. (Reading database ... 62143 files and directories currently installed.) Preparing to unpack binutils-common_2.34-6ubuntu1.9_amd64.deb ... Unpacking binutils-common:amd64 (2.34-6ubuntu1.9) ... Selecting previously unselected package binutils-x86-64-linux-gnu. Preparing to unpack binutils-x86-64-linux-gnu_2.34-6ubuntu1.9_amd64.deb ... Unpacking binutils-x86-64-linux-gnu (2.34-6ubuntu1.9) ... Selecting previously unselected package binutils. Preparing to unpack binutils_2.34-6ubuntu1.9_amd64.deb ... Unpacking binutils (2.34-6ubuntu1.9) ... Selecting previously unselected package jq. Preparing to unpack jq_1.6-1ubuntu0.20.04.1_amd64.deb ... Unpacking jq (1.6-1ubuntu0.20.04.1) ... Selecting previously unselected package libbinutils:amd64. Preparing to unpack libbinutils_2.34-6ubuntu1.9_amd64.deb ... Unpacking libbinutils:amd64 (2.34-6ubuntu1.9) ... Selecting previously unselected package libc-dev-bin. Preparing to unpack libc-dev-bin_2.31-0ubuntu9.16_amd64.deb ... Unpacking libc-dev-bin (2.31-0ubuntu9.16) ... Selecting previously unselected package libc6-dev:amd64. Preparing to unpack libc6-dev_2.31-0ubuntu9.16_amd64.deb ... Unpacking libc6-dev:amd64 (2.31-0ubuntu9.16) ... Selecting previously unselected package libcrypt-dev:amd64. Preparing to unpack libcrypt-dev_4.4.10-10ubuntu4_amd64.deb ... Unpacking libcrypt-dev:amd64 (1:4.4.10-10ubuntu4) ... Selecting previously unselected package libctf-nobfd0:amd64. Preparing to unpack libctf-nobfd0_2.34-6ubuntu1.9_amd64.deb ... Unpacking libctf-nobfd0:amd64 (2.34-6ubuntu1.9) ... Selecting previously unselected package libctf0:amd64. Preparing to unpack libctf0_2.34-6ubuntu1.9_amd64.deb ... Unpacking libctf0:amd64 (2.34-6ubuntu1.9) ... Selecting previously unselected package libdw1:amd64. Preparing to unpack libdw1_0.176-1.1ubuntu0.1_amd64.deb ... Unpacking libdw1:amd64 (0.176-1.1ubuntu0.1) ... Selecting previously unselected package libjq1:amd64. Preparing to unpack libjq1_1.6-1ubuntu0.20.04.1_amd64.deb ... Unpacking libjq1:amd64 (1.6-1ubuntu0.20.04.1) ... Selecting previously unselected package libnuma1:amd64. Preparing to unpack libnuma1_2.0.12-1_amd64.deb ... Unpacking libnuma1:amd64 (2.0.12-1) ... Selecting previously unselected package libonig5:amd64. Preparing to unpack libonig5_6.9.4-1_amd64.deb ... Unpacking libonig5:amd64 (6.9.4-1) ... Selecting previously unselected package libpcap-dev:amd64. Preparing to unpack libpcap-dev_1.9.1-3_amd64.deb ... Unpacking libpcap-dev:amd64 (1.9.1-3) ... Selecting previously unselected package libpcap0.8-dev:amd64. Preparing to unpack libpcap0.8-dev_1.9.1-3_amd64.deb ... Unpacking libpcap0.8-dev:amd64 (1.9.1-3) ... Selecting previously unselected package linux-libc-dev:amd64. Preparing to unpack linux-libc-dev_5.4.0-190.210_amd64.deb ... Unpacking linux-libc-dev:amd64 (5.4.0-190.210) ... Selecting previously unselected package linux-tools-5.4.0-190-generic. Preparing to unpack linux-tools-5.4.0-190-generic_5.4.0-190.210_amd64.deb ... Unpacking linux-tools-5.4.0-190-generic (5.4.0-190.210) ... Selecting previously unselected package linux-tools-5.4.0-190. Preparing to unpack linux-tools-5.4.0-190_5.4.0-190.210_amd64.deb ... Unpacking linux-tools-5.4.0-190 (5.4.0-190.210) ... Selecting previously unselected package linux-tools-common. Preparing to unpack linux-tools-common_5.4.0-190.210_all.deb ... Unpacking linux-tools-common (5.4.0-190.210) ... Selecting previously unselected package linux-tools-generic. Preparing to unpack linux-tools-generic_5.4.0-190.188_amd64.deb ... Unpacking linux-tools-generic (5.4.0-190.188) ... Selecting previously unselected package manpages-dev. Preparing to unpack manpages-dev_5.05-1_all.deb ... Unpacking manpages-dev (5.05-1) ... Setting up binutils-common:amd64 (2.34-6ubuntu1.9) ... Setting up libbinutils:amd64 (2.34-6ubuntu1.9) ... Setting up libc-dev-bin (2.31-0ubuntu9.16) ... Setting up libcrypt-dev:amd64 (1:4.4.10-10ubuntu4) ... Setting up libctf-nobfd0:amd64 (2.34-6ubuntu1.9) ... Setting up libctf0:amd64 (2.34-6ubuntu1.9) ... Setting up libdw1:amd64 (0.176-1.1ubuntu0.1) ... Setting up libnuma1:amd64 (2.0.12-1) ... Setting up libonig5:amd64 (6.9.4-1) ... Setting up linux-libc-dev:amd64 (5.4.0-190.210) ... Setting up linux-tools-common (5.4.0-190.210) ... Setting up manpages-dev (5.05-1) ... Setting up binutils-x86-64-linux-gnu (2.34-6ubuntu1.9) ...</pre>	

```
Setting up binutils (2.34-6ubuntu1.9) ...  
Setting up libc6-dev:amd64 (2.31-0ubuntu9.16) ...  
Setting up libjq1:amd64 (1.6-1ubuntu0.20.04.1) ...  
Setting up libpcap0.8-dev:amd64 (1.9.1-3) ...  
Setting up linux-tools-5.4.0-190 (5.4.0-190.210) ...  
Setting up jq (1.6-1ubuntu0.20.04.1) ...  
Setting up libpcap-dev:amd64 (1.9.1-3) ...  
Setting up linux-tools-5.4.0-190-generic (5.4.0-190.210) ...  
Setting up linux-tools-generic (5.4.0-190.188) ...  
Processing triggers for man-db (2.9.1-1) ...  
Processing triggers for libc-bin (2.31-0ubuntu9.16) ...
```

11. You should now be able to [Install an xNIC](#) without an internet connection.

How to View cloudSwXtch Bridge Logs

WHAT TO EXPECT

In this article, users will learn how to access cloudSwXtch Bridge logs for the purpose of troubleshooting. Common arguments that can be used when compiling information for the Support team at swXtch.io will also be discussed.

Support may also request for you to send xNIC logs. For more information, see [How to Find xNIC Logs](#).

cloudSwXtch Bridge Service Logs

For log requests, the standard command, `sudo journalctl -u` can be used with either `swxtch-bridge2` (Bridge Type 2) or `swxtch-bridge` (Bridge Type 1) to get a detailed breakdown of cloudSwXtch Bridge activity.

For example:

Bash	Copy
<pre>sudo journalctl -u swxtch-bridge2</pre>	

It is recommended to send logs to support@swxtch.io, coverign 24 hours worth of time, starting form before the issue to up until now.

Users can use any combination of arguments below to create their logs.

Accessing and Following Logs (-f)

The following command will begin to display cloudSwXtch Bridge logs at the time of the request. The `-f` argument will follow the logs and continually update. Logs prior to the call will not display.

Bash	Copy
<pre>sudo journalctl -u <swxtch-bridge2 OR swxtch-bridge> -f</pre>	

Note: A user will need to choose between `swxtch-bridge2` or `swxtch-bridge`.

Example:

Bash	Copy
<pre>sudo journalctl -u swxtch-bridge2 -n 200</pre>	

Displaying Logs within a Timeframe (--since --until)

The following command will display logs within a set timeframe (between 2 dates).

Bash	Copy
<pre>sudo journalctl - u <swxtch-bridge2 or swxtch-bridge> --since <yyyy-mm-dd> --until <yyyy-mm-dd></pre>	

Example:

Bash	Copy
<pre>sudo journalctl - u swxtch-bridge2 --since 2023-03-07 --until 2023-03-10</pre>	

--since "Yesterday", "today", "now"

You can also use the words "yesterday," "today", or "now" after `--since` to get logs from that time period.

Bash	Copy
<pre>sudo journalctl - u swxtch-bridge2 --since yesterday</pre>	

Displaying Logs since Last Boot (-b)

To display logs since last boot, users can use the `-b` argument.

Bash	Copy
<pre>sudo journalctl -u <swxtch-bridge2 OR swxtch-bridge> -b</pre>	

List boots (--list-boot)

The following argument can be used to list all the boots in date/time order.

Bash	Copy
<pre>sudo journalctl -u <swxtch-bridge2 or swxtch-bridge> --list-boot</pre>	

Exporting Logs (>)

The following command will export your logs to a .txt file. Logs should be emailed to support@swtch.io.

Bash	Copy
<pre>sudo journalctl -u <swtch-bridge2 or swtch-bridge> > <file-name>.txt</pre>	

Example:

Bash	Copy
<pre>sudo journalctl -u swtch-bridge2 > cloudswtch-test.txt</pre>	

You can also combine arguments to export logs from a timeframe or from last boot. It is recommended that logs should cover 24 hours worth of time, starting from before the issue to up until now.

Example:

Bash	Copy
<pre>LAST BOOT: sudo journalctl -u swtch-bridge2 -b > cloudswtch-test.txt TIMEFRAME: sudo journalctl -u swtch-bridge2 --since 2023-03-07 --until 2023-03-10 > cloudswtch-test.txt</pre>	

Change Logs to UTC (--utc)

To switch logs from local time to UTC, use the following argument:

Bash	Copy
<pre>sudo journalctl --utc</pre>	

How to Find xNIC Logs

WHAT TO EXPECT

In this article, you will learn how to find xNICs logs on your VM and how to alter its verbosity level.

swXtch.io Support may also request for you to send cloudSwXtch logs. For more information, see [How to View cloudSwXtch Logs for Troubleshooting](#).

Locating xNIC Logs

An xNIC installed on a virtual machine creates one .log file per day with the following naming structure: **swtch-xnic-YYYYMMDD.log**. If the file size exceeds the maximum within the same day (16MB), it will be renamed by adding a counter as a suffix. Then, a new file will be created.

To find your logs, use the following file paths:

Windows

- C:\Users\Public\swxlogs\xnic-control
- C:\Users\Public\swxlogs\xnic-data

Linux

- /var/log/swx/xnic-control
- /var/log/swx/xnic-data

For Linux, logs can also be viewed by using either journalctl examples below:

xnic-control

Bash	Copy
<pre>sudo journalctl -u swtch-xnic-control -n 500 -f</pre>	

xnic-data

Bash	Copy
<pre>sudo journalctl -u swtch-xnic-data -n 500 -f</pre>	

Please note: Standard journalctl arguments apply. The above examples use -n for number of lines and -f to follow.

Log File Deletion

Log files older than 30 days are automatically deleted.

What is verbosity?

Depending on the level of verbosity detailed in the xNIC config file, a log will contain different application messages and usage statistics. The default verbosity level after xNIC installation is 0, which means that no periodic statistics are being reported. It will only show start and stop information as well as critical errors.

A user can change the verbosity to pull more information out from their xNIC. The levels are detailed below:

- Level 0:** Only show start and stop info as well as critical errors. This is the default.
- Level 1:** Shows statistics and IGMP messages
- Level 2:** Additional control messages
- Level 3:** Hexadecimal dumps of control/config packages
- Level 4:** Hexadecimal dumps of data packages

An average user would typically only need up to Level 2 for troubleshooting issues with their xNIC.

Verbosity and File Size

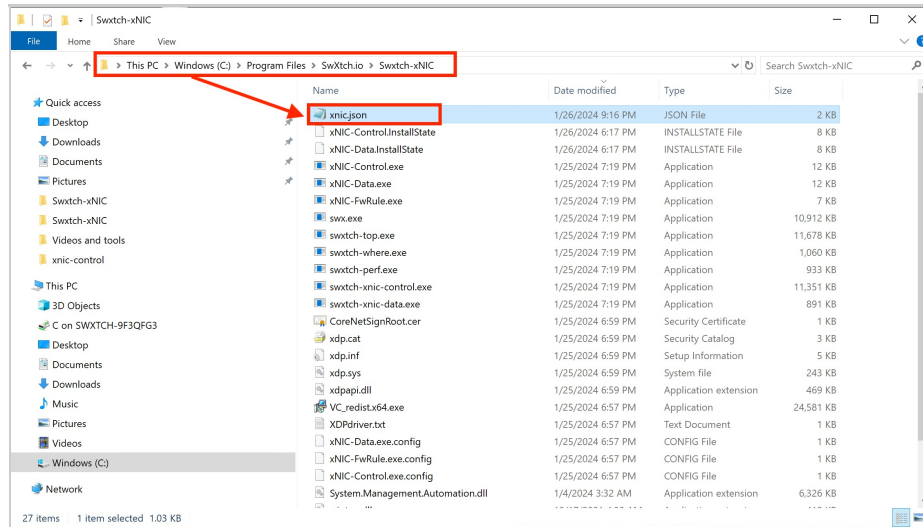
Please note that increasing the verbosity level of future logs will result in larger file sizes. It is recommended to revert back to the default Level 0 when testing and troubleshooting is complete.

How to Change Verbosity

To change the verbosity, a user can manually edit the xNIC config file on their VM.

For Windows:

- Go to the Swtch-xNIC folder on the VM you have an xNIC installed. Make sure it is the xNIC you want logs for.
 - C:\Program Files\SwXtch.io\Swtch-xNIC



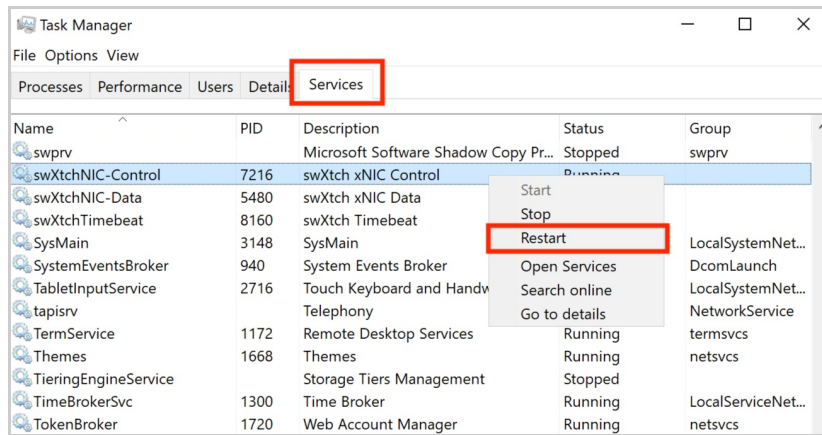
2. Open the **xnic.json** file.
3. For **xNIC-Data**, change the number next to "verbosity" so that it matches the level you desire. The default is 0.
4. For **xNIC-Control**, modify the **StatsReportWait** to change the time in seconds of the report stats.

```

xnic.json - Notepad
File Edit Format View Help
{
  "swxtch": "10.2.128.10",
  "controlInterface": "Ethernet",
  "dataInterface": "Ethernet 2",
  "dataPort": 9999,
  "xnicType": 1,
  "dataPlaneSpecs": {
    "verbosity": 0,
    "virtualInterface": {
      "name": "swxtch-tun0",
      "ip": "172.30.0.0",
      "subnet": "255.255.0.0",
      "mtu": 4096
    }
  },
  "ha": {
    "maxTimeToBufferPacketsMs": 50,
    "bufferSizeInPackets": 131072,
    "protocol": "rtp"
  },
  "streamSpecs": {
    "mmcProducerEnable": true,
    "multipleMulticastGroups": [
      {
        "parent": "239.2.2.2:4000",
        "children": [
          "239.1.1.1:4000",
          "239.1.1.2:4000"
        ]
      },
      {
        "parent": "239.3.3.3:4000",
        "children": [
          "239.1.1.3:4000",
          "239.1.1.4:4000"
        ]
      }
    ]
  },
  "statsReportWait": 60
}

```

5. **Save and Close** the json file.
6. Open "Task Manager" and go to the "Services" tab towards the top of the window.
7. To apply changes from the json file, scroll down to "swXtchNIC-Control" and right-click on it.
8. Select "Restart."
 - a. **Please note:** Only xNIC-Control has to be restarted to update both control and data logs.



Your selection in verbosity will now be applied to future logs. This should only be changed if directed by swXtch.io and ensure that they are set back to default when troubleshooting is complete.

For Linux:

1. Enter the following command to view your config file in the Bash terminal. Make sure it is on the xNIC you want logs for.

Text



```
Bash Copy
sudo nano /var/opt/swxtch/swxtch-xnic/swxtch-xnic.conf
```

2. For xNIC-Data, **change** the number next to "verbose" so that it matches the level you desire. The default is 0.
3. For xNIC-Control, **modify** the StatsReportWait to change the time in seconds of the report stats.



4. **Save and Exit** the file.
5. **Restart** your swxtch-xnic-control by using the following command. **Please note:** Only xNIC-Control has to be restarted to update both control and data logs.

Text



```
Bash Copy
sudo systemctl restart swxtch-xnic-control
```

Your selection in verbosity will now be applied to future logs. This should only be changed if directed by swXtch.io and ensure that they are set back to default when troubleshooting is complete.

PRO-TIP

Rename your existing log file before restarting the xNIC service in order to differentiate it with the freshly generated log file containing the new verbosity data.

How to License a cloudSwXtch

WHAT TO EXPECT

In this article, users will learn the appropriate steps for licensing their cloudSwXtch instance.

1. Log into the newly created cloudSwXtch VM.
2. Run the command:

Bash	Copy
<pre>sudo /swxtch/swxtch-top dashboard --swxtch localhost</pre>	

3. The swXtch-top dashboard will display.
- Alternatively, if you do not want to open swXtch-top, you can also use the following command to get the SwXtchID:

Bash	Copy
<pre>curl -s http://127.0.0.1/top/dashboard grep -m 2 -Eo '"fingerprint"[^,]*' head -1</pre>	

4. Copy the SwXtchID and email it to support@swxtch.io requesting a license.
5. When you receive the license, upload it onto the cloudSwXtch VM.
6. Move the license.json file to the /swxtch directory using the following command replacing user with the appropriate value:

Bash	Copy
<pre>sudo mv /home/<user>/license.json /swxtch</pre>	

7. Return to the swxtch-top dashboard again check the license took hold.

How to set MTU size

In some cases the MTU Size of the multicast group may exceed the 1500 set limit in Windows and Linux virtual machines. This article will explain how to increase the MTU size if this should occur.

To know if the MTU size has been exceeded Wireshark or tcpdump can be used. Below is an example from Wireshark.

Time	Source	Destination	Protocol	Length	Info
1 0.000000	172.30.0.11	239.192.10.160	UDP	1482	60000 → 4002 [BAD UDP LENGTH 1449 > IP PAYLOAD LENGTH] Len=1441
2 0.000000	172.30.0.11	239.192.10.160	UDP	1482	60000 → 4002 [BAD UDP LENGTH 1450 > IP PAYLOAD LENGTH] Len=1442
3 0.000000	172.30.0.11	239.192.10.160	UDP	1482	60000 → 4002 [BAD UDP LENGTH 1450 > IP PAYLOAD LENGTH] Len=1442
4 0.000000	172.30.0.11	239.192.10.160	UDP	1482	60000 → 4002 [BAD UDP LENGTH 1450 > IP PAYLOAD LENGTH] Len=1442
5 0.000203	172.30.0.11	239.192.10.160	UDP	1482	60000 → 4002 [BAD UDP LENGTH 1450 > IP PAYLOAD LENGTH] Len=1442
6 0.000203	172.30.0.11	239.192.10.160	UDP	1482	60000 → 4002 [BAD UDP LENGTH 1450 > IP PAYLOAD LENGTH] Len=1442

your title goes here

your content goes here

:::Note: The UDP length error shows it is exceeding the Length.

Linux update MTU Size:

First check MTU current size by running the following command:

None	Copy
ifconfig grep mtu	

Example:

None	Copy
someadmin@my-agent-101:~\$ ifconfig grep mtu enP43852s1: flags=6211<UP,BROADCAST,RUNNING,SLAVE,MULTICAST> mtu 1500 enP4589s2: flags=6211<UP,BROADCAST,RUNNING,SLAVE,MULTICAST> mtu 1500 eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500 eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500 lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536	

Note: The MTU size of eth1 = 1500

To change MTU size to 2000 for example - use the command below:

None	Copy
sudo ifconfig eth1 mtu 2000 up	

Validate it is set to new value in this case 2000 by running this command:

None	Copy
ifconfig grep mtu	

Example:

None	Copy
someadmin@my-agent-101:~\$ ifconfig grep mtu enP43852s1: flags=6211<UP,BROADCAST,RUNNING,SLAVE,MULTICAST> mtu 1500 enP4589s2: flags=6211<UP,BROADCAST,RUNNING,SLAVE,MULTICAST> mtu 2000 eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500 eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 2000 lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536	

Note: The MTU size of eth1 is now = 2000

Windows Update MTU Size

1. Check MTU Size by running this command:

None	Copy
netsh interface ipv4 show subinterfaces	

You will see a list of network interfaces.

2. Set the MTU Size (in this case to 2000) using the following commands:

None	Copy
netsh	

None	Copy
interface	

None	Copy
ipv4	

and finally to actually set the value:

```
None Copy
```

```
set subinterface "Local Area Connection" mtu=2000 store=persistent
```

Where "Local Area Connection" is the Ethernet adaptor to be set - for example:

```
Administrator: Windows PowerShell
PS C:\> ipconfig

Windows IP Configuration

Unknown adapter swtch-tun:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

Ethernet adapter Ethernet:

    Connection-specific DNS Suffix  . : qw4ieorzzsjedntce414ja
    Link-local IPv6 Address . . . . . : fe80::657a:1e18:2874:8
    IPv4 Address. . . . . : 10.2.192.15
    Subnet Mask . . . . . : 255.255.252.0
    Default Gateway . . . . . :

Ethernet adapter Ethernet 2:

    Connection-specific DNS Suffix  . : qw4ieorzzsjedntce414jaktbh.bx.internal.cloudapp.net
    Link-local IPv6 Address . . . . . : fe80::853c:a2ac:cf78:842f%114
    IPv4 Address. . . . . : 10.2.128.15
    Subnet Mask . . . . . : 255.255.252.0
    Default Gateway . . . . . : 10.2.128.1

PS C:\>
```

```
Administrator: Command Prompt - netsh
C:\Users\testadmin>netsh
netsh>interface
In future versions of Windows, Microsoft might remove the Netsh functionality
for TCP/IP.

Microsoft recommends that you transition to Windows PowerShell if you currently
use netsh to configure and manage TCP/IP.

Type Get-Command -Module NetTCP/IP at the Windows PowerShell prompt to view
a list of commands to manage TCP/IP.

Visit https://go.microsoft.com/fwlink/?LinkID=217627 for additional information
about PowerShell commands for TCP/IP.

netsh interface>ipv4
netsh interface ipv4>set subinterface "Ethernet 2" mtu=1200 store=persistent
Ok.
netsh interface ipv4>
```

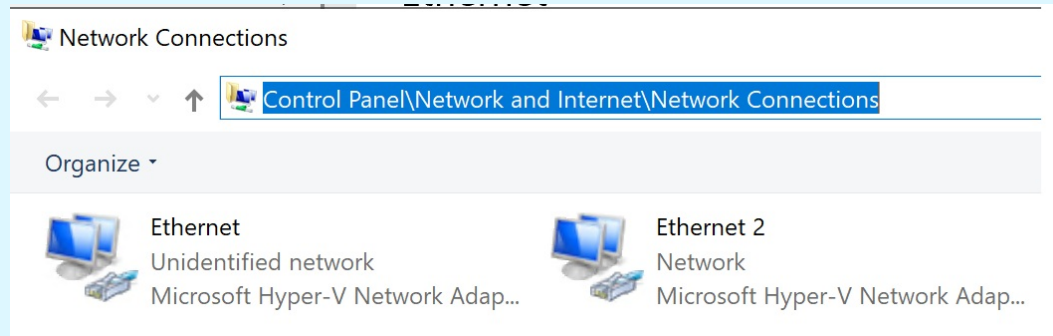
{height="" width=""}

2a. If you get the following error then follow steps after error:

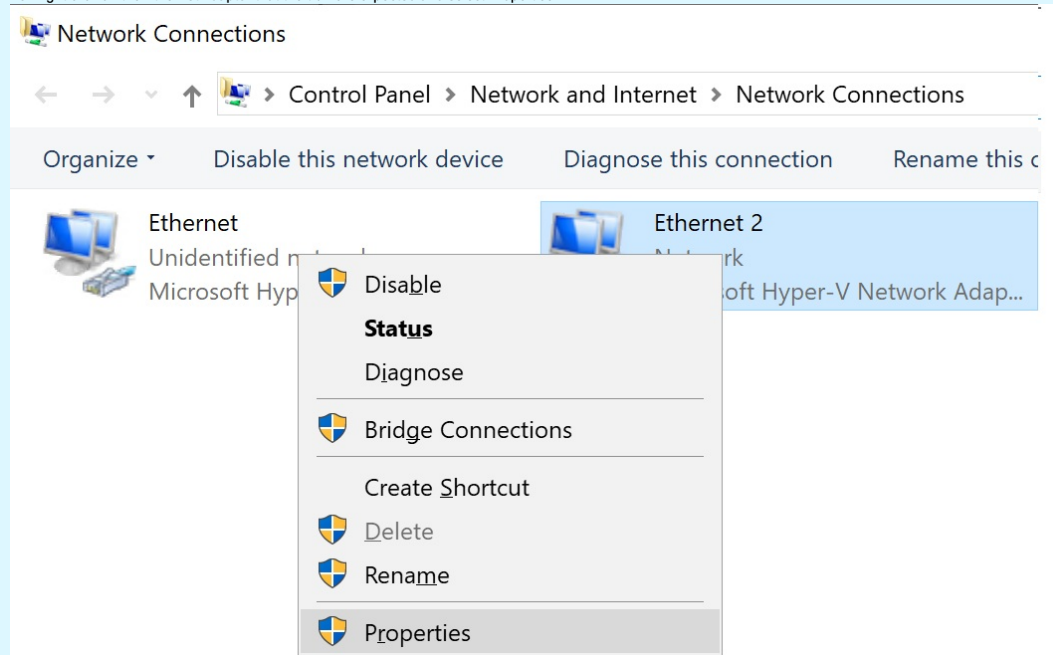
```
None Copy
```

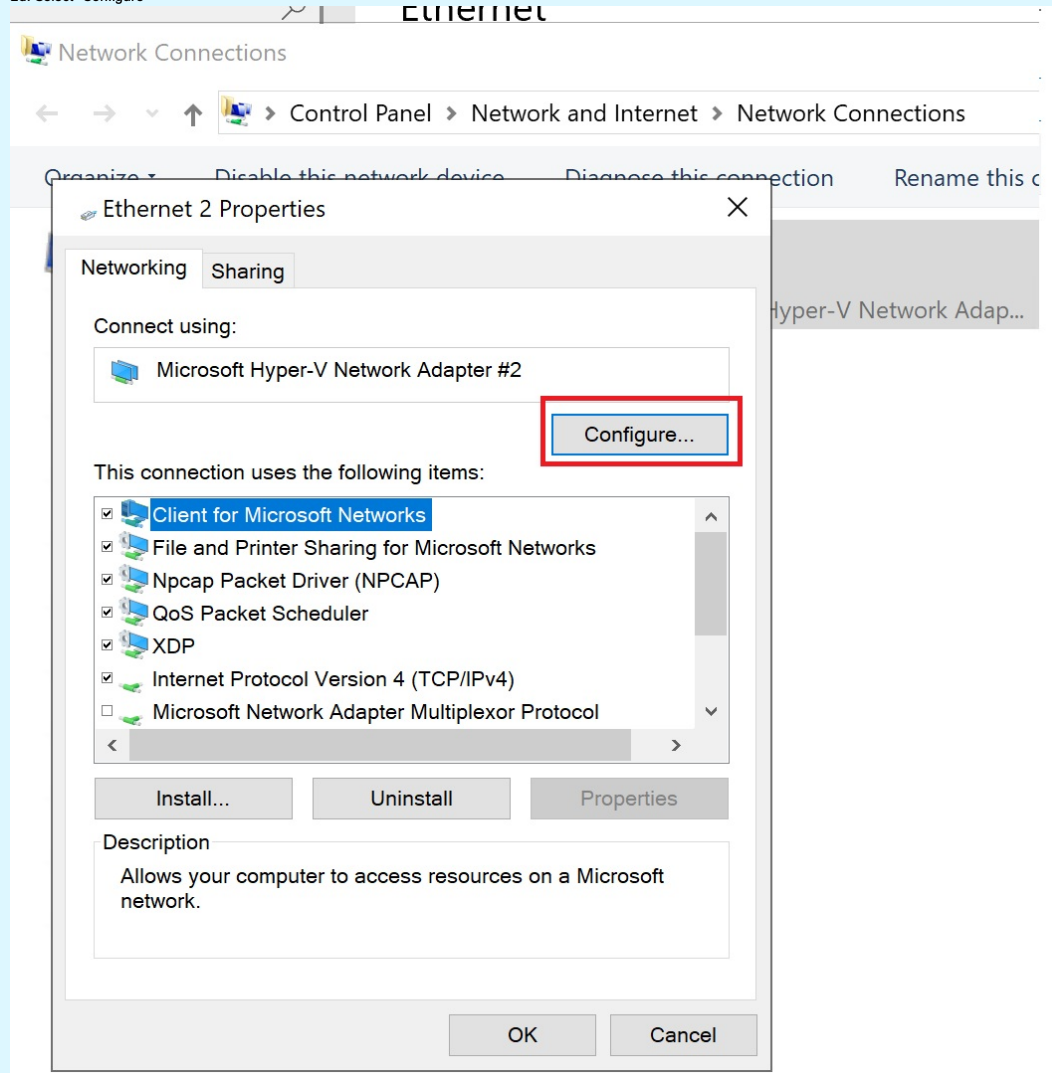
```
netsh interface ipv4>set subinterface "Ethernet 2" mtu=2000 store=persistent
The parameter is incorrect.
```

2b. Go to the Network connection → "Control Panel\Network and Internet\Network Connections"

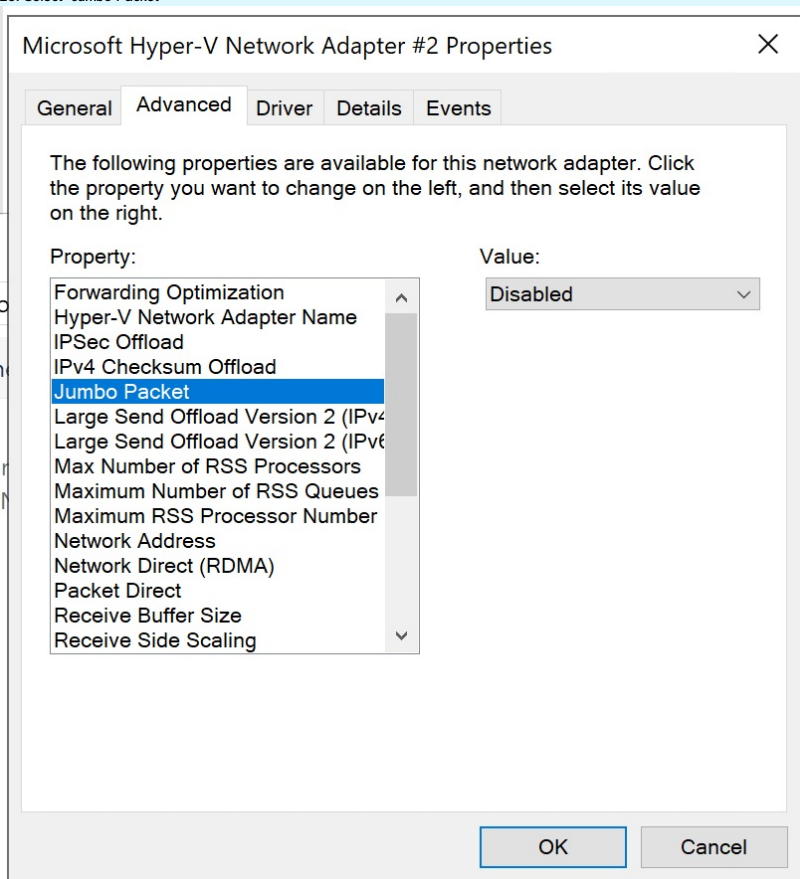


2c. Right click on the Ethernet Adaptor that the traffic is expected and select Properties.

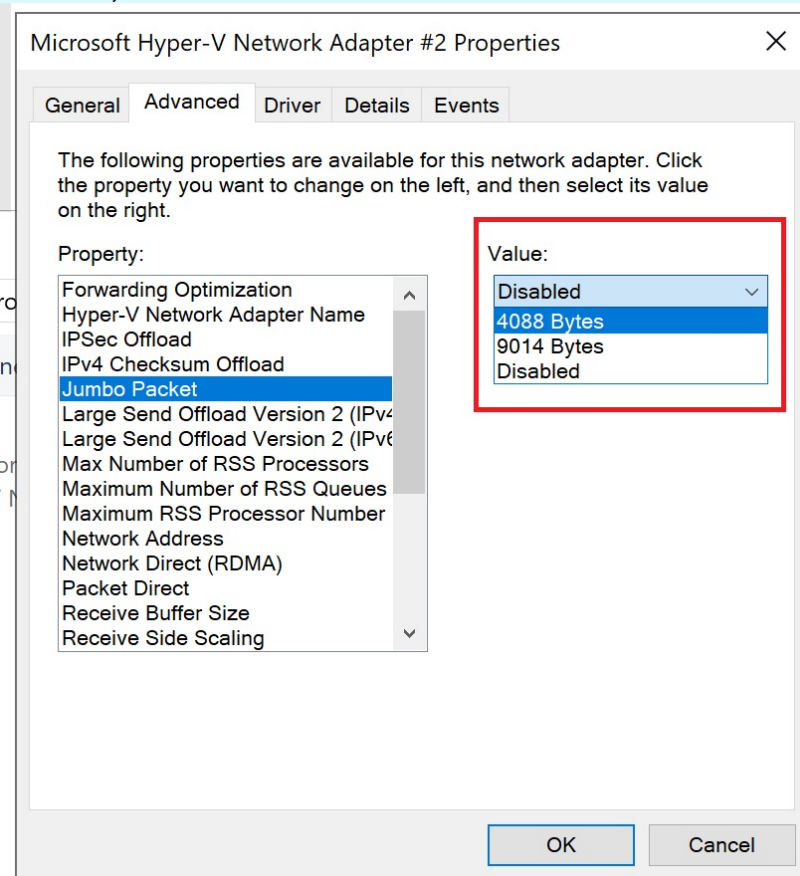




2e. Select "Jumbo Packet"



2f. Select "4088 Bytes" then select "OK".



3. Re-run step 2 to set MTU size

5. Reboot your computer

6. Re-run step 1 to validate the MTU size is correct

How to Peer between VPCs in Different Regions for AWS

WHAT TO EXPECT

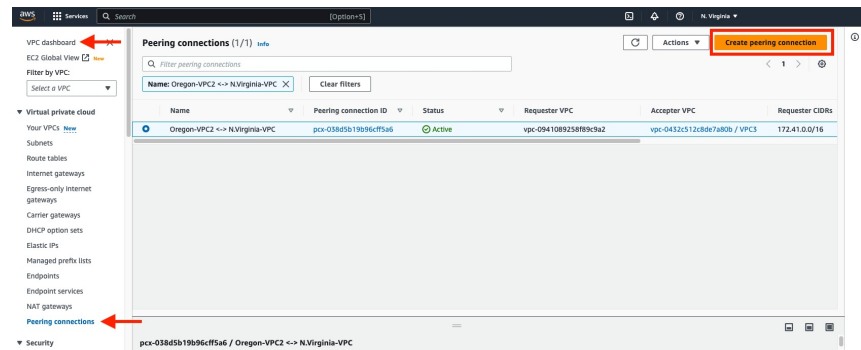
In order to successfully do Peering Connections between VPCs in different regions on AWS, a user must configure their route tables to allow traffic between instances. This will ensure that packets destined for a specific network segment on the other region/VPC/subnet are correctly routed.

In this article, users will learn how to [Create a Peering Connection between Different Regions](#), [Modify Route Tables](#) and [Edit Subnet Associations](#). Step 2 and 3 of the process will need to be repeated for both regions.

STEP ONE: Create a Peering Connection between Different Regions

1. Go to the VPC Dashboard and select Peering Connections.

2. Click Create peering connection.



3. Edit the following in the Create peering connection form:

a. Set a descriptive name. In the example, the user lists the connection between VPCs from Oregon and N. Virginia.

b. Select the VPC of the instance you want to connect from.

c. Select **Another Region** and select the destination region from the dropdown menu.

d. Enter the VPC ID of the target VPC in the target region.

e. Add any tag needed for organization purposes.

Create peering connection

A VPC peering connection is a networking connection between two VPCs that enables you to route traffic between them privately.

Peering connection settings

Name - optional

Create a tag with a key of 'Name' and a value that you specify.

Oregon-VPC2 <-> N.Virginia-VPC

Select a local VPC to peer with

VPC ID (Requester)

vpc-0432c512c8de7a80b (VPC1)

VPC CIDRs for vpc-0432c512c8de7a80b (VPC1)

CIDR

Status

Status reason

172.31.0.0/16

Associated

-

Select another VPC to peer with

Account

My account

Another account

Region

This Region (us-east-1)

Another Region

US West (Oregon) (us-west-2)

VPC ID (Acceptor)

vpc-0941089258f89c9a2

Tags

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key

Value - optional

O Name

O Oregon-VPC2 <-> N.Virginia-VPC

Remove

Add new tag

You can add 49 more tags.

Cancel

Create peering connection

4. Click **Create peering connection**. A new Peering Connection should now be listed for the region you're on. **Please note:** A "mirrored connection" will be created on the "destination" region. It must be accepted manually to be active.

5. Change to the other region.

©2024 IEX Group, Inc. and its subsidiaries, including swXtch.io. All rights reserved.

261

- Go to **Peering Connections**.
- Select the new **Peering Connection** listed as "Pending acceptance."

Peering connections (1/5) Info

Filter peering connections

Name	Peering connection ID	Status	Requester VPC	Accepter VPC	Requester CIDRs	Accepter CIDRs	Requester owner ID	Accepter owner ID
RegionalTestV...	pcx-0cfd2bf57c419028	Active	vpc-019af57e1ad2e78a0	vpc-032478a302937fd9e / SA...	172.31.0.0/16	172.31.0.0/16	639720666639	639720666639
~	pcx-012996f2cfff6d9c	Pending acceptance	vpc-0dd3720d5088eadf6	vpc-0941089258f89c9a2 / SA...	172.31.0.0/16	~	639720666639	639720666639
Oregon-VPC2 ...	pcx-0fd12221d86e30677	Active	vpc-0941089258f89c9a2 / SA...	vpc-0432c512c8de7a80b	172.41.0.0/16	172.31.0.0/16	639720666639	639720666639
MLP-VPC-Peer	pcx-06d570c3874f620f1	Active	vpc-093b945e59a8ef1a8 / SA...	vpc-0941089258f89c9a2 / SA...	172.52.0.0/16	172.41.0.0/16	639720666639	639720666639
SATest-1-and-...	pcx-098b8f5b53c70c920	Active	vpc-032478a302937fd9e / SA...	vpc-0941089258f89c9a2 / SA...	172.31.0.0/16	172.41.0.0/16	639720666639	639720666639

- Under the Actions dropdown, select **Accept request**.

Peering connections (1/5) Info

Filter peering connections

Name	Peering connection ID	Status	Requester VPC	Accepter VPC	Requester CIDRs	Accepter CIDRs	Requester owner ID	Accepter owner ID
RegionalTestV...	pcx-0cfd2bf57c419028	Active	vpc-019af57e1ad2e78a0	vpc-032478a302937fd9e / SA...	172.31.0.0/16	172.31.0.0/16	639720666639	639720666639
~	pcx-012996f2cfff6d9c	Pending acceptance	vpc-0dd3720d5088eadf6	vpc-0941089258f89c9a2 / SA...	172.31.0.0/16	~	639720666639	639720666639
Oregon-VPC2 ...	pcx-0fd12221d86e30677	Active	vpc-0941089258f89c9a2 / SA...	vpc-0432c512c8de7a80b	172.41.0.0/16	172.31.0.0/16	639720666639	639720666639
MLP-VPC-Peer	pcx-06d570c3874f620f1	Active	vpc-093b945e59a8ef1a8 / SA...	vpc-0941089258f89c9a2 / SA...	172.52.0.0/16	172.41.0.0/16	639720666639	639720666639
SATest-1-and-...	pcx-098b8f5b53c70c920	Active	vpc-032478a302937fd9e / SA...	vpc-0941089258f89c9a2 / SA...	172.31.0.0/16	172.41.0.0/16	639720666639	639720666639

STEP TWO: Modify Route Tables in Both Regions

Once the peering connections are created, the route table must be modified in both regions. Start with the 1st region and complete STEP TWO and STEP THREE.

- Go to the **VPC Dashboard**.
- Click on **Route tables** in the Virtual private cloud section.
- Select **Create route table** button.

VPC dashboard

EC2 Global View New

Filter by VPC: Select a VPC

Virtual private cloud

Your VPCs New

Subnets

Route tables

Internet gateways

Egress-only internet

Route tables (5) Info

Find resources by attribute or tag

Name	Route table ID	Explicit subnet associations	Edge associations	Main	VPC	Owner ID
Private subnet routes for VPC1	rtb-0b9b6dd248a3db666	-	-	No	vpc-0432c512c8de7a80b VPC1	639720666639
RegionalTestVPC-rt	rtb-06b64a11e29effed5	2 subnets	-	No	vpc-019af57e1ad2e78a0 RegionalTestVPC	639720666639
Public subnet routes for VPC1	rtb-0f73d496d1bed158c	subnet-01aaca91957af72d7 / VPC1-public	-	Yes	vpc-0432c512c8de7a80b VPC1	639720666639
Oregon-VPC2 <-> N.Virginia-VPC	rtb-016f9783e7c943796	2 subnets	-	No	vpc-0432c512c8de7a80b VPC1	639720666639
RegionalTestVPC-public-rtb	rtb-0d5c6db021244e9ec	subnet-058d27396c0dc1e08 / Public-subnet	-	Yes	vpc-019af57e1ad2e78a0 RegionalTestVPC	639720666639

- Edit the following in the **Create route table** form:
 - Enter a descriptive name.
 - Select the correct VPC.
 - Add any necessary tags.

VPC > Route tables > Create route table

Create route table Info

A route table specifies how packets are forwarded between the subnets within your VPC, the internet, and your VPN connection.

Route table settings

Name - optional
Create a tag with a key of 'Name' and a value that you specify.

my-route-table-01

VPC
The VPC to use for this route table.

vpc-0432c512c8de7a80b (VPC1)

Tags
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

No tags associated with the resource.

Add new tag

You can add 50 more tags.

Cancel Create route table

- Select **Create route table**.
- Select the **Route table ID** of the route table you just created.

Route tables (1/5) [Info](#)

Find resources by attribute or tag

	Name	Route table ID
<input type="checkbox"/>	Private subnet routes for VPC1	rtb-0b9bdd248a3dbe66
<input type="checkbox"/>	RegionalTestVPC-rt	rtb-06b64a11e29effed5
<input type="checkbox"/>	Public subnet routes for VPC1	rtb-0f73d496d1bed158c
<input checked="" type="checkbox"/>	Oregon-VPC2 <-> N.Virginia-VPC	rtb-016f9783e7c943796
<input type="checkbox"/>	RegionalTestVPC-public-rtb	rtb-0d6c6db021244e9ec

7. Select **Edit routes** button the next screen.

The screenshot shows the AWS Management Console interface for a VPC. The left sidebar contains navigation links for VPC dashboard, EC2 Global View, and various VPC resources. The main content area displays the 'Route tables' page for 'rtb-08dd6801fcf231277 / Oregon-VPC2<->N.Virginia-VPC3-data'. The 'Routes' tab is selected, showing a table with one route. The 'Edit routes' button is highlighted with a red box.

8. Add the **Destination** by entering the CIDR of the destination network.

9. Under **Target**, select the recently created **Peering Connection** from the list.

The screenshot shows the 'Edit routes' form in the AWS Management Console. The form has a table with columns: Destination, Target, Status, and Propagated. There are two rows of routes. The first row has Destination '172.31.0.0/16' and Target 'local'. The second row has Destination '172.41.0.0/16' and Target 'pcx-0fd12221d86e30677'. The 'Add route' button is at the bottom left. The 'Save changes' button is at the bottom right.

10. Click the **Save changes** button.

Internet Access

If you need the agents to have access to the internet, you will also need to add the route for the 0.0.0.0/0 towards the NAT gateway.

STEP THREE: Edit Subnet Associations

1. Select **Subnet associations** tab.

2. Select **Edit subnet associations** button under the **Explicit subnet associations** box.

The screenshot shows the AWS Management Console interface for a VPC. The left sidebar contains navigation links for VPC dashboard, EC2 Global View, and various VPC resources. The main content area displays the 'Route tables' page for 'rtb-08dd6801fcf231277 / Oregon-VPC2<->N.Virginia-VPC3-data'. The 'Subnet associations' tab is selected, showing a table with one row. The 'Edit subnet associations' button is highlighted with a red box.

3. Select the subnet(s) of the instance that must be connected to the destination.

VPC > Route tables > rtb-016f9783e7c943796 > Edit subnet associations

Edit subnet associations

Change which subnets are associated with this route table.

Available subnets (2/8)

Filter subnet associations

< 1 > ⌕

<input type="checkbox"/>	Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route table ID
<input type="checkbox"/>	Default VPC1-d	subnet-0d5aac918668cedbe	172.31.80.0/20	-	Main (rtb-0f73d496d1bed158c / Public subnet routes for VPC1)
<input type="checkbox"/>	Default VPC1-f	subnet-0d9977ffb242c0086	172.31.64.0/20	-	Main (rtb-0f73d496d1bed158c / Public subnet routes for VPC1)
<input type="checkbox"/>	VPC1-public	subnet-01aaca91957af72d7	172.31.112.0/24	-	rtb-0f73d496d1bed158c / Public subnet routes for VPC1
<input checked="" type="checkbox"/>	VPC1-ctrl-subnet	subnet-03de822f9248b91ff	172.31.16.0/20	-	rtb-016f9783e7c943796 / Oregon-VPC2 <-> N.Virginia-VPC
<input checked="" type="checkbox"/>	VPC1-data-subnet	subnet-01df022a4f373fa17	172.31.96.0/20	-	rtb-016f9783e7c943796 / Oregon-VPC2 <-> N.Virginia-VPC
<input type="checkbox"/>	Default VPC1-e	subnet-068d4969f6ec1457a	172.31.48.0/20	-	Main (rtb-0f73d496d1bed158c / Public subnet routes for VPC1)
<input type="checkbox"/>	Default VPC1-b	subnet-06e372352e6a55935	172.31.32.0/20	-	Main (rtb-0f73d496d1bed158c / Public subnet routes for VPC1)
<input type="checkbox"/>	Default VPC1-c	subnet-0c633106e2e3bc850	172.31.0.0/20	-	Main (rtb-0f73d496d1bed158c / Public subnet routes for VPC1)

Selected subnets

subnet-01df022a4f373fa17 / VPC1-data-subnet X subnet-03de822f9248b91ff / VPC1-ctrl-subnet X

Cancel Save associations

4. Click the **Save associations** button.

Security Groups

It is important that security groups on each EC2 and on each Subnet on both Regions match and should both encompass the port exceptions listed in the [cloudSwXtch System Requirements](#) article.

Repeat STEP TWO and THREE for the Other Region

How to Modify CPU Core Usage in Icore for Large Instances

WHAT TO EXPECT

There is a known issue where users with large instance types experience reduced performance. To solve this issue, users can manually modify their CPU core usage to ensure an optimal thread configuration.

Prerequisites

- Aim to use only one CPU per core to maximize efficiency.
- Restrict usage to CPUs from the same socket, most likely from SOCKET 0.
- Try not to use more than 14 CPUs as listed in the Icores setting.

If you have any questions or concerns, please contact support@swtch.io **before** attempting reconfiguration.

1. Check your core topology by using the following command on your cloudSwXtch VM:

PowerShellCopy

```
lscpu --extend --all
```

Example Output:

PowerShellCopy

```
CPU NODE SOCKET CORE L1d:L1i:L2:L3 ONLINE    MAXMHZ   MINMHZ
  0    0      0      0 0:0:0:0      yes 2800.0000 800.0000
  1    0      0      0 0:0:0:0      yes 2800.0000 800.0000
  2    0      0      1 1:1:1:0      yes 2800.0000 800.0000
  3    0      0      1 1:1:1:0      yes 2800.0000 800.0000
  4    0      0      2 2:2:2:0      yes 2800.0000 800.0000
  5    0      0      2 2:2:2:0      yes 2800.0000 800.0000
  6    0      0      3 3:3:3:0      yes 2800.0000 800.0000
  7    0      0      3 3:3:3:0      yes 2800.0000 800.0000
```

2. Adjust your thread assignment by editing the `/swxtch/Icores.json` configuration file. This file specifies which CPU your application should use.

This is an example of the format you can use in Icores.json:

PowerShellCopy

```
{
  "Icores": "1,3,4,6,8-12"
}
```

Media Use Cases

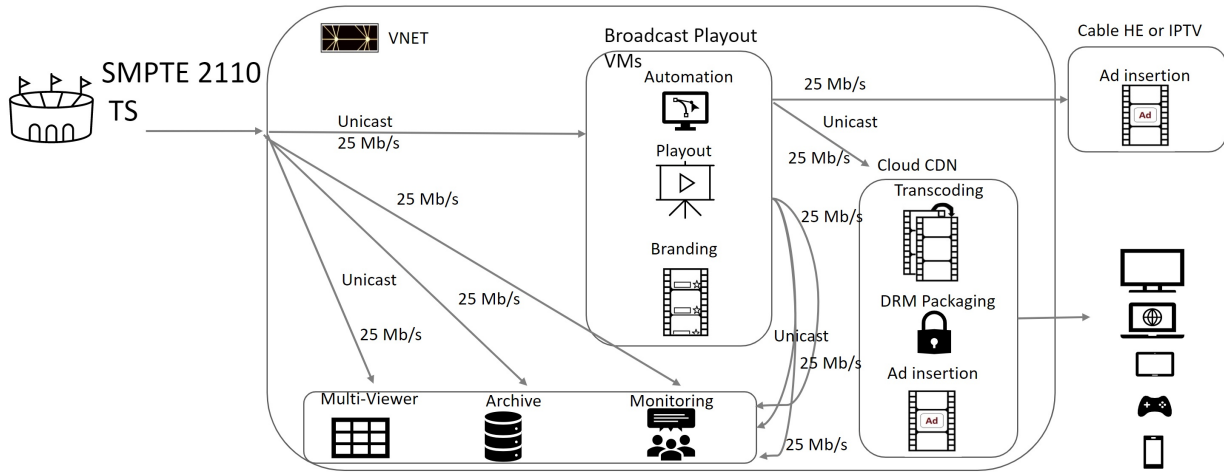
The media market can take advantage of several cloudSwXtch features such as:

- [Multicast](#)
- [Hitless Merge](#)
- [Compression support](#)
- [Protocol Fanout](#)
- [Disaster Recovery](#)

Media Multicast made easy with cloudswXtch

Media companies want to build dynamic workflows on the cloud, but clouds only support unicast workflows. This makes media workflows cumbersome as each stream would need to be configured for each receiver. Network provisioning and administration is complex, distributed, difficult to modify and must be replicated for every workflow as shown below:

Unicast Playback in cloud without cloudSwXtch

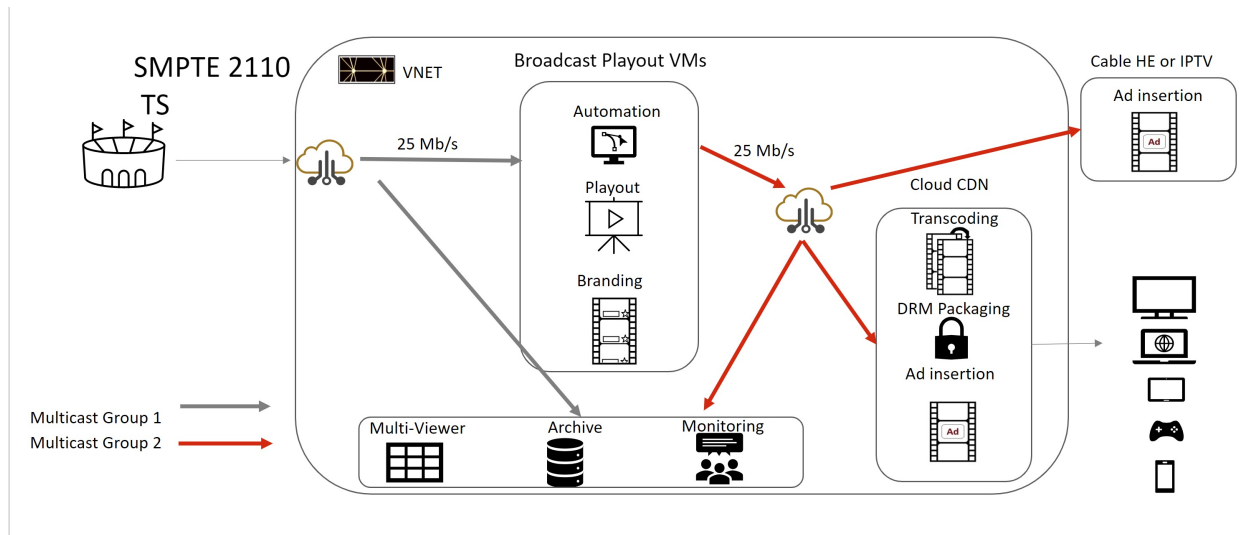


With unicast there are a number of issues:

- Network provisioning and administration is complex, distributed, difficult to modify and must be replicated for each channel or workflow
- The users cannot add endpoints without reconfiguring servers
- Larger VMs are required to support unicast which equates to higher cloud costs.
- Disaster Recovery is difficult to execute
- The load to the network is much larger
- SMPTE 2110 - 100+x more bandwidth

Multicast Playback in cloud with cloudSwXtch Multicast

cloudSwXtch enables true and seamless IP-multicast. Using multicast instead of unicast optimizes your network configuration and reduces your cloud distribution and egress costs. In addition, receivers can dynamically subscribe and unsubscribe to your streams as workflows dictate. cloudSwXtch eliminates having to configure and unconfigure unicast streams to accommodate configuration changes.




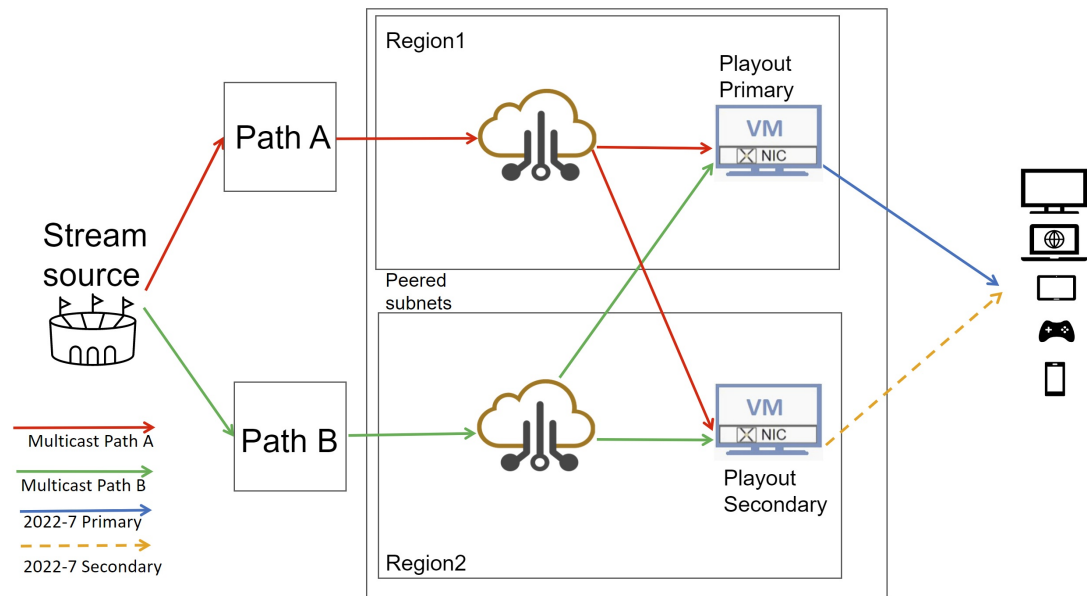
With cloudSwXtch Multicast:

- Network may be modified and extended simply by joining multicast groups, with powerful centralized control and monitoring.
- Users can dynamically add new endpoints without playback server (or any other workflows/products) involvement.
- VM Sizes are minimized to workflow/product needs
- Disaster recovery is easy to set-up
- Minimal network load

Hitless Merge - 2022-7

It is never good enough to have one broadcast instance, we all know things can and will go wrong. The show must always go on, media companies are used to having primary and backup streams to ensure the best user experience with NO downtime.

 cloudSwXtch SMPTE 2022-7 Hitless Merge protects against data path failures by supporting two or more data paths. It compares packet reception from the multiple streams, detecting dropped packets, and reconstructs the output stream in the correct packet order.

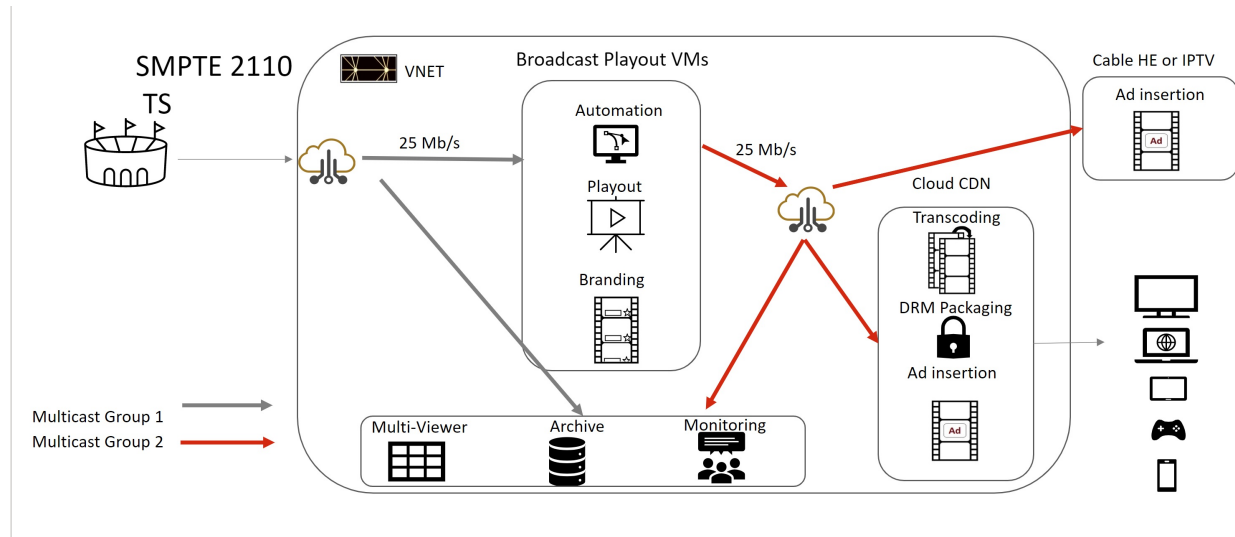


Media support for Compressed and Uncompressed Workflows

At **swXtch.io** we know that the media companies rely highly on both compressed and uncompressed content. **cloudSwXtch** has SMPTE 2110 support without the necessity of additional gateways or other on-ramp/off-ramp appliances. The **cloudSwXtch** architecture is designed to treat content the same whether it is compressed or uncompressed. This means the ingest of streams from on-prem to the cloud and the streaming of content within the cloud, whether unicast or multicast, is the same regardless of the content type. No SDK is required for uncompressed video, and the cloud network becomes an extension of your broadcast network.

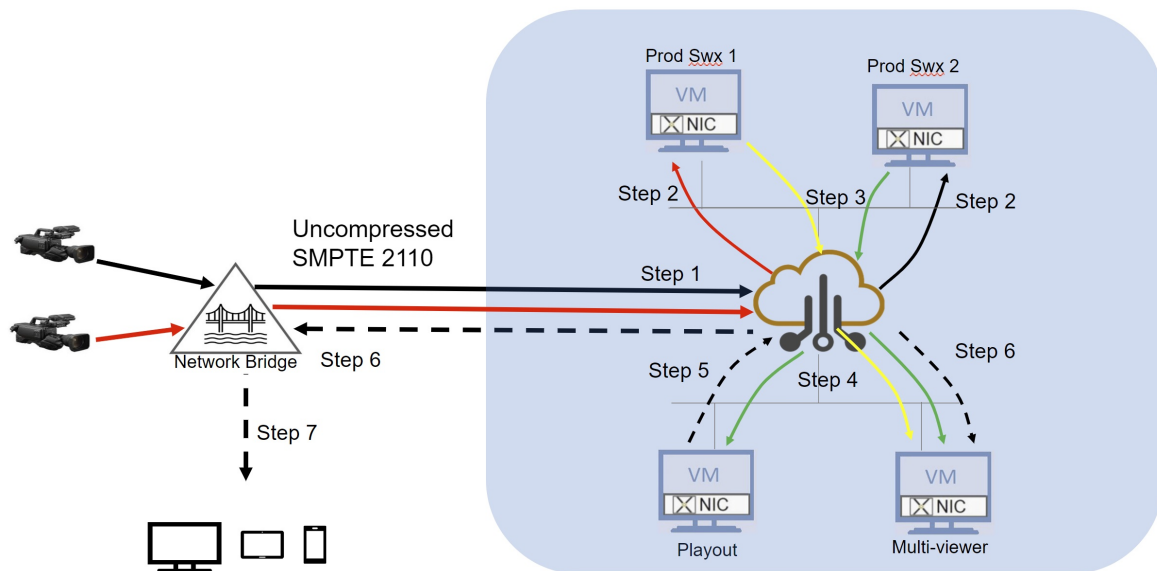
There are two workflow examples below, one is a compressed workflow and the other is an uncompressed workflow. The compressed workflow is a typical playout scenario where compressed inputs come into the cloud environment and are distributed via multicast to the necessary VM workloads by **cloudSwXtch**. All that is required is for the workloads to subscribe to the necessary multicast group(s). This eliminates the need to continually update unicast configurations to ensure your streams get to where they need to go. However, if there are workloads that only work with unicast, **cloudSwXtch** can map multicast streams to unicast devices.

Example Compressed Playout in the Cloud with SMPTE 2110 Multicast TS



Example Uncompressed Playout in the Cloud with SMPTE 2110 Multicast

Consider the following production workflow:



The workflow consists of a playout server which receives multiple camera feeds via 2 production switchers and determines which camera's to take to air. The **cloudSwXtch** is used to deliver the various streams via multicast to the workloads that subscribe to the stream:

Step 1: Two inputs red and black go from Network Bridge into **cloudSwXtch**.

Step 2: Red stream goes from **cloudSwXtch** to Production Switcher 1 and black stream goes to production switcher 2.

Step 3: The modified output stream from production switcher 1 is represented by the yellow path and the modified output stream from production switcher 2 is represented by the green path to the **cloudSwXtch**.

Step 4: All streams are multicasted to the multiviewer, via **cloudSwXtch**, so the director can make operational decisions.


Step 5: The playout server is directed to process and output one of the switcher outputs as represented by the dotted black to the **cloudSwXtch**.

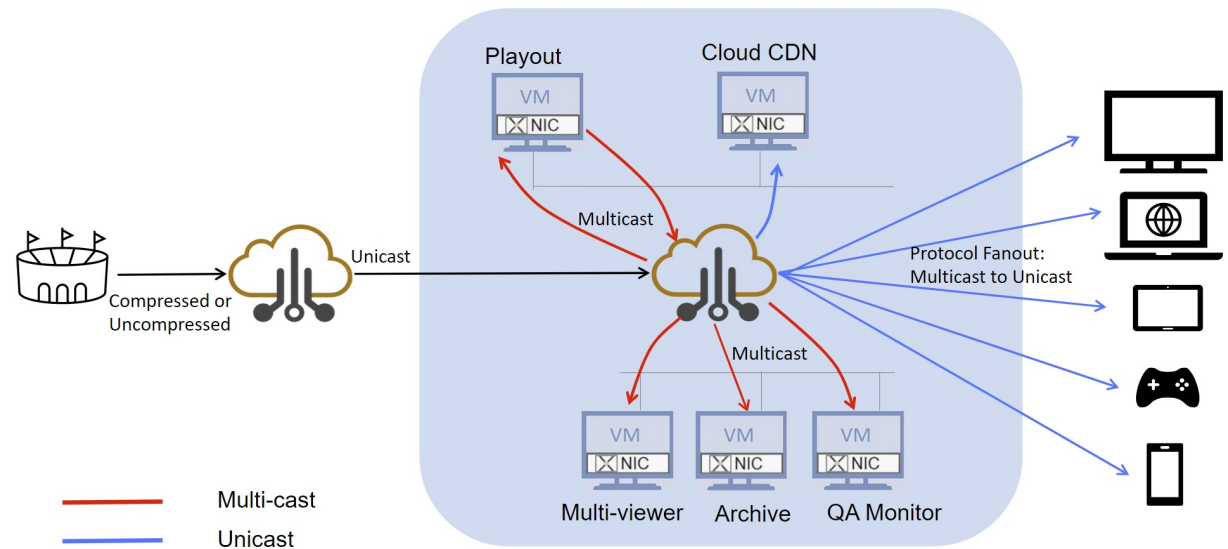
Step 6: **cloudSwXtch** outputs the stream to the multiviewer, and the network bridge.

Step 7: The network bridge distributes to the clients for viewing consumption.

Protocol Fanout

****Media companies have many devices. Some require unicast, and some require multicast. Configuring for each device can be difficult and supporting both unicast and multicast for the same stream is impossible. Additionally multicast is not offered in the cloud see .

 swXtch.io has the answer to your needs with the 'Protocol FanOut' feature which can take non-multicast packet protocols and fan them out in the same way that multicast does. It can forward a stream to many interested receivers or distribute a multicast stream to many unicast devices. This integrates unicast and multicast workflows in a way that hasn't been possible in the cloud.



Disaster Recovery

Disaster Recovery Scenerio

Coupling [Hitless Merge - 2022-7](#) with redundant media workloads ensures high availability uptime for critical content and provides a new method to create highly available disaster recovery pathways in and between clouds.

There are many configurations that **cloudSwXtch** can recommend for redundancy, one is depicted below.

Path Redundancy

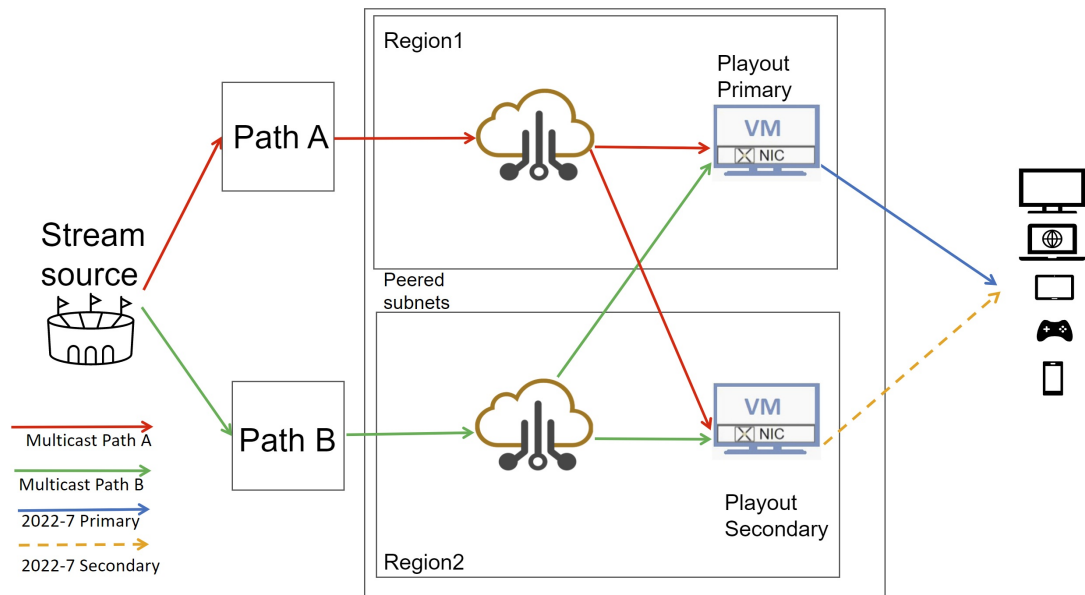
- The **cloudswXtch** in Region 1 can recieve the stream from Path A to Region 2.
- The **cloudswXtch** in Region 2 can recieve the stream from Path B to Region 1.
- If either path were to fail then the stream is still available in both Region 1 and 2 due to the redundancy.

Playout Redundancy

- Each Region has a playout system, "Primary Playout" in Region 1 and "Secondary Playout" in Region 2.
- If the "Primary Playout" should fail, the stream is still playing out in the "Secondary Playout".
- As long as it is just the playout server that fails, then there is still stream redundancy from Path A and Path B.

Region Redundancy

If one region should fail the playout should still succeed in the other Region.



This depiction only shows two stream paths, there could be a third or more. In any of these scenarios the paths could be in different regions or different clouds. This is done by using a **cloudSwXtch** as a [Bridge](#) between clouds or from on-prem to cloud.

Monitoring API

Overview

The swXtch.io Monitoring API allows the integration of data from various cloudSwXtch component types into third party tools for monitoring and topology purposes within a customer’s user interfaces. This section will outline the API, with examples of data results. **Unless otherwise noted, these API calls are only applicable to cloudSwXtch versions 3.0.0 or greater.**

Prerequisites

- A cloudSwXtch must exist as well as two or more agents with xNICs.
- Traffic (multicast, broadcast, SRT, RIST, or unicast) should be flowing between the cloudSwXtch and the endpoints.
 - **Note:** SRT, RIST and unicast require additional configuration on the cloudSwXtch.

By using a GET command, data will be provided in the response.

swXtch.io API Concepts Explained

Before you start, there are a number of key concepts and terms that are important to understand before diving into the complexities of the monitoring and topology API.

Source Address vs. Adjacent Address

In the example responses, users will notice both an source and adjacent address. A **source address** represents a producer’s control NIC. This is where the stream originated from.

The **adjacent address** represents the data NIC of either the producer/consumer that is sending out or receiving multicast data. In the monitoring API, this value is dependent on whether it is in the rxStreamLinks or the txStreamLinks section of the response.

- Under rxStreamLinks, the adjacent address is the IP address of the producer — an endpoint **sending traffic to** the cloudSwXtch
- Under txStreamLinks, the adjacent address is the IP address of the consumer — an endpoint **receiving traffic from** the cloudSwXtch.

In the topology API, this value is dependent on whether it is in the ingress or egress streamLinks section.

- Under **ingress**, the adjacent address is the IP address of the producer — an endpoint **sending traffic to** the cloudSwXtch
- Under **egress**, the adjacent address is the IP address of the consumer — an endpoint **receiving traffic from** the cloudSwXtch.

Protocol Types

In the streamLinks section of the cloudSwXtch stats example response, the API specifies the protocol of the stream.

- **xMC**: multicast or broadcast
- **SRT (C)**: SRT Caller
- **SRT (L)**: SRT Listener
- **RIST (C)**: RIST Caller
- **RIST (L)**: RIST Listener

Timestamps

To track time as a running total of seconds, the Timestamps are in Unix Timestamp. This count starts at the Unix Epoch on January 1st, 1970, at UTC. At any point in time, the API can be run, and certain metrics can be obtained from the response payload by calculating certain counter and timestamp Delta values.

Monitoring API

The monitoring API allows developers to get data from the cloudSwXtch as well as data about its xNIC’s. This data is broken down into 5 categories:

- [Topology](#)
- [Cloud information](#)
- [cloudSwXtch information](#)
- [cloudSwXtch Stats](#)
- [xNIC Totals](#)
- [xNIC Stats](#)

Topology

GET

/api/topology/v1/topology

- Get information about the components and streamLinks in the Topology Graph

URL:

PowerShell

Copy

http://<cloudSwXtch-control-IP>/api/topology/v1/topology

Example URL:

http://10.2.128.10/api/topology/v1/topology

Request:

Empty

Response:

200 - successful response

Example Response →

PowerShell

Copy

```
{
  "components": {
    "10.2.128.10": {
      "general": {
        "id": "10.2.128.10",
        "name": "dsd-core-100",
        "componentKind": {
          "code": "swxtch",
          "displayName": "swXtch"
        }
      },
      "environment": {
        "hostname": "dsd-core-100",
        "cloud": "AZURE",
        "osDistribution": "Ubuntu 20.04",
        "region": "eastus",
        "instanceType": "Standard_D8as_v4"
      },
      "hardware": {
        "nics": {
          "eth0": {
            "name": "eth0",
            "index": 2,
            "ip": "10.2.128.10",
            "subnetPrefix": "10.2.128.0/22",
            "subnetMask": "255.255.252.0",
            "mtu": 1500,
            "mac": "00:0d:3a:19:cc:e1",
            "broadcastIp": "10.2.131.255",
            "driver": "hv_netvsc",
            "pciAddress": "",
            "publicIp": null,
            "masterOf": null,
            "vpc": null
          },
          "eth1": {
            "name": "eth1",
            "index": 3,
            "ip": "10.2.192.116",
            "subnetPrefix": "10.2.192.0/22",
            "subnetMask": "255.255.252.0",
            "mtu": 1500,
            "mac": "00:0d:3a:54:0f:37",
            "broadcastIp": "10.2.195.255",
            "driver": "mlx5_core",
            "pciAddress": "a6bd:00:02.0",
            "publicIp": null,
            "masterOf": "enP42685s2",
            "vpc": null
          }
        },
        "vpcs": {}
      },
      "swxGeneral": {
        "type": "X1",
        "dataInterfaceName": "eth1",
        "controlInterfaceName": "eth0",
        "dataPort": 9999,
        "version": "dev.7f1520",
        "configuration": null,
        "services": null,
        "subscriptions": null
      },
      "parentId": null,
      "timing": {
        "Master": null,
        "LocalOffset": null,
        "RootOffset": null,
        "TimebeatPresent": null
      },
      "adaptors": [
        {
          "id": "16704273857340480058",
          "isConnected": false,
```



```

        "isActive": false,
        "alive": true,
        "clientIp": "0.0.0.0",
        "clientPort": 0,
        "sourceIp": "0.0.0.0",
        "sourcePort": 0,
        "socketIp": "0.0.0.0",
        "socketPort": 0,
        "timestamp": 1724082856715428500,
        "counters": {
            "txPackets": 0,
            "txBytes": 0,
            "rxPackets": 0,
            "rxBytes": 0,
            "txLostPackets": 0,
            "txLostBytes": 0,
            "rxLostPacket": 0,
            "rxLostBytes": 0,
            "txRetransmittedPackets": 0,
            "txRetransmittedBytes": 0,
            "rxRetransmittedPackets": 0,
            "rxRetransmittedBytes": 0
        },
        "info": {
            "id": "16704273857340480058",
            "protocol": "srt-caller",
            "direction": "egress",
            "streamIp": "239.4.2.3",
            "streamPort": 5400,
            "nodeIp": "10.2.128.37",
            "nodePort": 5401,
            "listenerPort": null,
            "options": []
        }
    },
    {
        "id": "4777854487999137884",
        "isConnected": false,
        "isActive": false,
        "alive": true,
        "clientIp": "0.0.0.0",
        "clientPort": 0,
        "sourceIp": "0.0.0.0",
        "sourcePort": 0,
        "socketIp": "0.0.0.0",
        "socketPort": 0,
        "timestamp": 1724082856715423300,
        "counters": {
            "txPackets": 0,
            "txBytes": 0,
            "rxPackets": 0,
            "rxBytes": 0,
            "txLostPackets": 0,
            "txLostBytes": 0,
            "rxLostPacket": 0,
            "rxLostBytes": 0,
            "txRetransmittedPackets": 0,
            "txRetransmittedBytes": 0,
            "rxRetransmittedPackets": 0,
            "rxRetransmittedBytes": 0
        },
        "info": {
            "id": "4777854487999137884",
            "protocol": "srt-caller",
            "direction": "ingress",
            "streamIp": "239.4.2.3",
            "streamPort": 5400,
            "nodeIp": "10.2.128.36",
            "nodePort": 5400,
            "listenerPort": null,
            "options": []
        }
    }
],
{
    "id": "5062225067104987852",
    "isConnected": false,
    "isActive": false,
    "alive": true,
    "clientIp": "0.0.0.0",
    "clientPort": 0,
    "sourceIp": "0.0.0.0",
    "sourcePort": 0,

```

```

"socketIp": "0.0.0.0",
"socketPort": 6000,
"timestamp": 1724082856715416700,
"counters": {
  "txPackets": 0,
  "txBytes": 0,
  "rxPackets": 0,
  "rxBytes": 0,
  "txLostPackets": 0,
  "txLostBytes": 0,
  "rxLostPacket": 0,
  "rxLostBytes": 0,
  "txRetransmittedPackets": 0,
  "txRetransmittedBytes": 0,
  "rxRetransmittedPackets": 0,
  "rxRetransmittedBytes": 0
},
"info": {
  "id": "5062225067104987852",
  "protocol": "srt-listener",
  "direction": "egress",
  "streamIp": "225.1.1.1",
  "streamPort": 1599,
  "nodeIp": null,
  "nodePort": null,
  "listenerPort": 6000,
  "options": []
}
},
{
  "id": "9489213241930010873",
  "isConnected": false,
  "isActive": false,
  "alive": true,
  "clientIp": "0.0.0.0",
  "clientPort": 0,
  "sourceIp": "0.0.0.0",
  "sourcePort": 0,
  "socketIp": "0.0.0.0",
  "socketPort": 1599,
  "timestamp": 1724082856715406300,
  "counters": {
    "txPackets": 0,
    "txBytes": 0,
    "rxPackets": 0,
    "rxBytes": 0,
    "txLostPackets": 0,
    "txLostBytes": 0,
    "rxLostPacket": 0,
    "rxLostBytes": 0,
    "txRetransmittedPackets": 0,
    "txRetransmittedBytes": 0,
    "rxRetransmittedPackets": 0,
    "rxRetransmittedBytes": 0
  },
  "info": {
    "id": "9489213241930010873",
    "protocol": "srt-listener",
    "direction": "ingress",
    "streamIp": "225.1.1.1",
    "streamPort": 1599,
    "nodeIp": null,
    "nodePort": null,
    "listenerPort": 1599,
    "options": []
  }
}
},
"stats": {
  "totals": {
    "eth1": {
      "tx": {
        "packets": 9032846,
        "bytes": 1734306432
      },
      "txRate": {
        "packets": 1834,
        "bytes": 352169
      },
      "rx": {
        "packets": 6521926,
        "bytes": 1252209792
      }
    }
  }
}

```

```

    },
    "rxRate": {
      "packets": 914,
      "bytes": 175414
    },
    "txDrops": null,
    "rxDrops": null,
    "timestamp": 1724082855663650000
  }
},
"streamLinks": {
  "eth1": {
    "ingress": {
      "Group = 239.1.1.2:3490 | Source = 10.2.128.15:50226 | Adjacent = 10.2.192.15:50226": {
        "packets": 6521926,
        "bytes": 1252209792,
        "protocol": "xMC",
        "protocolStats": null,
        "timestamp": 1724082855608192400,
        "packetsRate": 921,
        "bytesRates": 176757,
        "haPaths": [
          0
        ],
        "fragStats": null
      },
    },
    "egress": {
      "Group = 239.1.1.2:3490 | Source = 10.2.128.15:50226 | Adjacent = 10.2.192.24:9999": {
        "packets": 6491421,
        "bytes": 1246352832,
        "protocol": "xMC",
        "protocolStats": null,
        "timestamp": 1724082855634894800,
        "packetsRate": 921,
        "bytesRates": 176765,
        "haPaths": [
          0
        ],
        "fragStats": null
      },
      "Group = 239.1.1.2:3490 | Source = 10.2.128.15:50226 | Adjacent = 10.5.2.4:9999": {
        "packets": 2541425,
        "bytes": 487953600,
        "protocol": "xMC",
        "protocolStats": null,
        "timestamp": 1724082855634894800,
        "packetsRate": 921,
        "bytesRates": 176765,
        "haPaths": [
          0
        ],
        "fragStats": null
      }
    }
  }
},
"hsCounters": {
  "eth1": []
},
"hsSummary": {},
"slp": {
  "rxSlpStats": {},
  "txSlpStats": null
}
},
"isRemote": false
},
"10.2.128.15": {
  "general": {
    "id": "10.2.128.15",
    "name": "DSd-agent-105",
    "componentKind": {
      "code": "xnic",
      "displayName": "xNIC"
    }
  },
  "environment": {
    "hostname": "DSd-agent-105",
    "cloud": "AZURE",
    "osDistribution": "Windows Server 2019 Datacenter - Microsoft Windows [Version 10.0.17763.6189]",
    "region": "eastus",

```

```

    "instanceType": "Standard_D8s_v4"
  },
  "hardware": {
    "nics": {
      "Ethernet": {
        "name": "Ethernet",
        "index": 415,
        "ip": "10.2.192.15",
        "subnetPrefix": "10.2.192.0/22",
        "subnetMask": "255.255.252.0",
        "mtu": 4074,
        "mac": "00:22:48:1e:61:cb",
        "broadcastIp": "10.2.195.255",
        "driver": "netvsc.sys",
        "pciAddress": "34454:00:02.0",
        "publicIp": null,
        "masterOf": "Ethernet 626",
        "vpc": null
      },
      "Ethernet 2": {
        "name": "Ethernet 2",
        "index": 494,
        "ip": "10.2.128.15",
        "subnetPrefix": "10.2.128.0/22",
        "subnetMask": "255.255.252.0",
        "mtu": 4074,
        "mac": "00:22:48:1e:6b:5b",
        "broadcastIp": "10.2.131.255",
        "driver": "netvsc.sys",
        "pciAddress": "53755:00:02.0",
        "publicIp": null,
        "masterOf": "Ethernet 625",
        "vpc": null
      }
    },
    "vpcs": {}
  },
  "swxGeneral": {
    "type": "2",
    "dataInterfaceName": "Ethernet",
    "controlInterfaceName": "Ethernet 2",
    "dataPort": 9999,
    "version": "dev.7f1520",
    "configuration": {
      "controlInterface": "Ethernet 2",
      "dataInterface": "Ethernet",
      "dataPlaneSpecs": {
        "bpfPrograms": null,
        "verbosity": 0,
        "virtualInterface": {
          "ip": "172.30.0.15",
          "mtu": 4096,
          "name": "swxtch-tun0",
          "subnet": "255.255.252.0",
          "type": "tun"
        }
      }
    },
    "dataPort": 9999,
    "ha": {
      "bufferSizeInPackets": 131072,
      "maxTimeToBufferPacketsMs": 50,
      "protocol": "swxtch",
      "removeInactiveStreamTimeoutSec": 5
    },
    "nicsConfig": null,
    "overrideSrcIp": false,
    "primarySwxtchAddress": "http://10.2.128.10:80",
    "statsReportWait": 60,
    "streamSpecs": null,
    "subscriptionsPollingIntervalMs": 100,
    "swxtch": "10.2.128.10",
    "xnicRpcPort": 10002,
    "xnicType": 2
  },
  "services": [
    {
      "name": "swXtchNic-control",
      "isRunning": true
    },
    {
      "name": "swXtchNic-data",
      "isRunning": true
    }
  ]
}

```

```

    }
  ],
  "subscriptions": {
    "Ethernet": {
      "224.0.0.251": {
        "filter": "exclude",
        "srcList": []
      },
      "224.0.0.252": {
        "filter": "exclude",
        "srcList": []
      },
      "224.0.1.129": {
        "filter": "exclude",
        "srcList": []
      }
    }
  }
},
"parentId": "10.2.128.10",
"timing": {
  "Master": null,
  "LocalOffset": null,
  "RootOffset": null,
  "TimebeatPresent": null
},
"adaptors": null,
"stats": {
  "totals": {
    "Ethernet": {
      "tx": {
        "packets": 13037688,
        "bytes": 2503236096
      },
      "txRate": {
        "packets": 1105536,
        "bytes": 212262961
      },
      "rx": {
        "packets": 0,
        "bytes": 0
      },
      "rxRate": {
        "packets": 0,
        "bytes": 0
      },
      "txDrops": {
        "packets": 0,
        "bytes": 0
      },
      "rxDrops": {
        "packets": 0,
        "bytes": 0
      },
      "timestamp": 1724082852269631100
    }
  },
  "streamLinks": {
    "Ethernet": {
      "ingress": {},
      "egress": {
        "Group = 239.1.1.2:3490 | Source = 10.2.128.15:50226 | Adjacent = 10.2.192.116:3879": {
          "packets": 6518844,
          "bytes": 1251618048,
          "protocol": "xMC",
          "protocolStats": null,
          "timestamp": 1724082852270144000,
          "packetsRate": 0,
          "bytesRates": 0,
          "haPaths": [
            0
          ],
          "fragStats": null
        },
        "Group = 239.1.1.2:3490 | Source = 10.2.128.15:50226 | Adjacent = 10.5.2.6:3879": {
          "packets": 6518844,
          "bytes": 1251618048,
          "protocol": "xMC",
          "protocolStats": null,
          "timestamp": 1724082852270144000,
          "packetsRate": 0,
          "bytesRates": 0,

```

```

        "haPaths": [
            1
        ],
        "fragStats": null
    }
}
},
"hsCounters": {
    "Ethernet": [
        {
            "streamId": {
                "ip": 4286775818,
                "port": 35328
            },
            "pathStats": [
                {
                    "ingressPackets": 0,
                    "ingressBytes": 0,
                    "ingressPacketsRate": 0,
                    "ingressBytesRate": 0,
                    "missingPackets": 0,
                    "missingPacketsRate": 0,
                    "outputStreamPackets": 0,
                    "pathUsage": 0,
                    "outputStreamPacketsRate": 0
                },
                {
                    "ingressPackets": 1,
                    "ingressBytes": 275,
                    "ingressPacketsRate": 0,
                    "ingressBytesRate": 0,
                    "missingPackets": 0,
                    "missingPacketsRate": 0,
                    "outputStreamPackets": 1,
                    "pathUsage": 0,
                    "outputStreamPacketsRate": 0
                }
            ],
            "enqueueFailurePackets": 0,
            "egressPackets": 1,
            "egressBytes": 275,
            "egressPacketsRate": 0,
            "egressBytesRate": 0,
            "outputStreamLoss": 0,
            "outputStreamLossRate": 0,
            "senderIp": "10.2.128.75"
        },
        {
            "streamId": {
                "ip": 4290970122,
                "port": 35328
            },
            "pathStats": [
                {
                    "ingressPackets": 0,
                    "ingressBytes": 0,
                    "ingressPacketsRate": 0,
                    "ingressBytesRate": 0,
                    "missingPackets": 0,
                    "missingPacketsRate": 0,
                    "outputStreamPackets": 0,
                    "pathUsage": 0,
                    "outputStreamPacketsRate": 0
                },
                {
                    "ingressPackets": 1,
                    "ingressBytes": 275,
                    "ingressPacketsRate": 0,
                    "ingressBytesRate": 0,
                    "missingPackets": 0,
                    "missingPacketsRate": 0,
                    "outputStreamPackets": 1,
                    "pathUsage": 0,
                    "outputStreamPacketsRate": 0
                }
            ],
            "enqueueFailurePackets": 0,
            "egressPackets": 1,
            "egressBytes": 275,
            "egressPacketsRate": 0,
            "egressBytesRate": 0,

```

```

        "outputStreamLoss": 0,
        "outputStreamLossRate": 0,
        "senderIp": "10.2.192.82"
    }
}
],
"hsSummary": {
    "Ethernet": {
        "pathSummaries": [
            {
                "ingressPacketsTotal": 0,
                "ingressBytesTotal": 0,
                "missingPacketsTotal": 0,
                "outputStreamPacketsTotal": 0,
                "ingressPacketsRate": 0,
                "ingressBytesRate": 0,
                "missingPacketsRate": 0,
                "outputStreamPacketsRate": 0,
                "pathUsage": 0
            },
            {
                "ingressPacketsTotal": 2,
                "ingressBytesTotal": 550,
                "missingPacketsTotal": 0,
                "outputStreamPacketsTotal": 2,
                "ingressPacketsRate": 0,
                "ingressBytesRate": 0,
                "missingPacketsRate": 0,
                "outputStreamPacketsRate": 0,
                "pathUsage": 0
            }
        ],
        "enqueueFailurePackets": 0,
        "egressPackets": 2,
        "egressBytes": 550,
        "outputStreamLoss": 0,
        "egressPacketsRate": 0,
        "egressBytesRate": 0,
        "outputStreamLossRate": 0
    }
},
"slp": null
},
"isRemote": false
},
"10.2.128.27": {
    "general": {
        "id": "10.2.128.27",
        "name": "DSd-agent-101",
        "componentKind": {
            "code": "xnic",
            "displayName": "xNIC"
        }
    },
    "environment": {
        "hostname": "DSd-agent-101",
        "cloud": "AZURE",
        "osDistribution": "Ubuntu 20.04",
        "region": "eastus",
        "instanceType": "Standard_D4ds_v4"
    },
    "hardware": {
        "nics": {
            "eth0": {
                "name": "eth0",
                "index": 2,
                "ip": "10.2.128.27",
                "subnetPrefix": "10.2.128.0/22",
                "subnetMask": "255.255.252.0",
                "mtu": 1500,
                "mac": "60:45:bd:d5:0f:e1",
                "broadcastIp": "10.2.131.255",
                "driver": "mlx5_core",
                "pciAddress": "e629:00:02.0",
                "publicIp": null,
                "masterOf": "enP58921s1",
                "vpc": null
            },
            "eth1": {
                "name": "eth1",
                "index": 3,
                "ip": "10.2.192.24",

```

```

        "subnetPrefix": "10.2.192.0/22",
        "subnetMask": "255.255.252.0",
        "mtu": 1500,
        "mac": "60:45:bd:d5:0a:05",
        "broadcastIp": "10.2.195.255",
        "driver": "mlx5_core",
        "pciAddress": "1681:00:02.0",
        "publicIp": null,
        "masterOf": "enP5761s2",
        "vpc": null
    },
    "swxtch-tun0": {
        "name": "swxtch-tun0",
        "index": 6,
        "ip": "172.30.0.27",
        "subnetPrefix": "172.30.0.0/22",
        "subnetMask": "255.255.252.0",
        "mtu": 4096,
        "mac": "",
        "broadcastIp": "172.30.3.255",
        "driver": null,
        "pciAddress": null,
        "publicIp": null,
        "masterOf": null,
        "vpc": null
    }
},
"vpcs": {}
},
"swxGeneral": {
    "type": "2",
    "dataInterfaceName": "eth1",
    "controlInterfaceName": "eth0",
    "dataPort": 9999,
    "version": "dev.7f1520",
    "configuration": {
        "controlInterface": "eth0",
        "dataInterface": "eth1",
        "dataPlaneSpecs": {
            "bpfPrograms": [
                {
                    "attachPoint": "BPF_TC_INGRESS",
                    "interface": "eth1",
                    "name": "tc-ingress"
                },
                {
                    "attachPoint": "BPF_TC_EGRESS",
                    "interface": "eth1",
                    "name": "tc-egress"
                },
                {
                    "attachPoint": "BPF_TC_EGRESS",
                    "interface": "eth0",
                    "name": "tc-forwarder"
                }
            ]
        },
        "verbosity": 0,
        "virtualInterface": {
            "ip": "172.30.0.27",
            "mtu": 4096,
            "name": "swxtch-tun0",
            "subnet": "255.255.252.0",
            "type": "tun"
        }
    },
    "dataPort": 9999,
    "ha": {
        "bufferSizeInPackets": 131072,
        "maxTimeToBufferPacketsMs": 50,
        "protocol": "swxtch",
        "removeInactiveStreamTimeoutSec": 5
    },
    "nicsConfig": null,
    "overrideSrcIp": false,
    "primarySwxtchAddress": "http://10.2.128.10:80",
    "statsReportWait": 60,
    "streamSpecs": null,
    "subscriptionsPollingIntervalMs": 100,
    "swxtch": "10.2.128.10",
    "xnixRpcPort": 10002,
    "xnixType": 2
},

```



```

"services": [
  {
    "name": "swxtch-xnic-data",
    "isRunning": true
  },
  {
    "name": "swxtch-xnic-control",
    "isRunning": true
  }
],
"subscriptions": {
  "eth1": {
    "224.0.0.251": {
      "filter": "exclude",
      "srcList": []
    },
    "239.1.1.2": {
      "filter": "exclude",
      "srcList": []
    }
  }
},
"parentId": "10.2.128.10",
"timing": {
  "Master": null,
  "LocalOffset": null,
  "RootOffset": null,
  "TimebeatPresent": null
},
"adaptors": null,
"stats": {
  "totals": {
    "eth1": {
      "tx": {
        "packets": 0,
        "bytes": 0
      },
      "txRate": {
        "packets": 0,
        "bytes": 0
      },
      "rx": {
        "packets": 12811621,
        "bytes": 2459831232
      },
      "rxRate": {
        "packets": 1846,
        "bytes": 354383
      },
      "txDrops": {
        "packets": 0,
        "bytes": 0
      },
      "rxDrops": {
        "packets": 0,
        "bytes": 0
      }
    },
    "timestamp": 1724082856842266400
  }
},
"streamLinks": {
  "eth1": {
    "ingress": {
      "Group = 239.1.1.2:3490 | Source = 10.2.128.15:50226 | Adjacent = 10.2.192.116:5026": {
        "packets": 6492383,
        "bytes": 1246537536,
        "protocol": "xMC",
        "protocolStats": null,
        "timestamp": 1724082856842280400,
        "packetsRate": 923,
        "bytesRates": 177177,
        "haPaths": [
          0
        ],
        "fragStats": null
      },
      "Group = 239.1.1.2:3490 | Source = 10.2.128.15:50226 | Adjacent = 10.5.2.6:5026": {
        "packets": 6319238,
        "bytes": 1213293696,
        "protocol": "xMC",
        "protocolStats": null,

```

```

        "timestamp": 1724082856842280400,
        "packetsRate": 923,
        "bytesRates": 177206,
        "haPaths": [
            1
        ],
        "fragStats": null
    },
    },
    "egress": {}
}
},
"hsCounters": {
    "eth1": [
        {
            "streamId": {
                "ip": 33620463,
                "port": 41485
            },
            "pathStats": [
                {
                    "ingressPackets": 6492383,
                    "ingressBytes": 1246537536,
                    "ingressPacketsRate": 923,
                    "ingressBytesRate": 177177,
                    "missingPackets": 165,
                    "missingPacketsRate": 0,
                    "outputStreamPackets": 177731,
                    "pathUsage": 0,
                    "outputStreamPacketsRate": 0
                },
                {
                    "ingressPackets": 6319238,
                    "ingressBytes": 1213293696,
                    "ingressPacketsRate": 923,
                    "ingressBytesRate": 177206,
                    "missingPackets": 173311,
                    "missingPacketsRate": 0,
                    "outputStreamPackets": 6314818,
                    "pathUsage": 100,
                    "outputStreamPacketsRate": 923
                }
            ],
            "enqueueFailurePackets": 0,
            "egressPackets": 6492549,
            "egressBytes": 1246569408,
            "egressPacketsRate": 923,
            "egressBytesRate": 177206,
            "outputStreamLoss": 0,
            "outputStreamLossRate": 0,
            "senderIp": "10.2.128.15"
        }
    ]
},
"hsSummary": {
    "eth1": {
        "pathSummaries": [
            {
                "ingressPacketsTotal": 6492383,
                "ingressBytesTotal": 1246537536,
                "missingPacketsTotal": 165,
                "outputStreamPacketsTotal": 177731,
                "ingressPacketsRate": 923,
                "ingressBytesRate": 177177,
                "missingPacketsRate": 0,
                "outputStreamPacketsRate": 0,
                "pathUsage": 0
            },
            {
                "ingressPacketsTotal": 6319238,
                "ingressBytesTotal": 1213293696,
                "missingPacketsTotal": 173311,
                "outputStreamPacketsTotal": 6314818,
                "ingressPacketsRate": 923,
                "ingressBytesRate": 177206,
                "missingPacketsRate": 0,
                "outputStreamPacketsRate": 923,
                "pathUsage": 100
            }
        ],
        "enqueueFailurePackets": 0,
        "egressPackets": 6492549,

```

```

        "egressBytes": 1246569408,
        "outputStreamLoss": 0,
        "egressPacketsRate": 923,
        "egressBytesRate": 177206,
        "outputStreamLossRate": 0
    }
},
    "slp": null
},
    "isRemote": false
},
    "10.5.1.4": {
        "general": {
            "id": "10.5.1.4",
            "name": "DSd-agent-201",
            "componentKind": {
                "code": "xnic",
                "displayName": "xNIC"
            }
        },
        "environment": {
            "hostname": "DSd-agent-201",
            "cloud": "AZURE",
            "osDistribution": "Ubuntu 20.04",
            "region": "eastus",
            "instanceType": "Standard_D4s_v4"
        },
        "hardware": {
            "nics": {
                "eth0": {
                    "name": "eth0",
                    "index": 2,
                    "ip": "10.5.1.4",
                    "subnetPrefix": "10.5.1.0/24",
                    "subnetMask": "255.255.255.0",
                    "mtu": 1500,
                    "mac": "00:22:48:23:4c:48",
                    "broadcastIp": "10.5.1.255",
                    "driver": "mlx5_core",
                    "pciAddress": "1e1b:00:02.0",
                    "publicIp": null,
                    "masterOf": "enP7707s1",
                    "vpc": null
                },
                "eth1": {
                    "name": "eth1",
                    "index": 3,
                    "ip": "10.5.2.4",
                    "subnetPrefix": "10.5.2.0/24",
                    "subnetMask": "255.255.255.0",
                    "mtu": 1500,
                    "mac": "00:22:48:23:43:76",
                    "broadcastIp": "10.5.2.255",
                    "driver": "mlx5_core",
                    "pciAddress": "6df3:00:02.0",
                    "publicIp": null,
                    "masterOf": "enP28147s2",
                    "vpc": null
                }
            },
            "swxtch-tun0": {
                "name": "swxtch-tun0",
                "index": 6,
                "ip": "172.30.0.4",
                "subnetPrefix": "172.30.0.0/24",
                "subnetMask": "255.255.255.0",
                "mtu": 4096,
                "mac": "",
                "broadcastIp": "172.30.0.255",
                "driver": null,
                "pciAddress": null,
                "publicIp": null,
                "masterOf": null,
                "vpc": null
            }
        },
        "vpcs": {}
    },
    "swxGeneral": {
        "type": "2",
        "dataInterfaceName": "eth1",
        "controlInterfaceName": "eth0",
        "dataPort": 9999,

```

```

"version": "dev.7f1520",
"configuration": {
  "controlInterface": "eth0",
  "dataInterface": "eth1",
  "dataPlaneSpecs": {
    "bpfPrograms": [
      {
        "attachPoint": "BPF_TC_INGRESS",
        "interface": "eth1",
        "name": "tc-ingress"
      },
      {
        "attachPoint": "BPF_TC_EGRESS",
        "interface": "eth1",
        "name": "tc-egress"
      },
      {
        "attachPoint": "BPF_TC_EGRESS",
        "interface": "eth0",
        "name": "tc-forwarder"
      }
    ],
    "verbosity": 0,
    "virtualInterface": {
      "ip": "172.30.0.4",
      "mtu": 4096,
      "name": "swxtch-tun0",
      "subnet": "255.255.255.0",
      "type": "tun"
    }
  },
  "dataPort": 9999,
  "ha": {
    "bufferSizeInPackets": 131072,
    "maxTimeToBufferPacketsMs": 50,
    "protocol": "swxtch",
    "removeInactiveStreamTimeoutSec": 5
  },
  "nicsConfig": null,
  "overrideSrcIp": false,
  "primarySwxtchAddress": "http://10.5.1.6:80",
  "statsReportWait": 60,
  "streamSpecs": null,
  "subscriptionsPollingIntervalMs": 100,
  "swxtch": "10.5.1.6",
  "xnicRpcPort": 10002,
  "xnicType": 2
},
"services": [
  {
    "name": "swxtch-xnic-control",
    "isRunning": true
  },
  {
    "name": "swxtch-xnic-data",
    "isRunning": true
  }
],
"subscriptions": {
  "eth1": {
    "224.0.0.251": {
      "filter": "exclude",
      "srcList": []
    },
    "239.1.1.2": {
      "filter": "exclude",
      "srcList": []
    }
  }
}
},
"parentId": "10.2.128.10",
"timing": {
  "Master": null,
  "LocalOffset": null,
  "RootOffset": null,
  "TimebeatPresent": null
},
"adaptors": null,
"stats": {
  "totals": {
    "eth1": {

```

```

        "tx": {
            "packets": 0,
            "bytes": 0
        },
        "txRate": {
            "packets": 0,
            "bytes": 0
        },
        "rx": {
            "packets": 5083429,
            "bytes": 976018368
        },
        "rxRate": {
            "packets": 1847,
            "bytes": 354719
        },
        "txDrops": {
            "packets": 0,
            "bytes": 0
        },
        "rxDrops": {
            "packets": 0,
            "bytes": 0
        },
        "timestamp": 1724082856353285592
    },
    "streamLinks": {
        "eth1": {
            "ingress": {
                "Group = 239.1.1.2:3490 | Source = 10.2.128.15:50226 | Adjacent = 10.2.192.116:5026": {
                    "packets": 2541784,
                    "bytes": 488022528,
                    "protocol": "xMC",
                    "protocolStats": null,
                    "timestamp": 1724082856353297892,
                    "packetsRate": 924,
                    "bytesRates": 177344,
                    "haPaths": [
                        0
                    ],
                    "fragStats": null
                },
                "Group = 239.1.1.2:3490 | Source = 10.2.128.15:50226 | Adjacent = 10.5.2.6:5026": {
                    "packets": 2541645,
                    "bytes": 487995840,
                    "protocol": "xMC",
                    "protocolStats": null,
                    "timestamp": 1724082856353297892,
                    "packetsRate": 924,
                    "bytesRates": 177375,
                    "haPaths": [
                        1
                    ],
                    "fragStats": null
                }
            },
            "egress": {}
        }
    },
    "hsCounters": {
        "eth1": [
            {
                "streamId": {
                    "ip": 33620463,
                    "port": 41485
                },
                "pathStats": [
                    {
                        "ingressPackets": 2541784,
                        "ingressBytes": 488022528,
                        "ingressPacketsRate": 924,
                        "ingressBytesRate": 177344,
                        "missingPackets": 27,
                        "missingPacketsRate": 0,
                        "outputStreamPackets": 1526,
                        "pathUsage": 0,
                        "outputStreamPacketsRate": 0
                    },
                    {
                        "ingressPackets": 2541645,
                        "ingressBytes": 487995840,

```

```

        "ingressPacketsRate": 924,
        "ingressBytesRate": 177375,
        "missingPackets": 167,
        "missingPacketsRate": 0,
        "outputStreamPackets": 2540286,
        "pathUsage": 100,
        "outputStreamPacketsRate": 924
    }
},
    "enqueueFailurePackets": 0,
    "egressPackets": 2541812,
    "egressBytes": 488027904,
    "egressPacketsRate": 924,
    "egressBytesRate": 177375,
    "outputStreamLoss": 0,
    "outputStreamLossRate": 0,
    "senderIp": "10.2.128.15"
}
],
    "hsSummary": {
        "eth1": {
            "pathSummaries": [
                {
                    "ingressPacketsTotal": 2541784,
                    "ingressBytesTotal": 488022528,
                    "missingPacketsTotal": 27,
                    "outputStreamPacketsTotal": 1526,
                    "ingressPacketsRate": 924,
                    "ingressBytesRate": 177344,
                    "missingPacketsRate": 0,
                    "outputStreamPacketsRate": 0,
                    "pathUsage": 0
                },
                {
                    "ingressPacketsTotal": 2541645,
                    "ingressBytesTotal": 487995840,
                    "missingPacketsTotal": 167,
                    "outputStreamPacketsTotal": 2540286,
                    "ingressPacketsRate": 924,
                    "ingressBytesRate": 177375,
                    "missingPacketsRate": 0,
                    "outputStreamPacketsRate": 924,
                    "pathUsage": 100
                }
            ],
            "enqueueFailurePackets": 0,
            "egressPackets": 2541812,
            "egressBytes": 488027904,
            "outputStreamLoss": 0,
            "egressPacketsRate": 924,
            "egressBytesRate": 177375,
            "outputStreamLossRate": 0
        }
    },
    "slp": null
},
    "isRemote": false
},
    "10.5.1.6": {
        "general": {
            "id": "10.5.1.6",
            "name": "dsd-core-200",
            "componentKind": {
                "code": "swxtch",
                "displayName": "swXtch"
            }
        },
        "environment": {
            "hostname": "dsd-core-200",
            "cloud": "AZURE",
            "osDistribution": "Ubuntu 20.04",
            "region": "eastus",
            "instanceType": "Standard_D8s_v4"
        },
        "hardware": {
            "nics": {
                "eth0": {
                    "name": "eth0",
                    "index": 2,
                    "ip": "10.5.1.6",
                    "subnetPrefix": "10.5.1.0/24",

```

```

        "subnetMask": "255.255.255.0",
        "mtu": 1500,
        "mac": "00:22:48:24:75:06",
        "broadcastIp": "10.5.1.255",
        "driver": "hv_netvsc",
        "pciAddress": "",
        "publicIp": null,
        "masterOf": null,
        "vpc": null
    },
    "eth1": {
        "name": "eth1",
        "index": 3,
        "ip": "10.5.2.6",
        "subnetPrefix": "10.5.2.0/24",
        "subnetMask": "255.255.255.0",
        "mtu": 1500,
        "mac": "00:22:48:24:77:db",
        "broadcastIp": "10.5.2.255",
        "driver": "mlx5_core",
        "pciAddress": "76ac:00:02.0",
        "publicIp": null,
        "masterOf": "enP30380s2",
        "vpc": null
    }
},
"vpcs": {}
},
"swxGeneral": {
    "type": "X1",
    "dataInterfaceName": "eth1",
    "controlInterfaceName": "eth0",
    "dataPort": 9999,
    "version": "dev.7f1520",
    "configuration": null,
    "services": null,
    "subscriptions": null
},
"parentId": null,
"timing": {
    "Master": null,
    "LocalOffset": null,
    "RootOffset": null,
    "TimebeatPresent": null
},
"adaptors": [],
"stats": {
    "totals": {
        "eth1": {
            "tx": {
                "packets": 5204581,
                "bytes": 999279552
            },
            "txRate": {
                "packets": 1844,
                "bytes": 354105
            },
            "rx": {
                "packets": 2664303,
                "bytes": 511546176
            },
            "rxRate": {
                "packets": 925,
                "bytes": 177565
            },
            "txDrops": null,
            "rxDrops": null,
            "timestamp": 1724082855943717450
        }
    },
    "streamLinks": {
        "eth1": {
            "ingress": {
                "Group = 239.1.1.2:3490 | Source = 10.2.128.15:50226 | Adjacent = 10.2.192.15:50226": {
                    "packets": 2664303,
                    "bytes": 511546176,
                    "protocol": "xMC",
                    "protocolStats": null,
                    "timestamp": 1724082855918726621,
                    "packetsRate": 921,
                    "bytesRate": 176781,
                    "haPaths": [

```

```

        1
      ],
      "fragStats": null
    }
  },
  "egress": {
    "Group = 239.1.1.2:3490 | Source = 10.2.128.15:50226 | Adjacent = 10.2.192.24:9999": {
      "packets": 2663319,
      "bytes": 511357248,
      "protocol": "xMC",
      "protocolStats": null,
      "timestamp": 1724082855943490143,
      "packetsRate": 921,
      "bytesRates": 176772,
      "haPaths": [
        1
      ],
      "fragStats": null
    },
    "Group = 239.1.1.2:3490 | Source = 10.2.128.15:50226 | Adjacent = 10.5.2.4:9999": {
      "packets": 2541262,
      "bytes": 487922304,
      "protocol": "xMC",
      "protocolStats": null,
      "timestamp": 1724082855943490143,
      "packetsRate": 921,
      "bytesRates": 176772,
      "haPaths": [
        1
      ],
      "fragStats": null
    }
  }
},
"hsCounters": {
  "eth1": []
},
"hsSummary": {},
"slp": {
  "rxSlpStats": {},
  "txSlpStats": null
}
},
"isRemote": true
}
},
"timestamp": "2024-08-19T15:54:17.4656522Z",
"aliases": {
  "components": {},
  "sockets": {},
  "streams": {
    "225.1.1.1:1599": "SRT_L_I_225_1_1_1_1599",
    "239.4.2.3:5400": "SRT_C_239_4_2_3_1400"
  }
}
}
}

```

Cloud Information

GET

/swxtch/monitoring/v1/info/cloud

- Get information of the cloud for which the cloudSwXtch is installed on

URL:

Bash

Copy

http://<cloudSwXtch-control-IP>/swxtch/monitoring/v1/info/cloud

Example URL:

http://10.2.128.10/swxtch/monitoring/v1/info/cloud

Request:

Empty

Response:

200 - successful operation

Example Response:

ActionScript

Copy

```
{
  "azureData": {
    "subscriptionId": "b10209ad-ad22-4c26-8aef-be93b2f0bb58",
    "managedResourceGroupName": "saDevNetwork",
    "resourceGorupName": "saDevNetwork",
    "sku": null,
    "billingFields": null,
    "plan": "byol-001"
  },
  "awsData": null,
  "gcpData": null,
  "ociData": null,
  "alibabaData": null
}
```

cloudSwXtch Information

GET

/swxtch/monitoring/v1/info/swxtch

- Get information on the cloudSwXtch

URL:

Bash

Copy

```
curl http://<cloudSwXtch-control-IP>/swxtch/monitoring/v1/info/swxtch
```

Example URL:

http://10.2.128.10/swxtch/monitoring/v1/info/swxtch

Request:

Empty

Response:

200 - successful response

Example Response:

PowerShell

Copy

```
{
  "cloudSwxtchVersion": "3.0.0",
  "planType": "dsd-core-100",
  "cloudProvider": "AZURE",
  "ipAddr": "10.2.128.10",
  "swxtchGuid": "213de388-ccb5-4d93-9159-3fa9ad5585ea",
  "swxtchName": "dsd-core-100",
  "hostName": "dsd-core-100",
  "numCores": 1,
  "numXnics": null,
  "numBridges": null,
  "replStatus": "running",
  "authorized": true,
  "isMarketplace": true,
  "remainingDays": null,
  "license": null,
  "entitlements": {
    "maxClientCount": 40,
    "maxBridgeCount": 2,
    "bandwidthMbps": 25000,
    "enableMesh": true,
    "enableUnicast": true,
    "enableHA": true,
    "enableClockSync": true,
    "enableBridge": true,
    "enableWxckedEye": true,
    "enableAllowsMajorVersionUpdate": true,
    "enableTachyonLive": false
  },
  "subnetDataPrefix": "10.2.192.0/22",
  "subnetCtrlPrefix": "10.2.128.0/22",
  "dataGatewayIp": "10.2.192.1",
  "ctrlIp": "10.2.128.10",
  "ctrlPort": 10802,
  "gatewayMacAddr": "12:34:56:78:9a:bc",
  "replInfo": {
    "ctrlIp": "127.0.0.1",
    "ctrlPort": 9996,
    "dataIp": "10.2.192.116",
    "dataPort": 9999,
    "dataMac": "AA06VA83"
  },
  "licenseType": null,
  "licenseExpiryDate": null,
}
```

cloudSwXtch Stats

GET

/swxtch/monitoring/v1/stats/swxtch

- Get status of the cloudSwXtch

URL:

Bash

Copy

```
curl http://<cloudSwXtch-control-IP>/swxtch/monitoring/v1/stats/swxtch
```

Example URL:
http://10.2.128.10/switch/monitoring/v1/stats/switch
Request: Empty
Response:
200 - successful response

Example Response —>

PowerShellCopy

```
{
  "host": "10.2.192.116",
  "sequence": 9907,
  "rxCount": 18717937,
  "txCount": 21711090,
  "rxBytes": 3593894567,
  "txBytes": 4168642226,
  "rxBridgeBytes": 0,
  "rxBridgeCount": 0,
  "txBridgeBytes": 0,
  "txBridgeCount": 0,
  "timestamp": 1723834796290675315,
  "dropsByByteLimit": 0,
  "dropsByCountLimit": 0,
  "rxMeshPktCount": 0,
  "rxMeshBytes": 0,
  "txMeshPktCount": 0,
  "txMeshBytes": 0,
  "rxUnicastPktCount": 0,
  "rxUnicastBytes": 0,
  "txUnicastPktCount": 0,
  "txUnicastBytes": 0,
  "streams": {
    "rxStreamLinks": [
      {
        "packets": 13159186,
        "bytes": 2526563712,
        "protocol": "xMC",
        "protocolStats": null,
        "fragStats": null,
        "timestamp": 1723829520431370897,
        "groupAddress": "239.1.1.2:3490",
        "adjacentAddress": "10.2.192.15:57795",
        "sourceAddress": "10.2.128.15:57795"
      },
      {
        "packets": 3927329,
        "bytes": 754047168,
        "protocol": "xMC",
        "protocolStats": null,
        "fragStats": null,
        "timestamp": 1723834796288368432,
        "groupAddress": "239.1.1.2:3490",
        "adjacentAddress": "10.2.192.15:57449",
        "sourceAddress": "10.2.128.15:57449"
      },
      {
        "packets": 63503,
        "bytes": 12192576,
        "protocol": "xMC",
        "protocolStats": null,
        "fragStats": null,
        "timestamp": 1723834796288368432,
        "groupAddress": "239.1.1.3:3490",
        "adjacentAddress": "10.2.192.24:50583",
        "sourceAddress": "10.2.192.24:50583"
      }
    ],
    "txStreamLinks": [
      {
        "packets": 21258,
        "bytes": 4081536,
        "protocol": "xMC",
        "protocolStats": null,
        "fragStats": null,
        "timestamp": 1723834796289254064,
        "groupAddress": "239.1.1.3:3490",
        "adjacentAddress": "10.2.192.11:9999",
        "sourceAddress": "10.2.192.24:50583"
      }
    ]
  }
}
```

```

    "packets": 13136117,
    "bytes": 2522134464,
    "protocol": "xMC",
    "protocolstats": null,
    "fragStats": null,
    "timestamp": 1723829520439529491,
    "groupAddress": "239.1.1.2:3490",
    "adjacentAddress": "10.2.192.24:9999",
    "sourceAddress": "10.2.128.15:57795"
  },
  {
    "packets": 3927319,
    "bytes": 754045248,
    "protocol": "xMC",
    "protocolstats": null,
    "fragStats": null,
    "timestamp": 1723834796276642010,
    "groupAddress": "239.1.1.2:3490",
    "adjacentAddress": "10.2.192.24:9999",
    "sourceAddress": "10.2.128.15:57449"
  },
  {
    "packets": 3069122,
    "bytes": 589271424,
    "protocol": "xMC",
    "protocolstats": null,
    "fragStats": null,
    "timestamp": 1723834796289254064,
    "groupAddress": "239.1.1.2:3490",
    "adjacentAddress": "10.2.192.11:9999",
    "sourceAddress": "10.2.128.15:57449"
  }
]
}

```

For definitions of source/adjacent addresses and protocol types, see [swXtch.io API Concepts explained](#) above.

xNIC Totals

GET

/swxtch/monitoring/v1/stats/xnicsTotals

- Get status of the xNIC totals

URL:

Bash

Copy

```
curl http://<cloudSwxtch-control-IP>/swxtch/monitoring/v1/stats/xnicsTotals
```

Example URL:

http://10.2.128.10/swxtch/monitoring/v1/stats/xnicsTotals

Request:

Empty

Response:

200 - successful response

Example Response:

PowerShell

Copy

```
{
  "activeConnectionCount": null,
  "byteCounters": {
    "rxMulticastCount": 0,
    "rxTotalCount": 7885695384,
    "txMulticastCount": 0,
    "txTotalCount": 3500298564
  },
  "cloudSwxtchVersion": null,
  "dataAddress": null,
  "hostAddress": null,
  "latencies": null,
  "maxActiveConnections": null,
  "osDistribution": null,
  "packetCounters": {
    "-": 0,
    "rxDroppedCount": 0,
    "rxMulticastCount": 0,
    "rxTotalCount": 41070499,
    "txDroppedCount": 0,
    "txIgmpCount": 0,
    "txMulticastCount": 0,
    "txTotalCount": 18228612
  },
  "rxHaCounters": [],
  "rxMulticastGroups": [],
  "timestamp": 1723835599071011224,
  "txMulticastGroups": [],
  "xNicMode": null,
  "xNicType": null
}
```

xNIC Stats

GET

/swxtch/monitoring/v1/stats/xnics

- Get status of the xNICs

URL:

Bash

Copy

```
curl http://<cloudSwxtch-control-IP>/swxtch/monitoring/v1/stats/xnics
```

Example URL:

http://10.2.128.10/swxtch/monitoring/v1/stats/xnics

Request:

Empty

Response:

200- successful response

Example Response —>

ActionScript

Copy

```
{
  "10.2.128.13": {
    "activeConnectionCount": 2,
```

```

"byteCounters": {
  "rxMulticastCount": 0,
  "rxTotalCount": 6170369908,
  "txMulticastCount": 0,
  "txTotalCount": 2832
},
"cloudSwxtchVersion": "dev.ac84d0",
"dataAddress": "10.2.192.11",
"hostAddress": "10.2.128.13",
"latencies": {
  "buckets": {},
  "count": 0,
  "sum": 0
},
"maxActiveConnections": 2,
"osDistribution": "Ubuntu 20.04",
"packetCounters": {
  "-": 0,
  "rxDroppedCount": 0,
  "rxMulticastCount": 0,
  "rxTotalCount": 32137435,
  "txDroppedCount": 0,
  "txIgmpCount": 0,
  "txMulticastCount": 0,
  "txTotalCount": 22
},
"rxHaCounters": [
  {
    "byteCount": 1976033472,
    "destinationCount": 0,
    "groupIp": "239.1.1.3",
    "packetCount": 10291841,
    "protocolType": 0,
    "sourceIp": "",
    "sourcePort": 0,
    "updateTime": "0001-01-01T00:00:00Z"
  },
  {
    "byteCount": 1109021376,
    "destinationCount": 0,
    "groupIp": "239.1.1.2",
    "packetCount": 5776153,
    "protocolType": 0,
    "sourceIp": "",
    "sourcePort": 0,
    "updateTime": "0001-01-01T00:00:00Z"
  },
  {
    "byteCount": 1976052480,
    "destinationCount": 0,
    "groupIp": "239.1.1.3",
    "packetCount": 10291940,
    "protocolType": 0,
    "sourceIp": "",
    "sourcePort": 0,
    "updateTime": "0001-01-01T00:00:00Z"
  },
  {
    "byteCount": 1109027520,
    "destinationCount": 0,
    "groupIp": "239.1.1.2",
    "packetCount": 5776185,
    "protocolType": 0,
    "sourceIp": "",
    "sourcePort": 0,
    "updateTime": "0001-01-01T00:00:00Z"
  }
],
"timestamp": 1723837738833165495,
"txMulticastGroups": [
  {
    "byteCount": 294,
    "destinationCount": 0,
    "groupIp": "255.255.255.255",
    "packetCount": 2,
    "protocolType": 0,
    "sourceIp": "",
    "sourcePort": 0,
    "updateTime": "0001-01-01T00:00:00Z"
  },
  {
    "byteCount": 89,

```

```

        "destinationCount": 0,
        "groupId": "255.255.255.255",
        "packetCount": 1,
        "protocolType": 0,
        "sourceIp": "",
        "sourcePort": 0,
        "updateTime": "0001-01-01T00:00:00Z"
    }
},
"xNicMode": "HA",
"xNicType": "t2"
},
"10.2.128.15": {
    "activeConnectionCount": 2,
    "byteCounters": {
        "rxMulticastCount": 0,
        "rxTotalCount": 59072,
        "txMulticastCount": 0,
        "txTotalCount": 2546472892
    },
    "cloudSwxtchVersion": "dev.ac84d0",
    "dataAddress": "10.2.192.15",
    "hostAddress": "10.2.128.15",
    "latencies": {
        "buckets": {},
        "count": 0,
        "sum": 0
    },
    "maxActiveConnections": 2,
    "osDistribution": "Windows Server 2019 Datacenter - Microsoft Windows [Version 10.0.17763.6189]",
    "packetCounters": {
        "-": 0,
        "rxDroppedCount": 0,
        "rxMulticastCount": 0,
        "rxTotalCount": 340,
        "txDroppedCount": 0,
        "txIgmpCount": 0,
        "txMulticastCount": 0,
        "txTotalCount": 13262608
    },
    "rxHaCounters": [
        {
            "egressByteCount": 275,
            "egressPacketCount": 1,
            "enqueueFailureCount": 0,
            "groupId": {
                "ip": 4286775818,
                "port": 35328
            },
            "outputStreamLossCount": 0,
            "paths": [
                {
                    "ingressByteCount": 275,
                    "ingressPacketCount": 1,
                    "missingPacketCount": 0,
                    "usedPacketCount": 0
                },
                {
                    "ingressByteCount": 275,
                    "ingressPacketCount": 1,
                    "missingPacketCount": 0,
                    "usedPacketCount": 1
                }
            ],
            "senderIp": "10.2.128.75"
        },
        {
            "egressByteCount": 275,
            "egressPacketCount": 1,
            "enqueueFailureCount": 0,
            "groupId": {
                "ip": 4290970122,
                "port": 35328
            },
            "outputStreamLossCount": 0,
            "paths": [
                {
                    "ingressByteCount": 275,
                    "ingressPacketCount": 1,
                    "missingPacketCount": 0,
                    "usedPacketCount": 0
                }
            ],
        },
    ],

```

```

        {
            "ingressByteCount": 275,
            "ingressPacketCount": 1,
            "missingPacketCount": 0,
            "usedPacketCount": 1
        }
    ],
    "senderIp": "10.2.192.82"
},
],
"rxMulticastGroups": [
    {
        "byteCount": 1925,
        "destinationCount": 0,
        "groupId": "10.2.131.255",
        "packetCount": 7,
        "protocolType": 0,
        "sourceIp": "",
        "sourcePort": 0,
        "updateTime": "0001-01-01T00:00:00Z"
    },
    {
        "byteCount": 1116,
        "destinationCount": 0,
        "groupId": "10.2.195.255",
        "packetCount": 9,
        "protocolType": 0,
        "sourceIp": "",
        "sourcePort": 0,
        "updateTime": "0001-01-01T00:00:00Z"
    }
],
"timestamp": 1723837734873976100,
"txMulticastGroups": [
    {
        "byteCount": 1273105920,
        "destinationCount": 0,
        "groupId": "239.1.1.2",
        "packetCount": 6630760,
        "protocolType": 0,
        "sourceIp": "",
        "sourcePort": 0,
        "updateTime": "0001-01-01T00:00:00Z"
    },
    {
        "byteCount": 1273105920,
        "destinationCount": 0,
        "groupId": "239.1.1.2",
        "packetCount": 6630760,
        "protocolType": 0,
        "sourceIp": "",
        "sourcePort": 0,
        "updateTime": "0001-01-01T00:00:00Z"
    }
],
"xNicMode": "Normal",
"xNicType": "t2"
},
"10.2.128.27": {
    "activeConnectionCount": 2,
    "byteCounters": {
        "rxMulticastCount": 0,
        "rxTotalCount": 6094630132,
        "txMulticastCount": 0,
        "txTotalCount": 3968082038
    },
    "cloudSwxtchVersion": "dev.ac84d0",
    "dataAddress": "10.2.192.24",
    "hostAddress": "10.2.128.27",
    "latencies": {
        "buckets": {},
        "count": 0,
        "sum": 0
    },
    "maxActiveConnections": 2,
    "osDistribution": "Ubuntu 20.04",
    "packetCounters": {
        "-": 0,
        "rxDroppedCount": 0,
        "rxMulticastCount": 0,
        "rxTotalCount": 31742957,
        "txDroppedCount": 0,

```



```

    "txIcmpCount": 0,
    "txMulticastCount": 0,
    "txTotalCount": 20667102
  },
  "rxHaCounters": [
    {
      "egressByteCount": 1273770240,
      "egressPacketCount": 6634220,
      "enqueueFailureCount": 0,
      "groupIp": {
        "ip": 33620463,
        "port": 41485
      },
      "outputStreamLossCount": 0,
      "paths": [
        {
          "ingressByteCount": 1273767936,
          "ingressPacketCount": 6634208,
          "missingPacketCount": 11,
          "usedPacketCount": 18
        },
        {
          "ingressByteCount": 1273770240,
          "ingressPacketCount": 6634220,
          "missingPacketCount": 0,
          "usedPacketCount": 6634202
        }
      ],
      "senderIp": "10.2.128.15"
    }
  ],
  "rxMulticastGroups": [
    {
      "byteCount": 1771408320,
      "destinationCount": 0,
      "groupIp": "239.1.1.2",
      "packetCount": 9226085,
      "protocolType": 0,
      "sourceIp": "",
      "sourcePort": 0,
      "updateTime": "0001-01-01T00:00:00Z"
    },
    {
      "byteCount": 1273770240,
      "destinationCount": 0,
      "groupIp": "239.1.1.2",
      "packetCount": 6634220,
      "protocolType": 0,
      "sourceIp": "",
      "sourcePort": 0,
      "updateTime": "0001-01-01T00:00:00Z"
    },
    {
      "byteCount": 1775448576,
      "destinationCount": 0,
      "groupIp": "239.1.1.2",
      "packetCount": 9247128,
      "protocolType": 0,
      "sourceIp": "",
      "sourcePort": 0,
      "updateTime": "0001-01-01T00:00:00Z"
    },
    {
      "byteCount": 1273767936,
      "destinationCount": 0,
      "groupIp": "239.1.1.2",
      "packetCount": 6634208,
      "protocolType": 0,
      "sourceIp": "",
      "sourcePort": 0,
      "updateTime": "0001-01-01T00:00:00Z"
    }
  ],
  "timestamp": 1723837738670810658,
  "txMulticastGroups": [
    {
      "byteCount": 1984039872,
      "destinationCount": 0,
      "groupIp": "239.1.1.3",
      "packetCount": 10333541,
      "protocolType": 0,

```

```

        "sourceIp": "",
        "sourcePort": 0,
        "updateTime": "0001-01-01T00:00:00Z"
    },
    {
        "byteCount": 1984039872,
        "destinationCount": 0,
        "groupIp": "239.1.1.3",
        "packetCount": 10333541,
        "protocolType": 0,
        "sourceIp": "",
        "sourcePort": 0,
        "updateTime": "0001-01-01T00:00:00Z"
    },
    {
        "byteCount": 111,
        "destinationCount": 0,
        "groupIp": "255.255.255.255",
        "packetCount": 1,
        "protocolType": 0,
        "sourceIp": "",
        "sourcePort": 0,
        "updateTime": "0001-01-01T00:00:00Z"
    }
],
"xNicMode": "HA",
"xNicType": "t2"
}
}

```

Configuration API

Overview

The cloudSwXtch Configuration API is intended for use to integrate the cloudSwXtch data with third party tools for configuring High Availability and Protocol Fanout.

Prerequisites

A cloudSwXtch must exist as well as two or more agents with xNICs. To have data, agents must be producing and consuming data via the cloudSwXtch. By using a GET command, data will be provided in the response.

GET

/services/settings

URL:

PowerShellCopy

http://<cloudSwXtch-control-IP>/services/settings

Example URL:

http://10.2.128.10/services/settings

Request:

Empty

Response:

200 - successful operation

The wXcked Eye Settings API call will give users information based on the "Settings" page in the wXcked Eye UI, specifically General, High Availability, and Protocol Fanout. While it takes one call to get all this information, we will be split it into sections based on the appropriate tab.

Expand for an example response of the Settings call:

BashCopy

```
{
  "general": {
    "hostName": "dsd-core-100",
    "switchName": "dsd-core-100",
    "version": "dev.7f1520",
    "numCores": 1,
    "entitlements": {
      "maxClientCount": 40,
      "maxBridgeCount": 2,
      "bandwidthMbps": 25000,
      "enableMesh": true,
      "enableUnicast": true,
      "enableHA": true,
      "enableClockSync": true,
      "enableBridge": true,
      "enableWxckedEye": true,
      "enableAllowsMajorVersionUpdate": true,
      "enableTachyonLive": false
    },
    "subnetDataPrefix": "10.2.192.0/22",
    "subnetCtrlPrefix": "10.2.128.0/22",
    "dataGatewayIp": "10.2.192.1",
    "ctrlIp": "10.2.128.10",
    "ctrlPort": 10802,
    "gatewayMacAddr": "12:34:56:78:9a:bc",
    "replInfo": {
      "ctrlIp": "127.0.0.1",
      "ctrlPort": 9996,
      "dataIp": "10.2.192.116",
      "dataPort": 9999,
      "dataMac": "AA06VA83"
    },
    "numClientsConnected": 8,
    "hardwareInfo": {
      "control": {
        "meta": {
          "mac": "00:0d:3a:19:cc:e1",
          "ipAddress": "10.2.128.10",
          "ipBroadcast": "10.2.131.255",
```

```

        "ipSubnet": "10.2.128.0",
        "subnetMask": "255.255.252.0",
        "mtu": null,
        "driver": null,
        "masterOf": null,
        "ifIndex": null,
        "vpc": null,
        "publicIpAddress": null
    },
    "os": {
        "mac": "00:0d:3a:19:cc:e1",
        "ipAddress": "10.2.128.10",
        "ipBroadcast": "10.2.131.255",
        "ipSubnet": "10.2.128.0",
        "subnetMask": "255.255.252.0",
        "mtu": 1500,
        "driver": "hv_netvsc",
        "masterOf": null,
        "ifIndex": 2,
        "vpc": null,
        "publicIpAddress": null,
        "name": "eth0",
        "pciAddress": "",
        "state": "up"
    },
    "computed": {
        "isPreferredControlNic": true,
        "isPreferredDataNic": false
    }
},
"data": {
    "meta": {
        "mac": "00:0d:3a:54:0f:37",
        "ipAddress": "10.2.192.116",
        "ipBroadcast": "10.2.195.255",
        "ipSubnet": "10.2.192.0",
        "subnetMask": "255.255.252.0",
        "mtu": null,
        "driver": null,
        "masterOf": null,
        "ifIndex": null,
        "vpc": null,
        "publicIpAddress": null
    },
    "os": {
        "mac": "00:0d:3a:54:0f:37",
        "ipAddress": "10.2.192.116",
        "ipBroadcast": "10.2.195.255",
        "ipSubnet": "10.2.192.0",
        "subnetMask": "255.255.252.0",
        "mtu": 1500,
        "driver": "mlx5_core",
        "masterOf": "enP42685s2",
        "ifIndex": 3,
        "vpc": null,
        "publicIpAddress": null,
        "name": "eth1",
        "pciAddress": "a6bd:00:02.0",
        "state": "up"
    },
    "computed": {
        "isPreferredControlNic": false,
        "isPreferredDataNic": true
    }
}
},
"httpPort": 80
},
"mesh": {
    "config": null,
    "routingTable": null
},
"ha": {
    "config": {
        "uid": "cf81bcbb-2afd-ac36-72d0-a67ec56c742c",
        "name": "100-200HA",
        "paths": [
            {
                "name": "100",
                "color": "#fc0303",
                "swxtches": [
                    "10.2.128.10:80"
                ]
            }
        ]
    }
}
}

```

```

    ],
    {
      "name": "200",
      "color": "#033dfc",
      "swxtches": [
        "10.5.1.6:80"
      ]
    }
  ]
},
"membersData": {
  "10.2.128.10:80": {
    "ipAddr": "10.2.128.10:80",
    "isAlive": true,
    "host": null
  },
  "10.5.1.6:80": {
    "ipAddr": "10.5.1.6:80",
    "isAlive": true,
    "host": null
  }
}
},
"fanout": {
  "16704273857340480058": {
    "id": "16704273857340480058",
    "protocol": "srt-caller",
    "direction": "egress",
    "streamIp": "239.4.2.3",
    "streamPort": 5400,
    "nodeIp": "10.2.128.37",
    "nodePort": 5401,
    "listenerPort": null,
    "options": []
  },
  "4777854487999137884": {
    "id": "4777854487999137884",
    "protocol": "srt-caller",
    "direction": "ingress",
    "streamIp": "239.4.2.3",
    "streamPort": 5400,
    "nodeIp": "10.2.128.36",
    "nodePort": 5400,
    "listenerPort": null,
    "options": []
  },
  "5062225067104987852": {
    "id": "5062225067104987852",
    "protocol": "srt-listener",
    "direction": "egress",
    "streamIp": "225.1.1.1",
    "streamPort": 1599,
    "nodeIp": null,
    "nodePort": null,
    "listenerPort": 6000,
    "options": []
  },
  "9489213241930010873": {
    "id": "9489213241930010873",
    "protocol": "srt-listener",
    "direction": "ingress",
    "streamIp": "225.1.1.1",
    "streamPort": 1599,
    "nodeIp": null,
    "nodePort": null,
    "listenerPort": 1599,
    "options": []
  }
},
"channels": null,
"streams": {
  "exclusions": {
    "10.0.0.136": {
      "all": true,
      "ports": {},
      "except": {}
    },
    "10.0.0.197": {
      "all": true,
      "ports": {},
      "except": {}
    }
  }
}

```

```

    },
    "10.0.0.224": {
      "all": true,
      "ports": {},
      "except": {}
    },
    "10.0.0.54": {
      "all": true,
      "ports": {},
      "except": {}
    },
    "10.0.0.99": {
      "all": true,
      "ports": {},
      "except": {}
    },
    "10.0.1.193": {
      "all": true,
      "ports": {},
      "except": {}
    },
    "10.0.1.218": {
      "all": true,
      "ports": {},
      "except": {}
    },
    "10.0.1.244": {
      "all": true,
      "ports": {},
      "except": {}
    },
    "10.0.1.90": {
      "all": true,
      "ports": {},
      "except": {}
    },
    "10.2.131.255": {
      "all": true,
      "ports": {},
      "except": {}
    },
    "10.2.195.255": {
      "all": true,
      "ports": {},
      "except": {}
    },
    "10.5.1.255": {
      "all": true,
      "ports": {},
      "except": {}
    },
    "10.5.2.255": {
      "all": true,
      "ports": {},
      "except": {}
    },
    "172.30.0.255": {
      "all": true,
      "ports": {},
      "except": {}
    },
    "172.30.3.255": {
      "all": true,
      "ports": {},
      "except": {}
    },
    "255.255.255.255": {
      "all": true,
      "ports": {},
      "except": {}
    }
  },
  "groups": null,
  "colors": null,
  "aliases": {
    "225.1.1.1:1599": "CS_SRT_Lis_from-dsd-agent-104",
    "225.1.1.2:1599": "CS_SRT_Lis_from-dsd-agent-104e",
    "225.1.1.2:5400": "SRT-Caller_225.1.1.2",
    "225.1.1.4:1599": "Rist_C_I_225.1.1.4",
    "225.1.1.4:5700": "Rist_C_E_225.1.1.4",
    "234.1.1.2:5000": "test2",
    "234.3.3.3": "test"
  }

```

```

    },
    "streams": null,
    "excluded": null
  },
  "nodes": {
    "icons": {},
    "sizes": {},
    "positions": {
      "bridge-10.2.128.98": {
        "x": 314.33,
        "y": 320.17
      },
      "ss-10.2.128.10-10.2.128.36:5400-239.4.2.3:5400": {
        "x": 507.17,
        "y": 370.09
      },
      "ss-10.2.128.10-10.2.128.37:5401-239.4.2.3:5400": {
        "x": 542.94,
        "y": 315.28
      },
      "ss-10.2.128.10-1599-225.1.1.1:1599": {
        "x": 543.24,
        "y": 322.46
      },
      "ss-10.2.128.10-6000-225.1.1.1:1599": {
        "x": 540.65,
        "y": 304.84
      },
      "ss-10.2.128.15-opposite-225.1.1.1:1599-10.2.128.75:60041-10.2.128.15:49298": {
        "x": 737.89,
        "y": 145.89
      },
      "ss-10.2.128.98-2000-229.1.1.1:2000": {
        "x": 279.26,
        "y": 307.92
      },
      "ss-10.2.192.15-opposite-225.1.1.1:1599-10.2.128.75:35306-10.2.192.15:9999": {
        "x": 632.5,
        "y": 324
      },
      "ss-DSd-agent-102-opposite-239.4.2.3:5400-10.2.128.36:6165-10.2.192.11:9999": {
        "x": 992.06,
        "y": 268.46
      },
      "ss-DSd-agent-104-opposite-225.1.1.1:1599-10.2.128.75:60041-10.2.128.75:60041": {
        "x": 1498.03,
        "y": 749.39
      },
      "ss-DSd-agent-104-opposite-225.1.1.1:1599-10.2.128.75:63813-10.2.128.75:63813": {
        "x": 1498.03,
        "y": 749.39
      },
      "ss-DSd-agent-105-opposite-225.1.1.1:1599-10.2.128.75:17913-10.2.192.15:9999": {
        "x": 743.11,
        "y": 306.81
      },
      "ss-DSd-agent-105-opposite-225.1.1.1:1599-10.2.128.75:35306-10.2.192.15:9999": {
        "x": 788.06,
        "y": 307.15
      },
      "ss-DSd-agent-105-opposite-225.1.1.1:1599-10.2.128.75:60041-10.2.128.15:49298": {
        "x": 741.74,
        "y": 309.6
      },
      "ss-DSd-agent-105-opposite-225.1.1.1:1599-10.2.128.75:63813-10.2.128.15:58348": {
        "x": 739.81,
        "y": 320.84
      },
      "ss-DSd-agent-105-opposite-239.1.1.1:2000-10.2.128.15:64700-10.2.192.15:64700": {
        "x": 789.2,
        "y": 309.53
      },
      "ss-DSd-agent-105-opposite-239.255.255.250:1900-10.2.128.15:49407-10.2.192.15:49407": {
        "x": 790.27,
        "y": 312.59
      },
      "ss-DSd-agent-105-opposite-239.255.255.250:1900-10.2.128.15:51196-10.2.192.15:51196": {
        "x": 785.63,
        "y": 335.46
      },
      "ss-DSd-agent-105-opposite-239.255.255.250:1900-10.2.128.15:51309-10.2.192.15:51309": {
        "x": 739.85,

```

```

      "y": 321.38
    },
    "ss-DSd-agent-105-opposite-239.255.255.250:1900-10.2.128.15:54334-10.2.192.15:54334": {
      "x": 739.79,
      "y": 320.46
    },
    "ss-DSd-agent-105-opposite-239.255.255.250:1900-10.2.128.15:55490-10.2.192.15:55490": {
      "x": 784.67,
      "y": 336.6
    },
    "ss-DSd-agent-105-opposite-239.255.255.250:1900-10.2.128.15:55498-10.2.192.15:55498": {
      "x": 739.93,
      "y": 322.25
    },
    "ss-DSd-agent-105-opposite-239.255.255.250:1900-10.2.128.15:56007-10.2.192.15:56007": {
      "x": 783.82,
      "y": 301.44
    },
    "ss-DSd-agent-105-opposite-239.255.255.250:1900-10.2.128.15:56936-10.2.192.15:56936": {
      "x": 755.27,
      "y": 343.06
    },
    "ss-DSd-agent-105-opposite-239.255.255.250:1900-10.2.128.15:59396-10.2.192.15:59396": {
      "x": 785.88,
      "y": 335.13
    },
    "ss-DSd-agent-105-opposite-239.255.255.250:1900-10.2.128.15:59735-10.2.192.15:59735": {
      "x": 772.8,
      "y": 344.12
    },
    "ss-DSd-agent-105-opposite-239.255.255.250:1900-10.2.128.15:61159-10.2.192.15:61159": {
      "x": 739.91,
      "y": 322.07
    },
    "ss-DSd-agent-105-opposite-239.255.255.250:1900-10.2.128.15:62242-10.2.192.15:62242": {
      "x": 739.85,
      "y": 321.4
    },
    "ss-DSd-agent-105-opposite-239.255.255.250:1900-10.2.128.15:63298-10.2.192.15:63298": {
      "x": 791.15,
      "y": 321.19
    },
    "ss-DSd-agent-105-opposite-239.255.255.250:1900-10.2.128.15:63615-10.2.192.15:63615": {
      "x": 788.5,
      "y": 307.98
    },
    "ss-DSd-agent-105-opposite-239.255.255.250:1900-10.2.128.15:64766-10.2.192.15:64766": {
      "x": 740.04,
      "y": 323.17
    },
    "ss-DSd-agent-105-opposite-239.255.255.250:1900-10.2.192.15:49406-10.2.192.15:49406": {
      "x": 764.87,
      "y": 293.76
    },
    "ss-DSd-agent-105-opposite-239.255.255.250:1900-10.2.192.15:51195-10.2.192.15:51195": {
      "x": 739.85,
      "y": 321.44
    },
    "ss-DSd-agent-105-opposite-239.255.255.250:1900-10.2.192.15:51308-10.2.192.15:51308": {
      "x": 781.34,
      "y": 339.72
    },
    "ss-DSd-agent-105-opposite-239.255.255.250:1900-10.2.192.15:54333-10.2.192.15:54333": {
      "x": 783.83,
      "y": 301.45
    },
    "ss-DSd-agent-105-opposite-239.255.255.250:1900-10.2.192.15:55489-10.2.192.15:55489": {
      "x": 739.81,
      "y": 320.78
    },
    "ss-DSd-agent-105-opposite-239.255.255.250:1900-10.2.192.15:55497-10.2.192.15:55497": {
      "x": 786.38,
      "y": 334.46
    },
    "ss-DSd-agent-105-opposite-239.255.255.250:1900-10.2.192.15:56006-10.2.192.15:56006": {
      "x": 739.95,
      "y": 316.48
    },
    "ss-DSd-agent-105-opposite-239.255.255.250:1900-10.2.192.15:56935-10.2.192.15:56935": {
      "x": 790.91,
      "y": 315.58
    },
  },

```



```

"ss-Dsd-agent-105-opposite-239.255.255.250:1900-10.2.192.15:59395-10.2.192.15:59395": {
  "x": 739.81,
  "y": 320.77
},
"ss-Dsd-agent-105-opposite-239.255.255.250:1900-10.2.192.15:59734-10.2.192.15:59734": {
  "x": 774.53,
  "y": 295.4
},
"ss-Dsd-agent-105-opposite-239.255.255.250:1900-10.2.192.15:61158-10.2.192.15:61158": {
  "x": 786.32,
  "y": 334.54
},
"ss-Dsd-agent-105-opposite-239.255.255.250:1900-10.2.192.15:62241-10.2.192.15:62241": {
  "x": 787.32,
  "y": 333.06
},
"ss-Dsd-agent-105-opposite-239.255.255.250:1900-10.2.192.15:63297-10.2.192.15:63297": {
  "x": 788.67,
  "y": 308.33
},
"ss-Dsd-agent-105-opposite-239.255.255.250:1900-10.2.192.15:63614-10.2.192.15:63614": {
  "x": 791.15,
  "y": 321.1
},
"ss-Dsd-agent-105-opposite-239.255.255.250:1900-10.2.192.15:64765-10.2.192.15:64765": {
  "x": 786.65,
  "y": 304.87
},
"ss-dsd-core-100-main-225.1.1.1:1599-10.2.128.75:17913-10.2.192.15:9999": {
  "x": 444.76,
  "y": 300.27
},
"ss-dsd-core-100-main-225.1.1.1:1599-10.2.128.75:17913-10.2.192.20:9999": {
  "x": 539.93,
  "y": 302.8
},
"ss-dsd-core-100-main-225.1.1.1:1599-10.2.128.75:35306-10.2.192.15:9999": {
  "x": 456.93,
  "y": 358.77
},
"ss-dsd-core-100-main-225.1.1.1:1599-10.2.128.75:35306-10.2.192.20:9999": {
  "x": 539.95,
  "y": 302.85
},
"ss-dsd-core-100-main-225.1.1.1:1599-10.2.128.75:60041-10.2.128.15:49298": {
  "x": 473.51,
  "y": 272.94
},
"ss-dsd-core-100-main-225.1.1.1:1599-10.2.128.75:60041-10.2.128.75:60041": {
  "x": 543.22,
  "y": 318.98
},
"ss-dsd-core-100-main-225.1.1.1:1599-10.2.128.75:63813-10.2.128.15:58348": {
  "x": 453.39,
  "y": 286.84
},
"ss-dsd-core-100-main-225.1.1.1:1599-10.2.128.75:63813-10.2.128.75:63813": {
  "x": 543.25,
  "y": 321.61
},
"ss-dsd-core-100-main-239.1.1.1:2000-10.2.128.15:64700-10.2.192.15:64700": {
  "x": 543.26,
  "y": 320.65
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:51196-10.2.192.15:51196": {
  "x": 453.25,
  "y": 287
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:51196-10.2.192.20:9999": {
  "x": 539.99,
  "y": 302.96
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:51196-10.2.192.26:9999": {
  "x": 543.26,
  "y": 321.31
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:51196-10.2.192.80:9999": {
  "x": 472.37,
  "y": 368.6
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:51309-10.2.192.15:51309": {
  "x": 453.39,

```

```

        "y": 286.84
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:51309-10.2.192.20:9999": {
        "x": 445.44,
        "y": 298.8
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:51309-10.2.192.26:9999": {
        "x": 517.03,
        "y": 365.83
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:51309-10.2.192.80:9999": {
        "x": 475.97,
        "y": 272.08
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:54334-10.2.192.15:54334": {
        "x": 453.39,
        "y": 286.84
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:54334-10.2.192.20:9999": {
        "x": 444.25,
        "y": 301.49
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:54334-10.2.192.26:9999": {
        "x": 517.03,
        "y": 365.83
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:54334-10.2.192.80:9999": {
        "x": 539.01,
        "y": 300.53
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:55490-10.2.192.15:55490": {
        "x": 453.33,
        "y": 286.9
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:55490-10.2.192.20:9999": {
        "x": 441.8,
        "y": 309.1
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:55490-10.2.192.26:9999": {
        "x": 543.25,
        "y": 321.62
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:55490-10.2.192.80:9999": {
        "x": 445.36,
        "y": 298.97
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:55498-10.2.192.15:55498": {
        "x": 451.63,
        "y": 353.08
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:55498-10.2.192.20:9999": {
        "x": 541.53,
        "y": 307.77
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:55498-10.2.192.26:9999": {
        "x": 501.37,
        "y": 270.46
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:55498-10.2.192.80:9999": {
        "x": 474.88,
        "y": 272.45
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:56007-10.2.192.15:56007": {
        "x": 499.32,
        "y": 371.88
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:56007-10.2.192.20:9999": {
        "x": 541.44,
        "y": 307.45
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:56007-10.2.192.26:9999": {
        "x": 501.32,
        "y": 270.46
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:56007-10.2.192.80:9999": {
        "x": 537.68,
        "y": 344.3
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:59396-10.2.192.15:59396": {
        "x": 453.33,
        "y": 286.91
    },
    },

```

```

"ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:59396-10.2.192.20:9999": {
  "x": 476.53,
  "y": 370.1
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:59396-10.2.192.26:9999": {
  "x": 543.25,
  "y": 321.61
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:59396-10.2.192.80:9999": {
  "x": 537.09,
  "y": 296.58
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:61159-10.2.192.15:61159": {
  "x": 451.61,
  "y": 353.05
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:61159-10.2.192.20:9999": {
  "x": 535.27,
  "y": 293.47
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:61159-10.2.192.26:9999": {
  "x": 501.35,
  "y": 270.46
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:61159-10.2.192.80:9999": {
  "x": 443.03,
  "y": 304.77
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:62242-10.2.192.15:62242": {
  "x": 499.32,
  "y": 371.88
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:62242-10.2.192.20:9999": {
  "x": 517.42,
  "y": 365.61
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:62242-10.2.192.26:9999": {
  "x": 452.97,
  "y": 287.31
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.128.15:62242-10.2.192.80:9999": {
  "x": 541.69,
  "y": 308.41
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:51195-10.2.192.15:51195": {
  "x": 543.24,
  "y": 322.46
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:51195-10.2.192.20:9999": {
  "x": 445.62,
  "y": 298.42
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:51195-10.2.192.26:9999": {
  "x": 543.25,
  "y": 321.72
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:51195-10.2.192.80:9999": {
  "x": 537.6,
  "y": 344.45
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:51308-10.2.192.15:51308": {
  "x": 451.61,
  "y": 353.05
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:51308-10.2.192.20:9999": {
  "x": 537.24,
  "y": 345.14
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:51308-10.2.192.26:9999": {
  "x": 501.38,
  "y": 270.47
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:51308-10.2.192.80:9999": {
  "x": 467.34,
  "y": 366.22
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:54333-10.2.192.15:54333": {
  "x": 451.58,
  "y": 353.02
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:54333-10.2.192.20:9999": {
  "x": 490.36,

```

```

      "y": 269.59
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:54333-10.2.192.26:9999": {
      "x": 501.38,
      "y": 270.47
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:54333-10.2.192.80:9999": {
      "x": 472.51,
      "y": 368.66
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:55489-10.2.192.15:55489": {
      "x": 499.32,
      "y": 371.88
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:55489-10.2.192.20:9999": {
      "x": 540.04,
      "y": 303.09
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:55489-10.2.192.26:9999": {
      "x": 452.05,
      "y": 353.6
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:55489-10.2.192.80:9999": {
      "x": 534.59,
      "y": 292.43
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:55497-10.2.192.15:55497": {
      "x": 453.39,
      "y": 286.84
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:55497-10.2.192.20:9999": {
      "x": 539.72,
      "y": 302.25
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:55497-10.2.192.26:9999": {
      "x": 517.02,
      "y": 365.83
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:55497-10.2.192.80:9999": {
      "x": 537.93,
      "y": 343.8
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:56006-10.2.192.15:56006": {
      "x": 543.24,
      "y": 322.26
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:56006-10.2.192.20:9999": {
      "x": 539.65,
      "y": 302.07
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:56006-10.2.192.26:9999": {
      "x": 543.25,
      "y": 321.65
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:56006-10.2.192.80:9999": {
      "x": 475.58,
      "y": 272.21
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:59395-10.2.192.15:59395": {
      "x": 499.32,
      "y": 371.88
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:59395-10.2.192.20:9999": {
      "x": 512.24,
      "y": 368.21
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:59395-10.2.192.26:9999": {
      "x": 452.04,
      "y": 353.59
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:59395-10.2.192.80:9999": {
      "x": 442.9,
      "y": 305.15
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:61158-10.2.192.15:61158": {
      "x": 543.24,
      "y": 322.46
    },
    "ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:61158-10.2.192.20:9999": {
      "x": 512.21,
      "y": 368.22
    },
    },

```

```

"ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:61158-10.2.192.26:9999": {
  "x": 452.83,
  "y": 287.47
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:61158-10.2.192.80:9999": {
  "x": 472.27,
  "y": 368.56
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:62241-10.2.192.15:62241": {
  "x": 451.7,
  "y": 353.17
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:62241-10.2.192.20:9999": {
  "x": 441.98,
  "y": 308.37
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:62241-10.2.192.26:9999": {
  "x": 543.26,
  "y": 321.55
},
"ss-dsd-core-100-main-239.255.255.250:1900-10.2.192.15:62241-10.2.192.80:9999": {
  "x": 534.98,
  "y": 293.02
},
"ss-dsd-core-100-main-239.4.2.3:5400-10.2.128.36:5400-10.2.128.36:5400": {
  "x": 468.5,
  "y": 366.83
},
"ss-dsd-core-100-main-239.4.2.3:5400-10.2.128.36:5400-10.2.128.37:5401": {
  "x": 543.25,
  "y": 321.62
},
"ss-dsd-core-100-main-239.4.2.3:5400-10.2.128.36:6165-10.2.192.11:9999": {
  "x": 449.08,
  "y": 292.41
},
"ss-dsd-core-100-main-239.4.2.3:5400-10.2.128.36:6165-10.2.192.26:9999": {
  "x": 452.04,
  "y": 353.58
},
"ss-dsd-core-200-main-239.1.1.1:2000-10.2.128.15:64700-10.2.192.15:64700": {
  "x": 907.02,
  "y": 323.55
},
"ss-dsd-core-200-main-239.255.255.250:1900-10.2.128.15:49407-10.2.192.15:49407": {
  "x": 908.06,
  "y": 323.8
},
"ss-dsd-core-200-main-239.255.255.250:1900-10.2.128.15:51196-10.2.192.15:51196": {
  "x": 856.59,
  "y": 340.04
},
"ss-dsd-core-200-main-239.255.255.250:1900-10.2.128.15:51309-10.2.192.15:51309": {
  "x": 946.15,
  "y": 364.77
},
"ss-dsd-core-200-main-239.255.255.250:1900-10.2.128.15:54334-10.2.192.15:54334": {
  "x": 853.86,
  "y": 343.49
},
"ss-dsd-core-200-main-239.255.255.250:1900-10.2.128.15:55498-10.2.192.15:55498": {
  "x": 907.39,
  "y": 423.71
},
"ss-dsd-core-200-main-239.255.255.250:1900-10.2.128.15:56007-10.2.192.15:56007": {
  "x": 946.18,
  "y": 364.92
},
"ss-dsd-core-200-main-239.255.255.250:1900-10.2.128.15:56936-10.2.192.15:56936": {
  "x": 870.32,
  "y": 328.83
},
"ss-dsd-core-200-main-239.255.255.250:1900-10.2.128.15:59735-10.2.192.15:59735": {
  "x": 911.65,
  "y": 324.84
},
"ss-dsd-core-200-main-239.255.255.250:1900-10.2.128.15:61159-10.2.192.15:61159": {
  "x": 907.74,
  "y": 423.62
},
"ss-dsd-core-200-main-239.255.255.250:1900-10.2.128.15:62242-10.2.192.15:62242": {
  "x": 946.24,

```

```

        "y": 365.27
    },
    "ss-dsd-core-200-main-239.255.255.250:1900-10.2.128.15:63298-10.2.192.15:63298": {
        "x": 903.7,
        "y": 322.9
    },
    "ss-dsd-core-200-main-239.255.255.250:1900-10.2.128.15:63615-10.2.192.15:63615": {
        "x": 877.52,
        "y": 325.49
    },
    "ss-dsd-core-200-main-239.255.255.250:1900-10.2.128.15:64766-10.2.192.15:64766": {
        "x": 852.12,
        "y": 346.05
    },
    "ss-dsd-core-200-main-239.255.255.250:1900-10.2.192.15:49406-10.2.192.15:49406": {
        "x": 853.41,
        "y": 344.12
    },
    "ss-dsd-core-200-main-239.255.255.250:1900-10.2.192.15:51195-10.2.192.15:51195": {
        "x": 946.35,
        "y": 365.99
    },
    "ss-dsd-core-200-main-239.255.255.250:1900-10.2.192.15:51308-10.2.192.15:51308": {
        "x": 854.03,
        "y": 343.25
    },
    "ss-dsd-core-200-main-239.255.255.250:1900-10.2.192.15:54333-10.2.192.15:54333": {
        "x": 946.18,
        "y": 364.93
    },
    "ss-dsd-core-200-main-239.255.255.250:1900-10.2.192.15:55497-10.2.192.15:55497": {
        "x": 850.99,
        "y": 347.91
    },
    "ss-dsd-core-200-main-239.255.255.250:1900-10.2.192.15:56006-10.2.192.15:56006": {
        "x": 854.06,
        "y": 343.21
    },
    "ss-dsd-core-200-main-239.255.255.250:1900-10.2.192.15:56935-10.2.192.15:56935": {
        "x": 906.05,
        "y": 323.34
    },
    "ss-dsd-core-200-main-239.255.255.250:1900-10.2.192.15:59734-10.2.192.15:59734": {
        "x": 855.86,
        "y": 340.9
    },
    "ss-dsd-core-200-main-239.255.255.250:1900-10.2.192.15:61158-10.2.192.15:61158": {
        "x": 850.76,
        "y": 348.31
    },
    "ss-dsd-core-200-main-239.255.255.250:1900-10.2.192.15:62241-10.2.192.15:62241": {
        "x": 858.48,
        "y": 337.97
    },
    "ss-dsd-core-200-main-239.255.255.250:1900-10.2.192.15:63297-10.2.192.15:63297": {
        "x": 870.65,
        "y": 328.64
    },
    "ss-dsd-core-200-main-239.255.255.250:1900-10.2.192.15:63614-10.2.192.15:63614": {
        "x": 908.37,
        "y": 323.88
    },
    "ss-dsd-core-200-main-239.255.255.250:1900-10.2.192.15:64765-10.2.192.15:64765": {
        "x": 877.67,
        "y": 325.43
    },
    "ss-dsd-win10-100-opposite-225.1.1.1:1599-10.2.128.75:17913-10.2.192.20:9999": {
        "x": 617.08,
        "y": -108.29
    },
    "ss-dsd-win10-100-opposite-225.1.1.1:1599-10.2.128.75:35306-10.2.192.20:9999": {
        "x": 617.42,
        "y": -108.29
    },
    "ss-dsd-win10-100-opposite-239.255.255.250:1900-10.2.128.15:51196-10.2.192.20:9999": {
        "x": 616.64,
        "y": -108.31
    },
    "ss-dsd-win10-100-opposite-239.255.255.250:1900-10.2.128.15:51309-10.2.192.20:9999": {
        "x": 601.21,
        "y": -153.76
    },
    },

```

```

"ss-dsd-win10-100-opposite-239.255.255.250:1900-10.2.128.15:54334-10.2.192.20:9999": {
  "x": 635.03,
  "y": -152.96
},
"ss-dsd-win10-100-opposite-239.255.255.250:1900-10.2.128.15:55490-10.2.192.20:9999": {
  "x": 617.72,
  "y": -108.29
},
"ss-dsd-win10-100-opposite-239.255.255.250:1900-10.2.128.15:55498-10.2.192.20:9999": {
  "x": 601.06,
  "y": -153.64
},
"ss-dsd-win10-100-opposite-239.255.255.250:1900-10.2.128.15:56007-10.2.192.20:9999": {
  "x": 634.82,
  "y": -153.15
},
"ss-dsd-win10-100-opposite-239.255.255.250:1900-10.2.128.15:59396-10.2.192.20:9999": {
  "x": 616.47,
  "y": -108.31
},
"ss-dsd-win10-100-opposite-239.255.255.250:1900-10.2.128.15:61159-10.2.192.20:9999": {
  "x": 601.12,
  "y": -153.69
},
"ss-dsd-win10-100-opposite-239.255.255.250:1900-10.2.128.15:62242-10.2.192.20:9999": {
  "x": 601.18,
  "y": -153.74
},
"ss-dsd-win10-100-opposite-239.255.255.250:1900-10.2.192.15:51195-10.2.192.20:9999": {
  "x": 601.17,
  "y": -153.73
},
"ss-dsd-win10-100-opposite-239.255.255.250:1900-10.2.192.15:51308-10.2.192.20:9999": {
  "x": 616.53,
  "y": -108.31
},
"ss-dsd-win10-100-opposite-239.255.255.250:1900-10.2.192.15:54333-10.2.192.20:9999": {
  "x": 617.56,
  "y": -108.29
},
"ss-dsd-win10-100-opposite-239.255.255.250:1900-10.2.192.15:55489-10.2.192.20:9999": {
  "x": 635.08,
  "y": -152.91
},
"ss-dsd-win10-100-opposite-239.255.255.250:1900-10.2.192.15:55497-10.2.192.20:9999": {
  "x": 634.89,
  "y": -153.09
},
"ss-dsd-win10-100-opposite-239.255.255.250:1900-10.2.192.15:56006-10.2.192.20:9999": {
  "x": 601.05,
  "y": -153.63
},
"ss-dsd-win10-100-opposite-239.255.255.250:1900-10.2.192.15:59395-10.2.192.20:9999": {
  "x": 601.13,
  "y": -153.7
},
"ss-dsd-win10-100-opposite-239.255.255.250:1900-10.2.192.15:61158-10.2.192.20:9999": {
  "x": 616.47,
  "y": -108.31
},
"ss-dsd-win10-100-opposite-239.255.255.250:1900-10.2.192.15:62241-10.2.192.20:9999": {
  "x": 616.45,
  "y": -108.31
},
"ss-dsd-win10-100-opposite-239.4.2.3:5400-10.2.128.36:5400-10.2.128.36:5400": {
  "x": 617.69,
  "y": -108.29
},
"ss-dsd-win10-101-opposite-239.255.255.250:1900-10.2.128.15:51196-10.2.192.26:9999": {
  "x": 1721.4,
  "y": 991.44
},
"ss-dsd-win10-101-opposite-239.255.255.250:1900-10.2.128.15:51309-10.2.192.26:9999": {
  "x": 1721.4,
  "y": 991.44
},
"ss-dsd-win10-101-opposite-239.255.255.250:1900-10.2.128.15:54334-10.2.192.26:9999": {
  "x": 1698.44,
  "y": 948.92
},
"ss-dsd-win10-101-opposite-239.255.255.250:1900-10.2.128.15:55490-10.2.192.26:9999": {
  "x": 1721.4,

```

```

        "y": 991.44
    },
    "ss-dsd-win10-101-opposite-239.255.255.250:1900-10.2.128.15:55498-10.2.192.26:9999": {
        "x": 1698.44,
        "y": 948.92
    },
    "ss-dsd-win10-101-opposite-239.255.255.250:1900-10.2.128.15:56007-10.2.192.26:9999": {
        "x": 1731.15,
        "y": 944.11
    },
    "ss-dsd-win10-101-opposite-239.255.255.250:1900-10.2.128.15:59396-10.2.192.26:9999": {
        "x": 1698.44,
        "y": 948.92
    },
    "ss-dsd-win10-101-opposite-239.255.255.250:1900-10.2.128.15:61159-10.2.192.26:9999": {
        "x": 1721.4,
        "y": 991.44
    },
    "ss-dsd-win10-101-opposite-239.255.255.250:1900-10.2.128.15:62242-10.2.192.26:9999": {
        "x": 1698.44,
        "y": 948.92
    },
    "ss-dsd-win10-101-opposite-239.255.255.250:1900-10.2.192.15:51195-10.2.192.26:9999": {
        "x": 1698.44,
        "y": 948.92
    },
    "ss-dsd-win10-101-opposite-239.255.255.250:1900-10.2.192.15:51308-10.2.192.26:9999": {
        "x": 1698.44,
        "y": 948.92
    },
    "ss-dsd-win10-101-opposite-239.255.255.250:1900-10.2.192.15:54333-10.2.192.26:9999": {
        "x": 1721.4,
        "y": 991.44
    },
    "ss-dsd-win10-101-opposite-239.255.255.250:1900-10.2.192.15:55489-10.2.192.26:9999": {
        "x": 1698.44,
        "y": 948.92
    },
    "ss-dsd-win10-101-opposite-239.255.255.250:1900-10.2.192.15:55497-10.2.192.26:9999": {
        "x": 1721.4,
        "y": 991.44
    },
    "ss-dsd-win10-101-opposite-239.255.255.250:1900-10.2.192.15:56006-10.2.192.26:9999": {
        "x": 1693.26,
        "y": 974.1
    },
    "ss-dsd-win10-101-opposite-239.255.255.250:1900-10.2.192.15:59395-10.2.192.26:9999": {
        "x": 1721.4,
        "y": 991.44
    },
    "ss-dsd-win10-101-opposite-239.255.255.250:1900-10.2.192.15:61158-10.2.192.26:9999": {
        "x": 1698.44,
        "y": 948.92
    },
    "ss-dsd-win10-101-opposite-239.255.255.250:1900-10.2.192.15:62241-10.2.192.26:9999": {
        "x": 1721.4,
        "y": 991.44
    },
    "ss-dsd-win10-101-opposite-239.4.2.3:5400-10.2.128.36:5400-10.2.128.37:5401": {
        "x": 1721.4,
        "y": 991.44
    },
    "ss-dsd-win10-101-opposite-239.4.2.3:5400-10.2.128.36:6165-10.2.192.26:9999": {
        "x": 1698.44,
        "y": 948.92
    },
    "ss-dsd-win11-100-opposite-239.255.255.250:1900-10.2.128.15:51196-10.2.192.80:9999": {
        "x": 715.1,
        "y": -8.41
    },
    "ss-dsd-win11-100-opposite-239.255.255.250:1900-10.2.128.15:51309-10.2.192.80:9999": {
        "x": 713.64,
        "y": -8.6
    },
    "ss-dsd-win11-100-opposite-239.255.255.250:1900-10.2.128.15:54334-10.2.192.80:9999": {
        "x": 715.29,
        "y": -8.4
    },
    "ss-dsd-win11-100-opposite-239.255.255.250:1900-10.2.128.15:55490-10.2.192.80:9999": {
        "x": 737,
        "y": -50.95
    },
    },

```



```

"ss-dsd-win11-100-opposite-239.255.255.250:1900-10.2.128.15:55498-10.2.192.80:9999": {
  "x": 702.98,
  "y": -55.11
},
"ss-dsd-win11-100-opposite-239.255.255.250:1900-10.2.128.15:56007-10.2.192.80:9999": {
  "x": 713.97,
  "y": -8.55
},
"ss-dsd-win11-100-opposite-239.255.255.250:1900-10.2.128.15:59396-10.2.192.80:9999": {
  "x": 702.98,
  "y": -55.11
},
"ss-dsd-win11-100-opposite-239.255.255.250:1900-10.2.128.15:61159-10.2.192.80:9999": {
  "x": 737.03,
  "y": -50.91
},
"ss-dsd-win11-100-opposite-239.255.255.250:1900-10.2.128.15:62242-10.2.192.80:9999": {
  "x": 713.56,
  "y": -8.62
},
"ss-dsd-win11-100-opposite-239.255.255.250:1900-10.2.192.15:51195-10.2.192.80:9999": {
  "x": 736.89,
  "y": -51.07
},
"ss-dsd-win11-100-opposite-239.255.255.250:1900-10.2.192.15:51308-10.2.192.80:9999": {
  "x": 703.14,
  "y": -55.22
},
"ss-dsd-win11-100-opposite-239.255.255.250:1900-10.2.192.15:54333-10.2.192.80:9999": {
  "x": 736.87,
  "y": -51.09
},
"ss-dsd-win11-100-opposite-239.255.255.250:1900-10.2.192.15:55489-10.2.192.80:9999": {
  "x": 715.49,
  "y": -8.38
},
"ss-dsd-win11-100-opposite-239.255.255.250:1900-10.2.192.15:55497-10.2.192.80:9999": {
  "x": 714.02,
  "y": -8.55
},
"ss-dsd-win11-100-opposite-239.255.255.250:1900-10.2.192.15:56006-10.2.192.80:9999": {
  "x": 702.99,
  "y": -55.12
},
"ss-dsd-win11-100-opposite-239.255.255.250:1900-10.2.192.15:59395-10.2.192.80:9999": {
  "x": 713.83,
  "y": -8.57
},
"ss-dsd-win11-100-opposite-239.255.255.250:1900-10.2.192.15:61158-10.2.192.80:9999": {
  "x": 715.2,
  "y": -8.4
},
"ss-dsd-win11-100-opposite-239.255.255.250:1900-10.2.192.15:62241-10.2.192.80:9999": {
  "x": 703.07,
  "y": -55.17
},
"swxtch-10.2.128.10": {
  "x": 491.83,
  "y": 321
},
"swxtch-10.5.1.6": {
  "x": 895.5,
  "y": 373.67
},
"vm-10.2.128.15": {
  "x": 758.89,
  "y": 131.06
},
"vm-10.2.192.15": {
  "x": 658.22,
  "y": 324
},
"xnic-10.2.128.104": {
  "x": 1217.66,
  "y": 466
},
"xnic-10.2.128.106": {
  "x": 917.66,
  "y": 166
},
"xnic-10.2.128.13": {
  "x": 1017.66,

```

```

        "y": 266
      },
      "xnic-10.2.128.15": {
        "x": 765.49,
        "y": 319.47
      },
      "xnic-10.2.128.27": {
        "x": 1917.66,
        "y": 1166
      },
      "xnic-10.2.128.36": {
        "x": 617.66,
        "y": -134
      },
      "xnic-10.2.128.37": {
        "x": 1717.66,
        "y": 966
      },
      "xnic-10.2.128.74": {
        "x": 717.66,
        "y": -34
      },
      "xnic-10.2.128.75": {
        "x": 1517.66,
        "y": 766
      }
    },
    "aliases": null,
    "nodes": null
  },
  "components": {
    "entries": {
      "10.2.128.10": {
        "name": "dsd-core-100",
        "componentKind": {
          "code": "swxtch",
          "displayName": "swXtch"
        },
        "environment": {
          "hostname": "dsd-core-100",
          "cloud": "AZURE",
          "osDistribution": "Ubuntu 20.04",
          "region": "eastus",
          "instanceType": "Standard_D8as_v4"
        },
        "interfaces": {
          "eth0": {
            "ip": "10.2.128.10",
            "name": "eth0"
          },
          "eth1": {
            "ip": "10.2.192.116",
            "name": "eth1"
          }
        }
      },
      "10.2.128.104": {
        "name": "aks-nodepool1-23164585-vmss00001Z",
        "componentKind": {
          "code": "xnic",
          "displayName": "xNIC"
        },
        "environment": {
          "hostname": "aks-nodepool1-23164585-vmss00001Z",
          "cloud": "AZURE",
          "osDistribution": "Ubuntu 20.04",
          "region": "eastus",
          "instanceType": "Standard_Ds2_v2"
        },
        "interfaces": {
          "cilium_host": {
            "ip": "10.0.0.54",
            "name": "cilium_host"
          },
          "eth0": {
            "ip": "10.2.128.104",
            "name": "eth0"
          },
          "swxtch-tun0": {
            "ip": "172.30.0.104",
            "name": "swxtch-tun0"
          }
        }
      }
    }
  }
}

```

```

    },
    "10.2.128.106": {
      "name": "aks-nodepool1-23164585-vmss00001Y",
      "componentKind": {
        "code": "xnic",
        "displayName": "xNIC"
      },
      "environment": {
        "hostname": "aks-nodepool1-23164585-vmss00001Y",
        "cloud": "AZURE",
        "osDistribution": "Ubuntu 20.04",
        "region": "eastus",
        "instanceType": "Standard_Ds2_v2"
      },
      "interfaces": {
        "eth0": {
          "ip": "10.2.128.106",
          "name": "eth0"
        },
        "swtch-tun0": {
          "ip": "172.30.0.106",
          "name": "swtch-tun0"
        }
      }
    },
    "10.2.128.13": {
      "name": "DSd-agent-102",
      "componentKind": {
        "code": "xnic",
        "displayName": "xNIC"
      },
      "environment": {
        "hostname": "DSd-agent-102",
        "cloud": "AZURE",
        "osDistribution": "Ubuntu 20.04",
        "region": "eastus",
        "instanceType": "Standard_D4s_v4"
      },
      "interfaces": {
        "eth0": {
          "ip": "10.2.128.13",
          "name": "eth0"
        },
        "eth1": {
          "ip": "10.2.192.11",
          "name": "eth1"
        },
        "swtch-tun0": {
          "ip": "172.30.0.13",
          "name": "swtch-tun0"
        }
      }
    },
    "10.2.128.15": {
      "name": "DSd-agent-105",
      "componentKind": {
        "code": "xnic",
        "displayName": "xNIC"
      },
      "environment": {
        "hostname": "DSd-agent-105",
        "cloud": "AZURE",
        "osDistribution": "Windows Server 2019 Datacenter - Microsoft Windows [Version 10.0.17763.6189]",
        "region": "eastus",
        "instanceType": "Standard_D8s_v4"
      },
      "interfaces": {
        "Ethernet": {
          "ip": "10.2.192.15",
          "name": "Ethernet"
        },
        "Ethernet 2": {
          "ip": "10.2.128.15",
          "name": "Ethernet 2"
        }
      }
    },
    "10.2.128.27": {
      "name": "DSd-agent-101",
      "componentKind": {
        "code": "xnic",

```

```

        "displayName": "xNIC"
    },
    "environment": {
        "hostname": "DSd-agent-101",
        "cloud": "AZURE",
        "osDistribution": "Ubuntu 20.04",
        "region": "eastus",
        "instanceType": "Standard_D4ds_v4"
    },
    "interfaces": {
        "eth0": {
            "ip": "10.2.128.27",
            "name": "eth0"
        },
        "eth1": {
            "ip": "10.2.192.24",
            "name": "eth1"
        },
        "swtch-tun0": {
            "ip": "172.30.0.27",
            "name": "swtch-tun0"
        }
    }
},
"10.2.128.75": {
    "name": "DSd-agent-104",
    "componentKind": {
        "code": "xnic",
        "displayName": "xNIC"
    },
    "environment": {
        "hostname": "DSd-agent-104",
        "cloud": "AZURE",
        "osDistribution": "Windows Server 2019 Datacenter - Microsoft Windows [Version 10.0.17763.6189]",
        "region": "eastus",
        "instanceType": "Standard_D8s_v4"
    },
    "interfaces": {
        "Ethernet": {
            "ip": "10.2.128.75",
            "name": "Ethernet"
        },
        "Ethernet 2": {
            "ip": "10.2.192.82",
            "name": "Ethernet 2"
        }
    }
},
"10.2.128.98": {
    "name": "DSd-bridge-100",
    "componentKind": {
        "code": "bridge",
        "displayName": "Bridge"
    },
    "environment": {
        "hostname": "DSd-bridge-100",
        "cloud": "AZURE",
        "osDistribution": "Ubuntu 20.04",
        "region": "eastus",
        "instanceType": "Standard_D4ds_v4"
    },
    "interfaces": {
        "eth0": {
            "ip": "10.2.128.98",
            "name": "eth0"
        },
        "eth1": {
            "ip": "10.2.192.90",
            "name": "eth1"
        }
    }
},
"10.5.1.4": {
    "name": "DSd-agent-201",
    "componentKind": {
        "code": "xnic",
        "displayName": "xNIC"
    },
    "environment": {
        "hostname": "DSd-agent-201",
        "cloud": "AZURE",
        "osDistribution": "Ubuntu 20.04",

```

```

        "region": "eastus",
        "instanceType": "Standard_D4s_v4"
    },
    "interfaces": {
        "eth0": {
            "ip": "10.5.1.4",
            "name": "eth0"
        },
        "eth1": {
            "ip": "10.5.2.4",
            "name": "eth1"
        },
        "swxtch-tun0": {
            "ip": "172.30.0.4",
            "name": "swxtch-tun0"
        }
    }
},
"10.5.1.6": {
    "name": "dsd-core-200",
    "componentKind": {
        "code": "swxtch",
        "displayName": "swXtch"
    },
    "environment": {
        "hostname": "dsd-core-200",
        "cloud": "AZURE",
        "osDistribution": "Ubuntu 20.04",
        "region": "eastus",
        "instanceType": "Standard_D8s_v4"
    },
    "interfaces": {
        "eth0": {
            "ip": "10.5.1.6",
            "name": "eth0"
        },
        "eth1": {
            "ip": "10.5.2.6",
            "name": "eth1"
        }
    }
}
}
},
"vms": null,
"aliases": {
    "components": {},
    "sockets": {
        "10.2.128.36:5400": "10.2.128.36_win-10-100",
        "10.2.128.37:5401": "10_2_128_37_5401_Win-10_101"
    },
    "streams": {
        "225.1.1.1:1599": "SRT_L_I_225_1_1_1_1599",
        "239.4.2.3:5400": "SRT_C_239_4_2_3_1400"
    }
},
"bridges": {
    "10.2.128.98": {
        "ctrlIp": "10.2.128.98",
        "hostname": "DSd-bridge-100",
        "dataIp": "10.2.192.90",
        "gatewayMac": "12:34:56:78:9a:bc",
        "lastTime": "2024-08-19T16:43:15.8684573Z",
        "mcGroups": []
    }
},
"topology": {
    "graph": {
        "nClusterLinks": 4,
        "minPps": 1
    }
},
"tachyonlive": null
}

```

General

The general tab shows information about the cloudSwXtch networking and entitlements.

Cluster > Topology Graph > Settings

Settings

General

Mesh

High Availability

Protocol Fanout

Channels

Streams

Aliases

Components

Tachyon Live

Summary

Version: dev.44af38

Size: 4

Control IP: 172.41.128.25

Control Port: 10802

Subnet Data Prefix: 172.41.21.0/24

Subnet Control Prefix: 172.41.128.0/22

Gateway IP: 172.41.21.1

Licensing: License File

License expiration: 2033-Nov-8

Entitlements

Max Clients	Max Bridges	Max Bandwidth (Mbps)	Enabled Features						
			Mesh	High Availability	Protocol Fanout	PTP	wXcked Eye	Major Version Update	Tachyon Live
30	1	50000	✓	✓	✓	✓	✓	✓	✗

Hardware

Expand to see the portion of the Settings call detailing information on the General tab —>

Bash

Copy

```
{
  "general": {
    "hostname": "dsd-core-100",
    "switchName": "dsd-core-100",
    "version": "dev.7f1520",
    "numCores": 1,
    "entitlements": {
      "maxClientCount": 40,
      "maxBridgeCount": 2,
      "bandwidthMbps": 25000,
      "enableMesh": true,
      "enableUnicast": true,
      "enableHA": true,
      "enableClockSync": true,
      "enableBridge": true,
      "enablewXckedEye": true,
      "enableAllowsMajorVersionUpdate": true,
      "enableTachyonLive": false
    },
    "subnetDataPrefix": "10.2.192.0/22",
    "subnetCtrlPrefix": "10.2.128.0/22",
    "dataGatewayIp": "10.2.192.1",
    "ctrlIp": "10.2.128.10",
    "ctrlPort": 10802,
    "gatewayMacAddr": "12:34:56:78:9a:bc",
    "replInfo": {
      "ctrlIp": "127.0.0.1",
      "ctrlPort": 9996,
      "dataIp": "10.2.192.116",
      "dataPort": 9999,
      "dataMac": "AA06VA83"
    },
    "numClientsConnected": 8,
    "hardwareInfo": {
      "control": {
        "meta": {
          "mac": "00:0d:3a:19:cc:e1",
          "ipAddress": "10.2.128.10",
          "ipBroadcast": "10.2.131.255",
          "ipSubnet": "10.2.128.0",
          "subnetMask": "255.255.252.0",
          "mtu": null,
          "driver": null,
          "masterOf": null,
          "ifIndex": null,

```

©2024 IEX Group, Inc. and its subsidiaries, including swXtch.io. All rights reserved.

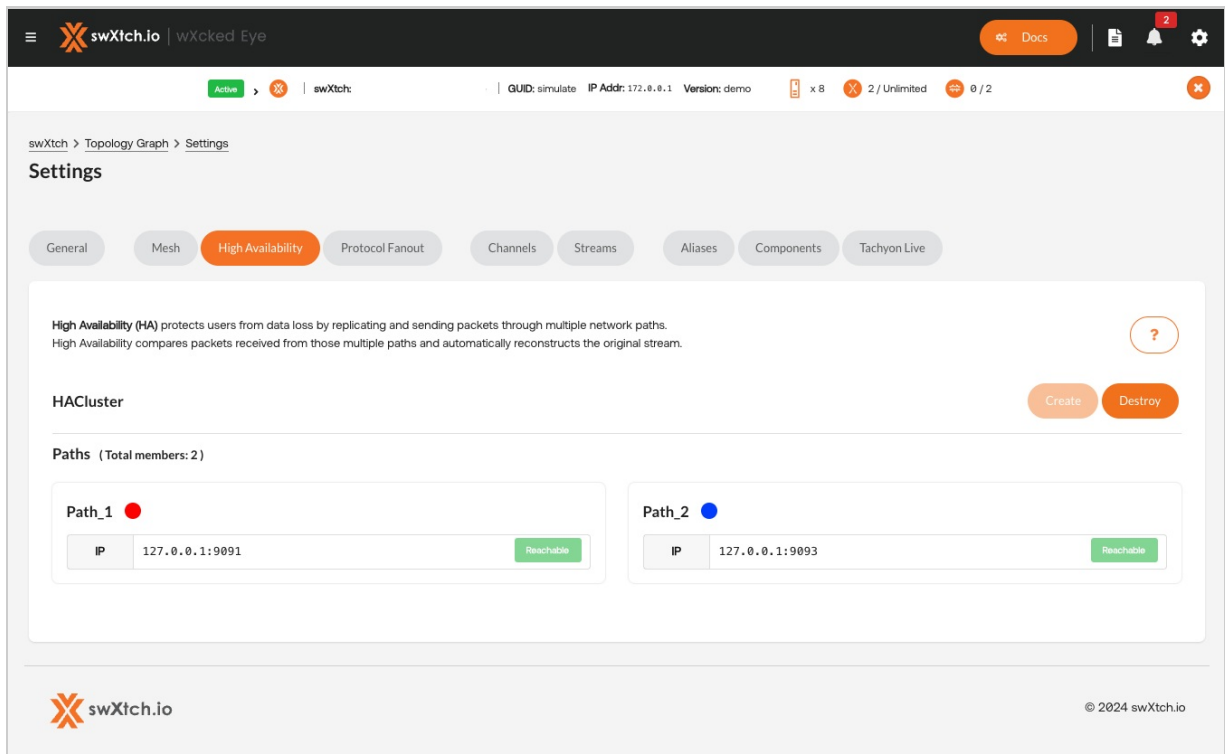
```

        "vpc": null,
        "publicIpAddress": null
    },
    "os": {
        "mac": "00:0d:3a:19:cc:e1",
        "ipAddress": "10.2.128.10",
        "ipBroadcast": "10.2.131.255",
        "ipSubnet": "10.2.128.0",
        "subnetMask": "255.255.252.0",
        "mtu": 1500,
        "driver": "hv_netvsc",
        "masterOf": null,
        "ifIndex": 2,
        "vpc": null,
        "publicIpAddress": null,
        "name": "eth0",
        "pciAddress": "",
        "state": "up"
    },
    "computed": {
        "isPreferredControlNic": true,
        "isPreferredDataNic": false
    }
},
"data": {
    "meta": {
        "mac": "00:0d:3a:54:0f:37",
        "ipAddress": "10.2.192.116",
        "ipBroadcast": "10.2.195.255",
        "ipSubnet": "10.2.192.0",
        "subnetMask": "255.255.252.0",
        "mtu": null,
        "driver": null,
        "masterOf": null,
        "ifIndex": null,
        "vpc": null,
        "publicIpAddress": null
    },
    "os": {
        "mac": "00:0d:3a:54:0f:37",
        "ipAddress": "10.2.192.116",
        "ipBroadcast": "10.2.195.255",
        "ipSubnet": "10.2.192.0",
        "subnetMask": "255.255.252.0",
        "mtu": 1500,
        "driver": "mlx5_core",
        "masterOf": "enP42685s2",
        "ifIndex": 3,
        "vpc": null,
        "publicIpAddress": null,
        "name": "eth1",
        "pciAddress": "a6bd:00:02.0",
        "state": "up"
    },
    "computed": {
        "isPreferredControlNic": false,
        "isPreferredDataNic": true
    }
}
},
"httpPort": 80
},

```

High Availability

The High Availability tab shows the cloudSwXtches that are configured to be in a HA 2022-7 configuration. Note that both High Availability and Mesh are configured here for example purposes. Mesh and High Availability are, however, not compatible with the same cloudSwXtches.



Below is a portion of the Settings call response detailing information on High Availability:

```
Bash Copy

"ha": {
  "config": {
    "uid": "cf81bcb-2afd-ac36-72d0-a67ec56c742c",
    "name": "100-200HA",
    "paths": [
      {
        "name": "100",
        "color": "#fc0303",
        "swxtches": [
          "10.2.128.10:80"
        ]
      },
      {
        "name": "200",
        "color": "#033dfc",
        "swxtches": [
          "10.5.1.6:80"
        ]
      }
    ]
  },
  "membersData": {
    "10.2.128.10:80": {
      "ipAddr": "10.2.128.10:80",
      "isAlive": true,
      "host": null
    },
    "10.5.1.6:80": {
      "ipAddr": "10.5.1.6:80",
      "isAlive": true,
      "host": null
    }
  }
},
}
```

Create HA

swxtch/ha/v1/create

POST

swxtch/ha/v1/create

- Creates an High Availability configuration

URL:

Bash

Copy

```
curl http://<cloudSwXtch-control-IP>/swxtch/ha/v1/create
```

Example URL:

http://10.2.128.10/swxtch/ha/v1/create

Example Request Body:

Bash

Copy

```
{
  "uid": "0",
  "name": "Donna-ha",
  "paths": [
    {
      "name": "path_1",
      "swxtches": [
        "10.2.128.10:80"
      ]
    },
    {
      "name": "path_2",
      "swxtches": [
        "10.1.1.6:80"
      ]
    },
    {
      "name": "path_3",
      "swxtches": [
        "10.5.1.6:80"
      ]
    }
  ]
}
```

Responses:

200 - successful operation

Example Value:

Bash

Copy

```
{
  "joinClusterResultItems": null,
  "error": null
}
```

Get HA List

swxtch/ha/v1/show



GET

swxtch/ha/v1/show

- Returns a list of cloudSwXtches connected in an HA configuration.

URL:

Bash

Copy

```
curl http://<cloudSwXtch-control-IP>/swxtch/ha/v1/show
```

Example URL:

http://10.2.128.10/swxtch/ha/v1/show

Example Request Body:

None

Responses:

200 - successful operation

Example Value →



Bash

Copy

```
{
  "clusterConfig": {
    "uid": "377a7cd7-98ed-47e3-27c9-fcf123de530a",
    "name": "Donna-ha",
    "paths": [
      {
        "name": "path_1",
        "color": null,
        "swxtches": [
          "10.2.128.10:80"
        ]
      },
      {
        "name": "path_2",
        "color": null,
        "swxtches": [
          "10.1.1.6:80"
        ]
      },
      {
        "name": "path_3",
        "color": null,
        "swxtches": [
          "10.5.1.6:80"
        ]
      }
    ]
  },
  "memberData": {
    "10.1.1.6:80": {
      "ipAddr": "10.1.1.6:80",
      "isAlive": true,
      "host": null
    },
    "10.2.128.10:80": {
      "ipAddr": "10.2.128.10:80",
      "isAlive": true,
      "host": null
    },
    "10.5.1.6:80": {
      "ipAddr": "10.5.1.6:80",
      "isAlive": true,
      "host": null
    }
  }
}
```

Destroy HA

swxtch/ha/v1/destroy

^

POST

swxtch/ha/v1/destroy

- Destroys a cloudSwXtch from the HA configuration

URL:

Bash

Copy

```
curl http://<cloudSwXtch-control-IP>/swxtch/ha/v1/destroy
```

Example URL:

```
http://10.2.128.10/swxtch/ha/v1/destroy
```

Example Request Body:

```
None
```

Responses:

```
200 - successful operation
```

Example Value:

BashCopy

```
{
  "Message": "HA destroyed successfully"
}
```

Protocol Fanout Adaptors

The Protocol Fanout tab shows the cloudSwXtches that are configured for Protocol Fanout: multicast, UDP (Unicast), SRT Caller/Listener, and RIST Caller/Listener.

swXtch.io | wXcked Eye

Active | swXtch: ip-172-41-128-25 | GUID: 1-00085b8-ab407e64 | IP Addr: 172.41.128.25 | Version: dev-2024-08 | 4 / 30 | 0 / 1

Cluster > Settings

Settings

GeneralMeshHigh AvailabilityProtocol FanoutChannelsStreamsAliasesComponentsTachyon Live

Protocol Fanout enables users to configure **unicast adaptors** to send and receive traffic through the swXtch using different protocols (UDP, SRT and RIST).
Ingress direction describes receiving unicast traffic in any supported protocol and converting it into a multicast stream on the swXtch.
Egress describes sending copies of a single input stream in any supported protocol to multiple destinations.
This gives each destination the option of being a different protocol from the input stream (for instance, an UDP input being sent to a set of multicast destinations and, additionally, to one using SRT).

Adaptors (8)Filter by ProtocolDirectionAdd Adaptor

Component	Protocol	Direction	Stream		Node		Listener Port	Status	Action
			IP: Port	Name	IP: Port	Name			
ip-172-41-128-25	SRT Listener	Egress	225.1.1.1:5500	SRTListenerIngress			5888	Active	editdelete
ip-172-41-128-25	SRT Listener	Ingress	225.1.1.1:5500	SRTListenerIngress			1599	Active	editdelete
ip-172-41-128-25	SRT Caller	Egress	225.1.2.2:5000	SRT_Caller	172.41.128.113:2222	dr-s2-ore-Win2019_1		Inactive	editdelete
ip-172-41-128-25	RIST Listener	Egress	225.1.2.3:5000	RistIngress			5643	Inactive	editdelete
ip-172-41-128-25	RIST Caller	Egress	225.1.2.4:5000	RistCaller	172.41.129.63:1598	dr-s2-ore-Win2019_2 Egress Rist Cal		Inactive	editdelete

swXtch.io

© 2024 swXtch.io

Below is a portion of the Settings call response detailing information on Protocol Fanout.

```
"fanout": {
  "16704273857340480058": {
    "id": "16704273857340480058",
    "protocol": "srt-caller",
    "direction": "egress",
    "streamIp": "239.4.2.3",
    "streamPort": 5400,
    "nodeIp": "10.2.128.37",
    "nodePort": 5401,
    "listenerPort": null,
    "options": []
  },
  "4777854487999137884": {
    "id": "4777854487999137884",
    "protocol": "srt-caller",
    "direction": "ingress",
    "streamIp": "239.4.2.3",
    "streamPort": 5400,
    "nodeIp": "10.2.128.36",
    "nodePort": 5400,
    "listenerPort": null,
    "options": []
  },
  "5062225067104987852": {
    "id": "5062225067104987852",
    "protocol": "srt-listener",
    "direction": "egress",
    "streamIp": "225.1.1.1",
    "streamPort": 1599,
    "nodeIp": null,
    "nodePort": null,
    "listenerPort": 6000,
    "options": []
  },
  "9489213241930010873": {
    "id": "9489213241930010873",
    "protocol": "srt-listener",
    "direction": "ingress",
    "streamIp": "225.1.1.1",
    "streamPort": 1599,
    "nodeIp": null,
    "nodePort": null,
    "listenerPort": 1599,
    "options": []
  }
},
}
```

swxtch/protocols/v1/adaptors/add

POST

swxtch/protocols/v1/adaptors/add

- Add an adaptor and specify the protocol and direction for fanout

URL:

PowerShell

Copy

```
http://<cloudSwXtch-Control-IP>/swxtch/protocols/v1/adaptors/add
```

Example URL:

```
http://10.2.128.10/swxtch/protocols/v1/adaptors/add
```

Request Body:

[SRT available options](#)[RIST available options](#)

- UDP - Ingress

PowerShell

Copy

```
{
  "protocol": "udp",
  "direction": "ingress",
  "streamIp": "224.0.0.1",
  "streamPort": 4000,
  "listenerPort": 3500
}
```

- UDP - Egress

PowerShell

Copy

```
{
  "protocol": "udp",
  "direction": "egress",
  "streamIp": "224.0.0.2",
  "streamPort": 4001,
  "nodeIp": "10.0.0.1",
  "nodePort": 3501,
  "nodeMac": "00:22:48:37:8a:f9"
}
```

- SRT Caller - Ingress

PowerShell

Copy

```
{
  "protocol": "srt-caller",
  "direction": "ingress",
  "streamIp": "224.0.0.3",
  "streamPort": 4002,
  "nodeIp": "10.0.0.2",
  "nodePort": 3502
}
```

- SRT Caller - Egress

PowerShell

Copy

```
{
  "protocol": "srt-caller",
  "direction": "egress",
  "streamIp": "224.0.0.4",
  "streamPort": 4003,
  "nodeIp": "10.0.0.3",
  "nodePort": 3503
}
```

- SRT Listener - Ingress

PowerShell

Copy

```
{
  "protocol": "srt-listener",
  "direction": "ingress",
  "streamIp": "224.0.0.5",
  "streamPort": 4004,
  "listenerPort": 3504
}
```

- SRT Listener - Egress

PowerShell

Copy

```
{
  "protocol": "srt-listener",
  "direction": "egress",
  "streamIp": "224.0.0.6",
  "streamPort": 4005,
  "listenerPort": 3505
}
```

- RIST Caller - Ingress

PowerShell

Copy

```
{
  "protocol": "rist-caller",
  "direction": "ingress",
  "streamIp": "227.0.0.3",
  "streamPort": 4012,
  "nodeIp": "10.1.2.2",
  "nodePort": 4555
}
```

• RIST Caller - Egress

PowerShell

Copy

```
{
  "protocol": "srt-caller",
  "direction": "egress",
  "streamIp": "224.0.0.4",
  "streamPort": 4003,
  "nodeIp": "10.0.0.3",
  "nodePort": 3503
}
```

• RIST Listener - Ingress

PowerShell

Copy

```
{
  "protocol": "rist-listener",
  "direction": "ingress",
  "streamIp": "226.2.2.2",
  "streamPort": 8000,
  "listenerPort": 3502
}
```

• RIST Listener - Egress

PowerShell

Copy

```
{
  "protocol": "rist-listener",
  "direction": "egress",
  "streamIp": "225.1.2.3",
  "streamPort": 5000,
  "listenerPort": 8000
}
```

Responses:

200 - Success

Example Value:

None

swxtch/protocols/v1/adaptors/update



POST

swxtch/protocols/v1/adaptors/update

- Update adaptor

URL:

PowerShell

Copy

http://<cloudSwxtch-Control-IP>/swxtch/protocols/v1/adaptors/update

Example URL:

http://10.2.128.10/swxtch/protocols/v1/adaptors/update

Request Body:

- [SRT available options](#)
- [RIST available options](#)

PowerShell

Copy

```
{
  "id": "12741484679446106659",
  "options": [
    {
      "name": "RISTO_CONGESTION_CONTROL_MODE",
      "value": "off"
    },
    {
      "name": "RISTO_TIMING_MODE",
      "value": "arrival"
    },
    {
      "name": "RISTO_MIN_RETRIES",
      "value": "1"
    },
    {
      "name": "RISTO_MAX_RETRIES",
      "value": "2"
    },
    {
      "name": "RISTO_RECOVERY_MAXBITRATE",
      "value": "50000"
    },
    {
      "name": "RISTO_RECOVERY_LEN_MAX",
      "value": "100"
    },
    {
      "name": "RISTO_RECOVERY_LEN_MIN",
      "value": "100"
    }
  ]
}
```

Responses:

200 - Responses

Example Value:

None

swxtch/protocols/v1/adaptors/remove

^

POST

swxtch/protocols/v1/adaptors/remove

Your content goes here

URL:

PowerShell

Copy

http://<cloudSwXtch-control-IP>/swxtch/protocols/v1/adaptors/remove

Example URL:

http://10.2.128.10/swxtch/protocols/v1/adaptors/remove

Request Body:

PowerShell

Copy

```
{
  "ids": [
    "12741484679446106659",
    "22731422274546106659"
  ]
}
```

Responses:

200 - Success

Example Value:

None

swxtch/protocols/v1/adaptors/show

^

POST

swxtch/protocols/v1/adaptors/show

Your content goes here

URL:

PowerShell

Copy

http://<cloudSwXtch-control-IP>/swxtch/protocols/v1/adaptors/show

Example URL:

http://10.2.128.10/swxtch/protocols/v1/adaptors/show

Request Body:

None

Responses:

200 - Success

Example Value:


```
{
  "10092296471997387357": {
    "id": "10092296471997387357",
    "protocol": "udp",
    "direction": "ingress",
    "streamIp": "224.0.0.1",
    "streamPort": 4000,
    "nodeIp": null,
    "nodePort": null,
    "listenerPort": 3500,
    "options": null
  },
  "14090786360518526144": {
    "id": "14090786360518526144",
    "protocol": "srt-caller",
    "direction": "ingress",
    "streamIp": "224.0.0.3",
    "streamPort": 4002,
    "nodeIp": "10.0.0.2",
    "nodePort": 3502,
    "listenerPort": null,
    "options": null
  },
  "14689058685391465390": {
    "id": "14689058685391465390",
    "protocol": "srt-caller",
    "direction": "egress",
    "streamIp": "224.0.0.4",
    "streamPort": 4003,
    "nodeIp": "10.0.0.3",
    "nodePort": 3503,
    "listenerPort": null,
    "options": null
  },
  "17428466851953349955": {
    "id": "17428466851953349955",
    "protocol": "srt-listener",
    "direction": "egress",
    "streamIp": "224.0.0.6",
    "streamPort": 4005,
    "nodeIp": null,
    "nodePort": null,
    "listenerPort": 3505,
    "options": null
  },
  "17465424783484296836": {
    "id": "17465424783484296836",
    "protocol": "rist-caller",
    "direction": "ingress",
    "streamIp": "227.0.0.3",
    "streamPort": 4012,
    "nodeIp": "10.1.2.2",
    "nodePort": 4555,
    "listenerPort": null,
    "options": null
  },
  "18141476381590292227": {
    "id": "18141476381590292227",
    "protocol": "srt-listener",
    "direction": "ingress",
    "streamIp": "224.0.0.5",
    "streamPort": 4004,
    "nodeIp": null,
    "nodePort": null,
    "listenerPort": 3504,
    "options": null
  },
  "86391891196034799": {
    "id": "86391891196034799",
    "protocol": "udp",
    "direction": "egress",
    "streamIp": "224.0.0.5",
    "streamPort": 4004,
    "nodeIp": "10.0.0.2",
    "nodePort": 4000,
    "listenerPort": null,
    "options": []
  }
}
```

POST

swtch/protocols/v1/adaptors/stats

Your content goes here

URL:

PowerShell

Copy

`http://<cloudSwXtch-control-IP>/swtch/protocols/v1/adaptors/stats`

Example URL:

`http://10.2.128.10/swtch/protocols/v1/adaptors/stats`

Request Body:

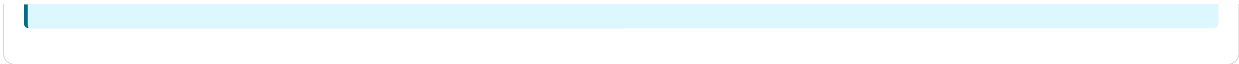
None

Responses:

200 - Success

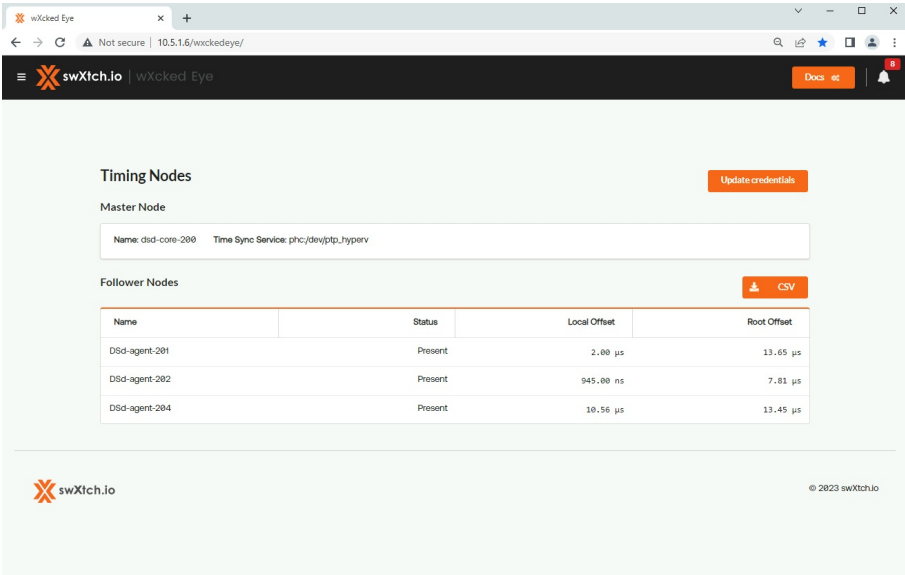
Example Value:

```
[
  {
    "id": "11151931551572244000",
    "counters": {
      "timestamp": 1692713703747,
      "txPackets": 0,
      "rxPackets": 123,
      "txPacketsLoss": 0,
      "rxPacketsLoss": 4,
      "txRetransmittedPackets": 0,
      "rxRetransmittedPackets": 24
    },
    "info": {
      "protocol": "srt-listener",
      "direction": "ingress",
      "streamIp": "234.1.1.1",
      "streamPort": 2500,
      "listenerPort": 3000
    }
  },
  {
    "id": "4155232358006180400",
    "counters": {
      "timestamp": 1692713703747,
      "txPackets": 57,
      "rxPackets": 0,
      "txPacketsLoss": 2,
      "rxPacketsLoss": 0,
      "txRetransmittedPackets": 14,
      "rxRetransmittedPackets": 0
    },
    "info": {
      "protocol": "srt-caller",
      "direction": "egress",
      "streamIp": "230.1.1.1",
      "streamPort": 2500,
      "nodeIp": "10.1.1.1",
      "nodePort": 4545
    }
  },
  {
    "id": "11151931551572244001",
    "counters": {
      "timestamp": 1692713703747,
      "txPackets": 0,
      "rxPackets": 255,
      "txPacketsLoss": 0,
      "rxPacketsLoss": 17,
      "txRetransmittedPackets": 0,
      "rxRetransmittedPackets": 3
    },
    "info": {
      "protocol": "rist-listener",
      "direction": "ingress",
      "streamIp": "229.1.1.1",
      "streamPort": 4368,
      "listenerPort": 5741
    }
  },
  {
    "id": "4155232358006180402",
    "counters": {
      "timestamp": 1692713703747,
      "txPackets": 74,
      "rxPackets": 0,
      "txPacketsLoss": 5,
      "rxPacketsLoss": 0,
      "txRetransmittedPackets": 11,
      "rxRetransmittedPackets": 0
    },
    "info": {
      "protocol": "rist-caller",
      "direction": "egress",
      "streamIp": "227.1.2.3",
      "streamPort": 3687,
      "nodeIp": "20.1.2.3",
      "nodePort": 7777
    }
  }
]
```



Precision Timing Protocol (PTP)

The Timing Nodes page displays information regarding clock sync configuration for the cloudSwXtch. The page in wXcked Eye will only populate with information if the user has the PTP feature enabled.



PTP Master

swxtch/ptp/v1/master

POST

swxtch/ptp/v1/master

- Assign the cloudSwXtch as the Master Node to receive time from the True Clock Source.

URL:

Bash

Copy

curl http://<cloudSwXtch-control-IP>/swxtch/ptp/v1/master

Example URL:

http://10.2.128.10/swxtch/ptp/v1/master

Example Request Body:

None

Example Response:

Bash

Copy

{
 "name": "core-100",
 "displayname": "phc:/dev/ptp_hyperv",
 "type": "primary"
}

PTP Followers

swxtch/ptp/v1/followers

POST

swxtch/ptp/v1/followers

- Assign xNICs as PTP Follower Nodes.

URL:

Bash

Copy

```
curl http://<cloudSwXtch-control-IP>/swxtch/ptp/v1/followers
```

Example URL:

http://10.2.128.10/swxtch/ptp/v1/followers

Example Request Body:

None

Example Response —>

Bash

Copy

```
[
  {
    "name": "agent-101",
    "xnicPresent": true,
    "timebeatPresent": true,
    "localOffset": 2634.765912,
    "rootOffset": 22687.686621
  },
  {
    "name": "agent-104",
    "xnicPresent": true,
    "timebeatPresent": true,
    "localOffset": 4878.807692,
    "rootOffset": 17720.857275
  },
  {
    "name": "agent-105",
    "xnicPresent": true,
    "timebeatPresent": true,
    "localOffset": 10955.28,
    "rootOffset": 29702.527246
  },
  {
    "name": "agent-201",
    "xnicPresent": true,
    "timebeatPresent": null,
    "localOffset": null,
    "rootOffset": null
  },
  {
    "name": "agent-204",
    "xnicPresent": true,
    "timebeatPresent": null,
    "localOffset": null,
    "rootOffset": null
  },
  {
    "name": "aks-nodepool1-40504797-vmss00000C",
    "xnicPresent": true,
    "timebeatPresent": null,
    "localOffset": null,
    "rootOffset": null
  },
  {
    "name": "aks-nodepool1-40504797-vmss00000D",
    "xnicPresent": true,
    "timebeatPresent": null,
    "localOffset": null,
    "rootOffset": null
  }
]
```

Update PTP Credentials

swxtch/ptp/v1/updateTimebeatCredentials

swxtch/ptp/v1/updateTimebeatCredentials

- Update PTP credentials

URL:

Bash

Copy

```
curl http://<cloudSwXtch-control-IP>/swxtch/ptp/v1/updateTimebeatCredentials
```

Example URL:

http://10.2.128.10/swxtch/ptp/v1/updateTimebeatCredentials

Example Request Body:

Bash

Copy

```
{
  "username": "Admin1",
  "password": "TimePassword"
}
```

Example Response:

Bash

Copy

```
"Credentials updated successfully"
```

Release Notes

WHAT TO EXPECT

In this section, users will find release notes for current and past versions of our product. If you have any questions regarding your particular version, please contact us at info@swtch.io.

To see past versions of our documentation site, please check out our [Resources](#) page.

3.1.0 Release Notes - October 7, 2024

New Features

- Two new arguments have been added to the xNIC and Bridge JSON file for improved flexibility and control over high availability configurations.
 - HA Enable: The enable flag allows users to tell the xNIC/cloudSwXtch Bridge whether or not it should join a cloudSwXtch cluster for data and control traffic.
 - Deduplication: Users can now specify in the xNIC JSON file whether or not to deduplicate data plane traffic in favor of their own application at the endpoint.

New Improvements

- cloudSwXtch Bridge Type 3**
 - The number of necessary 3rd party packages for cloudSwXtch Bridge 3 installation have been reduced. Please see [here](#) for updated list. (#3463)
 - A new --ag or --airgap argument has been added when running the installation script for air-gapped environments. Using ag requires users to download 3rd party packages onto their VM before bridge installation since there will be no internet access.
 - A new Firewall exception has been added to the cloudSwXtch Bridge Type 2 and 3 installer for IGMP. Review all firewall exceptions [here](#).
- xNIC**
 - Precision Time Protocol installation is now optional and disabled by default.
- wXcked Eye Topology Graph**
 - Users can add cloud to ground subscriptions for the cloudSwXtch Bridge in the Component Information panel.
 - Both the View By/Filter By bar and the Filters panel in the wXcked Eye Topology Graph are now collapsible. (#5913)
 - The Glossary, Settings, Center Graph, and Resize Node controls have been relocated to the right hand side of the graph. (#5913)
 - The Component Information panel is now moveable across the Topology Graph.
 - There is a new Interfaces layer which displays a streamSocket's interface. (#5912)
 - The velocity of the stream links will animate differently based on bps. (#6041)
 - The graph is now reactive to zoom with StreamSocket information only displayed when fully zoomed in.
- swXtch-top**
 - Filtering is now implemented in every view in swXtch-top. (#5941)

Known Issues

- The filtering option in swXtch-top will not work if a user clicks into the Filter search bar.
 - Workaround:** While using swXtch-top, do not click into the Filter search bar. Simply type the value you're searching for, followed by an asterisk (*), to begin filtering.
 - Reminder:** Filtering is case-sensitive.
- The argument to enable PTP for xNIC on Windows is incorrectly defined when using the --help command as `--ptp true`.
 - Workaround:** Users should use `-ptp_install true` when enabling PTP for the xNIC on Windows.

3.0.0 Release Notes - September 6, 2024

New Features:

- A new way of bridging with cloudSwXtch Bridge Type 3!**
 - With added support on RHEL 9 and for non-Mellanox NICs, cloudSwXtch Bridge Type 3 is a DPDK-based variant, touting higher throughput and the ability to run **swXtch Lossless UDP for Ground-to-Cloud**. For more information on installation, see [here](#).
 - This includes major enhancements, such as configurable parameters that can be manually tuned when establishing a ground-to-cloud link.
 - High availability and Protocol Fanout is configurable in wXcked Eye for cloudSwXtch Bridge.
- We've added support for Cinnafilm's Tachyon LIVE for cloudSwXtch**, adding a multitude of video transformations in wXcked Eye. For more information about installation, contact support@swtch.io.
- We've added multiple NIC support for High Availability configuration on the xNIC**. This allows users to completely separate 2022-7 or high availability traffic for each path. In addition, users can enable multiple agents to communicate with multiple cloudSwXtches without requiring peering. For more information, see [here](#).
- A new and improved way of seeing things with wXcked Eye!**
 - The **wXcked Eye Topology graph** has been overhauled, introducing a slew of new features to allow for real-time monitoring with new streams and channels views.
 - Each graph component opens an information sidebar when selected. This information includes general, environment, hardware, total rx/tx, related streams, and related adaptors.
 - A new Layers toggle feature allows for users to view different combinations of data as required: name components, clouds, streams, rates, regions, sockets, instances, and precision time protocol.
 - Protocol Fanout** is now configurable directly in the Topology Graph. Users can add and edit adaptors via the components information panel.
 - High Availability and swXtch Lossless UDP stats are now visible directly on the Topology Graph.

- A new [Topology Table](#) view gives users a detailed look into individual streams and channels. Additionally, users can view streams in a tree view with components pulled directly from the topology graph.
- A new [wXcked Eye Components](#) page allows users to merge interfaces into a single VM, removing duplicate virtual machines in the topology graph. This page also allows users to manually enter missing environment information for non-xNIC VMs.
- A new wXcked Eye [Aliases](#) tab allows users to assign user-friendly names and colors for Streams, Sockets, and Topology Graph Components.
- An updated [Support](#) page allows users to define log ranges by number of days and/or hours.
- The Settings page has a new [Channels](#) tab, allowing users to add channels based on streams linked by converter.
- Users can specify certain colors for HA paths, streams, and channels to carry throughout the wXcked Eye UI.
- Users can configure their cloudSwXtch for Tachyon LIVE via the wXcked Eye Settings page. Please contact support@swXtch.io for more information.

New Improvements:

- **Updates to swXtch-top:**
 - swXtch-top has been refactored with all new layout and updated navigation menu.
 - A new [Components](#) view to mirror what users see in wXcked Eye for their individual topology graph nodes.
 - Users can also see swXtch Lossless UDP stats from Ground-to-Cloud in a new [Lossless](#) view.
 - A new [Subscriptions](#) view details different source list configurations for single source multicast and multiple multicast groups.
 - A new [Configurations](#) page displays xNIC JSON configuration files.
 - Users can now navigate between different views by clicking directly into swXtch-top.
 - A new [Setup](#) function allows users to modify what columns are visible in a view and the width of those columns.
 - To see all the updates to the interface, see [swXtch-top](#) under Testing cloudSwXtch.
- **Updates to API:**
 - Our [Monitoring](#) and [Configuration API](#) has been updated with new responses. A [topology API](#) call has been added as well.
 - An issue with /stats/xnics showing multiple entries in RxMulticastGroups for the same Group IP has been resolved.
 - The Dashboard API (/wXckedEye/v1/dashboard) has been deprecated. Please use the [Topology API](#) call.
 - The [Protocol Fanout Adaptor API](#) calls have been streamlined with different body/payloads for each protocol. Old calls for specific protocols have been deprecated.

Known Issues:

- There are IP conflicts when multiple xNICs on different VPCs have a ctrl IP with similar host names. (4857)
 - **Workaround:** Avoid using the same IP and hostname between VPCs.
- Users may experience reduced performance on instances with large configurations. (5821)
 - **Workaround:** The issue stems from suboptimal thread configuration. For more information on how to manually modify core CPU usage, see [here](#).

2.2.0 Release Notes - March 18, 2024

New Features:

- **Say hello to updates to Bridge Type 2!**
 - [Protocol Fanout](#) is now available on the cloudSwXtch Bridge in Type 2. Previously, it was only available on the cloudSwXtch in the cloud. For more information on configuring the cloudSwXtch Bridge for Protocol Fanout, see the following [article](#).
 - **2022-7 for High Availability** is now configurable on a cloudSwXtch Bridge in both Type 2. For more information, see the [High Availability](#) article under Configuring cloudSwXtch.
- **Keep things secure with our verifiable xNIC installers!**
 - We now provide an optional way to verify the legitimacy of our xNIC installer script and installation packages against signed hash files with a swXtch.io public key. For more information, see [Verifying Installer Files](#) in the [Install xNIC on Linux](#) articles.
 - Currently, only available on Linux.

New Improvements:

- **Improved performance for AWS:**
 - c6in or m6in EC2 instance families are now recommended for cloudSwXtch instances running version 2.2.0 or greater. This ensures that users get the best performance out of their cloudSwXtch network on AWS. Any current m5zn cloudSwXtch based instances should migrate to one of the two family types.
- **Updates to swXtch-top:**
 - A new Bridge view has been added to swXtch-top, displaying data flow stats specifically for Bridge Type 2. For more information, see [swXtch-top](#) article.

Bug Fixes:

- Fixed issue where a double join was occurring in Cloud-to-Ground.
- Fix stream stats in swXtch-top showing intermittent negative values when the replicator service is restarted.

Known Issues:

- There are IP conflicts when multiple xNICs on different VPCs have a ctrl IP with similar host names. (4857)
 - **Workaround:** Avoid using the same IP and hostname between VPCs.
- **[Resolved 3.0.0]** /stats/xnics can show multiple entries in RxMulticastGroups for the same Group Ip. (4893)
 - **Workaround:** None. This will be addressed in an upcoming release.

2.1.3 Release Notes - March 5, 2024

New Improvements:

- **Updates to swXtch-top:**
 - A new Search bar in the interface will allow users to filter items in the xNICs, Streams and Notification views. (4743)
 - Navigating between swXtch-top's views will require hitting both the CTRL button on a user's keyboard and the associated letter. For example, to get to the xNICs view, a user will need to key in CTRL + X to switch to that page.
 - The Hardware view is now called "Environment."
 - A new Notifications view has been added. (4709)
 - Both the HA Summary and HA view will only list in the navigation menu when HA is configured.
 - For more information, check out the [swxtch-top](#) page under Testing cloudSwXtch.
- **Updates to wXcked Eye:**
 - The wXcked Eye Navigation menu will now properly scroll if using the UI on a lower resolution screen. (5029) [Known Issue 2.1.2]
- **Updates to swx support:**
 - Users can now export xNIC logs using the swx support command. For more information, see [How to View cloudSwXtch Logs for Troubleshooting](#). (4574)

Bug Fixes:

- We corrected a bug that would happen when PTP service start-up fails and the master node in swxtch-top shows as empty. (5110) [Known Issue 2.1.2]

Known Issues:

- There are IP conflicts when multiple xNICs on different VPCs have a ctrl IP with similar host names. (4857)
 - **Workaround:** Avoid using the same IP and hostname between VPCs.
- **[Resolved 3.0.0]** /stats/xnics can show multiple entries in RxMulticastGroups for the same Group Ip. (4893)
 - **Workaround:** None. This will be addressed in an upcoming release.

2.1.2 Release Notes - February 15, 2024

New Improvements:

- **Updates to xNIC:**
 - Users can now download and install the xNIC without having a valid license authorizing their cloudSwXtch. However, to begin packet flow from the cloudSwXtch to the xNIC(s), a license is required. (4974)
- **Updates to swx support:**
 - New parameters have been added to the swx support command. For a complete list, please see [How to View cloudSwXtch Logs for Troubleshooting](#). (4574)
- **Updates to wXcked Eye:**
 - The [Protocol Fanout stats page](#) will now display the Alias name by the stream IP address. The page also allows users to filter Adaptors by multiple protocols. (4572)
 - The [Protocol Fanout tab in Settings](#) will also now display the Alias name next to the Stream and Node IP addresses in the Adaptors table. (4572)
 - The [Support page](#) will now allow users to export xNIC logs.(4574)
- **Updates to Licensing Server:**
 - The [Licensing Server](#) will no longer require a manual start after initial install. (4919) [Known issue in 2.1.1]

Known Issues:

- **[Resolved 2.1.3]** PTP service start-up fails and switch-top shows that the master node is empty. (5110)
 - **Workaround:** Use the following four commands in succession to remedy this issue:

PowerShell

Copy

```
sudo systemctl start elasticsearch.service
sudo systemctl enable elasticsearch.service
sudo systemctl start grafana-server.service
sudo systemctl enable grafana-server.service
```

- There are IP conflicts when multiple xNICs on different VPCs have a ctrl IP with similar host names. (4857)
 - **Workaround:** Avoid using the same IP and hostname between VPCs.
- **[Resolved 2.1.3]** wXcked Eye navigation menu does not scroll.
 - **Workaround:** Ensure you're using wXcked Eye in a high-res screen.
- **[Resolved 3.0.0]** /stats/xnics can show multiple entries in RxMulticastGroups for the same Group Ip. (4893)
 - **Workaround:** None. This will be addressed in an upcoming release.

2.1.1 Release Notes - February 5, 2024

New Improvements:

- **Updates to wXcked Eye:**
 - The Alias tab is now split into two separate pages, one for streams and exclusions and another for nodes.
- **Updates to Prometheus API:**
 - `swx_maxClientCount` and `swx_numClientsConnected` are two new metrics added to Prometheus while both `swx_xnic_maxActiveConnections` and `swx_xnic_activeConnectionCount` have been deprecated.
 - When calling for metrics regarding nodes, only current active ones will display. Inactive or disconnected nodes will no longer list.
- **Updates to xNIC Logs:**
 - File paths in Windows and Linux for xNIC logs have been changed. Please see the article, [How to Find xNIC Logs](#), for more information.
- **Updates to xnic.json file:**
 - "childs" has been updated to "children" in the xnic.json file for configuring HA paths. For more information, please see [High Availability](#) under Configuring cloudSwXtch.

Known Issues:

- **[Resolved 2.1.2]** The Licensing Server does not automatically start after install. (4919)
 - **Workaround:** After the initial installation, please use the following command to start the licensing server.

PowerShell	Copy
sudo systemctl start swxtch-license-svr	

Subsequent start-ups will happen automatically when the licensing server VM is turned on.

- There are IP conflicts when multiple xNICs on different VPCs have a ctrl IP with similar host names. (4857)
 - **Workaround:** Avoid using the same IP and hostname between VPCs.
- **[Resolved 2.1.3]** wXcked Eye navigation menu does not scroll.
 - **Workaround:** Ensure you're using wXcked Eye in a high-res screen.
- **[Resolved 3.0.0]** /stats/xnics can show multiple entries in RxMulticastGroups for the same Group Ip. (4893)
 - **Workaround:** None. This will be addressed in an upcoming release.

2.1.0 Release Notes - January 2024

New Features:

- **Users can now configure their High Availability for Multiple Multicast Groups (mMC-7)**. For more information on how to configure the xNIC for mMC-7, please see [High Availability](#) under Configuring cloudSwXtch. (4239)
- The cloudSwXtch can now perform **Source Specific Multicast (SSM)** as a method of delivering multicast packets where the receiver includes or excludes them through a specific source address.
- A new **swx support** command is now available for users to export a tar.gz file detailing statistical data stored within a cloudSwXtch during a set period of time. This is especially helpful when troubleshooting with swXtch.io Support. (4107, 4488)
- A self-hosted Licensing Server is now available for installation on your cloudSwXtch network. This allows users to self manage cloudSwXtch licenses instead of having to request one for each instance from swXtch.io. Read more about it [here](#).
- **Kubernetes support expanded** to include GCP (GKE) and AWS (EKS) in addition to existing Azure (AKS) support.

New Improvements:

- **Updates to wXcked Eye:**
 - The new **wXcked Eye Cluster view** provides users with an expansive look of all cloudSwXtches connected in a high availability or mesh configuration. Serving as the new main page, users can easily navigate between cloudSwXtches for additional metrics. A universal Cluster quick view side panel has also been added, displaying up to 5 cloudSwXtches. (4568)
 - A new **wXcked Eye Support Page** has been added, allowing users to export a JSON file with a full cloudSwXtch report over a certain period of time. This is especially helpful when troubleshooting with swXtch.io Support. (4269)
 - A new batch select and delete feature has been added to the **Aliases** tab under Settings. (4320)
 - Users can now exclude certain streams via IP addresses from wXcked Eye and swXtch-top monitoring. (3732)
 - The Hardware panel in the **General** tab under Settings now lists metadata and OS for the control and data subnets. (4351, 4352)
 - A new **Protocol Fanout Stats** page displays SRT and RST data flow to and from the cloudSwXtch. Previously, only multicast data flow was displayed across the UI. (4322)
 - A new cloudSwXtch information banner now displays at the top of each page in wXcked Eye. (4571)
 - The Network Graph is now called **Topology**.
 - **Timing Nodes** page has been relocated to under Monitoring.
 - wXcked Eye is fully scalable with responsive views and page resizing. (4328)
 - wXcked Eye and swXtch-top are both unified in their performance metrics and cloudSwXtch information.
 - Users can now return to previously visited pages in the UI using a navigation history tool at the top of each page.
- **Updates to xNIC:**
 - **The xNIC installer script will automatically install xNIC Type 2 as the default and recommended xNIC.** From now on, whenever the documentation refers to the xNIC, it means Type 2. (4249)

- The xNIC installer script will automatically open recommended ports in a user's firewall.
- [HA Configuration for an xNIC](#) is now automatic as long as only one multicast group is required. (4305, 4460)
- The xNIC configuration file is now a .json.
- Precision Time Protocol (PTP) install is now optional for both the cloudSwXtch and the xNIC. (4368)
- The xNIC has been enhanced to work with a single-NIC configuration if the cloudSwXtch's ctrl and data NIC share a single subnet.
- Support for the following Operating Systems has been added: AlmaLinux 8.8, Amazon Linux 2023, RHEL 9.2, Rocky Linux 8 + 9, and Ubuntu 22.04. For a comprehensive list, see [xNIC System Requirements](#). (4414, 4314, 3371, 4261, 4154)
- **Updates to xNIC on K8s:**
 - The procedure to install the xNIC on a Kubernetes cluster has been updated with a new installer. See [Install xNIC on K8s Cluster](#) for more information. (3131, 4358)
- **Updates to Prometheus API:**
 - The [Prometheus API](#) metrics are now `swx_core` for the cloudSwXtch and `swx_xnic` for the xNICs. (4437)
 - The data for the xNICs are now grouped together, allowing users to filter by name.
- **Updates to swXtch-top:**
 - A new [Hardware](#) view has been added to swxtch-top, displaying metadata and OS for control and data subnets. (4351)
 - xNIC will now display as either Type 1 (t1) or Type 2 (t2). Type 2 is the default xNIC. (4445)
- **Updates to swXtch-perf:**
 - Single Source Multicast (SSM) include and exclude consumer commands have been added. In addition, a command to show the bps of all packet headers has also been included. For more information, see [swxtch-perf](#) article. (4386,
- **Bridge Installation**
 - Bridge installation command has been updated.

Bug Fixes:

- Fixed an issue with Protocol Fanout Egress Adaptors not working with xNIC Type 2. (4520)

2.0.34 Release Notes - October 2023

New Features:

- **Say hello to cloudSwXtch on Google Cloud Platform (GCP) and Oracle Cloud Infrastructure (OCI)!**
 - cloudSwXtch is now available on GCP as a Cloud agnostic VM install. For more information, see [Cloud agnostic cloudSwXtch VM Install](#). These instructions can be applied to any cloud as an alternative method of installation.
 - cloudSwXtch is now available as a BYOL in the OCI Marketplace. For more information, see [cloudSwXtch on OCI](#).
 - Please note: Both installations will require a license from swXtch.io.
- Users can now **simplify** their cloudSwXtch and xNIC networking structure with the **new single-subnet configuration**. For more information, see [System Requirements for cloudSwXtch and xNIC](#).

New Improvements:

- **Updates to wXcked Eye:**
 - Improved usability for Protocol Fanout configuration by creating a separate tab for Stream and Node Aliases. This new naming mechanism will work for both Protocol Fanout and for the Network Graph. (3733, 3840)
 - Rename High Availability paths directly in the UI (3849)
 - wXcked Eye and swXtch-top will now warn users if their cloudSwXtch license has less than 7 days left or has expired. (4101, 4199, 4205)
 - Added new warning messages in HA and Mesh settings page to remind users that these two configurations are mutually exclusive. (3794)
 - Fixed zoom scrolling capabilities in the wXcked Eye Network Graph to improve ease of use. (3657)
- **Expanded SMPTE 2022-7 Support:** For more information, see [High Availability](#) under Configuring cloudSwXtch. (3784)
- Added support for Ubuntu 22.04 with xNIC2 as the default. (4109)
- Added support for Amazon Linux 2 and Linux 2023. (3821)

2.0.10 Release Notes - July 2023

New Features:

- **Introducing wXcked Eye**, a brand new user interface to monitor and configure your cloudSwXtch environment. For more information on its functionality, see [Using wXcked Eye for cloudSwXtch](#). (2714)
- Added support for xNIC installation on AKS Cilium Native. For more information, see [Install xNIC on AKS Cilium Native](#). (3841)
- Added support for Prometheus and Grafana Monitoring for cloudSwXtch. For more information, see [Prometheus Monitoring](#). (3598)
- Added support for Protocol Fanout and Conversion including SRT to the wXcked Eye UI. For more information, see [Protocol Conversion and Fanout with wXcked Eye](#). (3181)
- Implemented Simple Packet Fragmentation for cloudSwXtch Bridge Type 2. (3276)

New Improvements:

- swxtch-top view 5 and 6 will now be dynamically hidden when a high availability configuration is destroyed. (3797)
- swxtch-top will now display a user's entitlements. (3738)

- Added support for Linux 2 on AWS for swXtch-xnic installation. (3570)
- Added in xNIC installer capability to add control and data NIC as well as unattended option. For more information, see [Install xNIC on Linux](#) or [Install xNIC on Windows](#). (3696)
- **Updates to API** - For more information on the changes below, see [Monitoring API](#) and [Configuration API](#). (2714)
 - The following additions break out the `http://<cloudSwXtch_IP>/api/wxckedeye/v1/dashboard` API into separate calls. The Dashboard call is still available, but will be deprecated in 2.1.0.
 - Added a new call to get information about the cloudSwXtch
 - Added a new call to get information about the Cloud
 - Added a new call to get statistics on the cloudSwXtch
 - Added a new call to get statistics on the xNIC's
 - Added a new call to get xNIC totals data
 - For mesh data the `listMembersAddress` was deprecated and replaced with `show`.
 - The word "switch" has been replaced with "swXtch" in all calls.
 - Added new calls for Streams to be used with Protocol Fanout: Add Stream, Update Streams, Show Streams and Remove Stream.
 - Added new calls for Nodes to be used with Protocol Fanout: Add Node, Update Node, Show Node and Remove Node.
 - Protocol Fanout Unicast changes:
 - Enable call has been changed from "http://<cloudSwXtch>/swxtch/unicast-adaptor/enable" to "http://<cloudSwXtch>/swxtch/protocols/udp/v1/ingress/enable".
 - Enable call has been changed from "http://<cloudSwXtch>/swxtch/unicast-adaptor/join" to "http://<cloudSwXtch>/swxtch/protocols/udp/v1/ingress/join", additionally, the body format has changed to add "Target Port" and "Group Port".
 - Added Unicast Adaptor "Ingress Show" call.
 - Added Unicast Adaptor "Egress Show" call.
 - Leave call has been changed from "http://<cloudSwXtch>/swxtch/unicast-adaptor/leave" to "http://<cloudSwXtch>/swxtch/protocols/udp/v1/ingress/leave", additionally, the body format has changed to add "Target Port" and "Group Port".
 - Disable call has been changed from "http://<cloudSwXtch>/swxtch/unicast-adaptor/disable" to "http://<cloudSwXtch>/swxtch/protocols/udp/v1/ingress/disable".
 - Protocol Fanout SRT for both "Caller" and "Listener" modes have been added.
 - PTP API has been added to get timing data and to update timing credentials

Bug Fixes:

- Fixed an install error when Timebeat returned a fail status. (3472)
- Fixed an issue where Mesh configuration was sometimes not persisted. (3640)
- Fixed an issue for xNIC2 in Windows, optimizing it for lower CPU utilization when idle. (2805)

1.9.85 Release Notes

New Features

- **Azure Air-Gapped** (No Internet) Image and Installation instructions are now available. (3694)

New Improvements:

- **Updates to PTP interface config:**
 - Modified the installation files for cloudSwXtch and xNIC so that they support Linux Ubuntu 18.x.20.x for both xNIC Type 1 and xNIC Type 2. (3522)
- We updated `install_swxtch.sh` to force a reboot if the UIO driver/module was installed. (3322)
- Added capability to add Gateway IP to Bridge Config.

Bug Fixes:

- **swXtch-top:**
 - Outdated or incorrect text has been fixed for the Mesh and Config views. (3560, 3562)
 - We removed duplicated flow values. (3436)
 - We fixed spacing in between rows in HA Paths view. (3546)
- We fixed BigPacket detection and reset BigPacket counter for Linux xNIC2. (3401)

1.9.73 Release Notes

New Features

- We added support for **Precision Time Protocol (PTP)**. (3068)
- **Updates to the wXcked Eye UI:**
 - The wXcked Eye UI main page is now the **Network Graph**, presenting users with a high-level view of their cloudSwXtch and its endpoints. For more information on this functionality, please refer to the "wXcked Eye Network Graph" article. (2771)

- Under Configuration, a new **Settings** page has been added, allowing users to configure their cloudSwXtches for **Mesh**, **High Availability** and **Protocol Fanout**. A **General** tab displays information regarding their cloudSwXtch and the entitlements allowed by their license. (3087)
- Under Configuration, a new **Notifications** page displays updates regarding a user's cloudSwXtch. (3344)
- Under Configuration, a new **Timing Nodes** page has been added, displaying information on PTP. (3113)
- **Updates to swXtch-top:**
 - We added three new views: **HA Streams**, **Config** and **PTP**.
 - We condensed HA and multi-cloudSwXtch information into one column. (3275)

Improvements:

- **xNIC2 for Windows** will now allow tunneling and broadcast traffic to pass.
- Updates to **swtch-perf** on **Windows** and **Linux**: (3117)
 - **Consumer loopback mode**
 - **Producer loopback mode**
 - **Increase logging information (dbg)**
- Updates to **swtch-perf** on **Windows**: (3117)
 - **One-way latency**
 - **RTT latency**
 - **One way/RTT latency buckets**
- Added Windows 11 support.
- **Linux xNIC** now uses the payload size of the received packets instead of the Data NIC MTU size when producing a loopback. This will prevent packets from dropping. (3180)
- Updated the maximum payload length for **swtch-perf** to the maximum UDP payload length size (65507) - SWX header size (32 bytes) = 65475 bytes. This will allow swtch-perf to be used in more versatile scenarios with large packets and not be constrained by payload length. (3274)

Bug Fixes:

- We corrected issues with **installing xNIC on Centos7 for AWS**. (3349)
- We fixed an issue where **Windows xNIC2** wasn't able to send packets with a payload of 1470 and **bigpackets** counter didn't modify its value in the situation. This detection of big packets was also fixed on Windows and Linux xNIC1. (3401)

Resources

PDFs

[cloudSwXtch User Guide v1.9.85](#)

[cloudSwXtch User Guide v2.0.34](#)

[cloudSwXtch User Guide v2.1.0](#)

[cloudSwXtch User Guide v2.1.1](#)

[cloudSwXtch User Guide v2.1.2](#)

[cloudSwXtch User Guide v2.3.2](#)

[cloudSwXtch User Guide v3.0.0](#)

Quotas

cloudSwXtch

All bandwidth and packet per second values are aggregate values (i.e ingress + egress) unless otherwise noted.

Name	Default Value	Configurable
Multicast Packet Size	Up to 3750	Yes
Endpoint Connections	Unlimited	NA
Max Throughput per cloudSwXtch	Up to 100 Gb/s	No
Max Bandwidth per flow	Up to 15 Gb/s	No
Max Packets per second per cloudSwXtch	Up to 10M	No
Max cloudSwXtch instances per mesh	32	No
Max Bridge instances per cloudSwXtch	4	No
Max fanout outputs per cloudSwXtch	1000	No

cloudSwXtch Sizing

cloudSwXtch Multicast (Marketplace)

# Endpoints	Bandwidth	Core	Memory	Hard Drive
10 (max)	100 Mbps (max)	8	16GB DDR	64GB SSD

cloudSwXtch BYOL (Marketplace)

# Endpoints	Bandwidth	Core	Memory	Hard Drive
Up to 100	2 Gb/s (max)	16+	16GB DDR	64GB SSD
Up to 200	More than 2Gb/s	64+	16GB DDR	64GB SSD

xNIC

Name	Default Value	Configurable
Multicast Packet Size	Up to 3750	Yes
Multicast Groups	Unlimited	NA
Max cloudSwXtch Connections	4	No
Max Bandwidth	Up to 15 Gb/s	Yes

