KVM Virtual Sensor (vSensor) Deployment Guide

Version: August 27, 2025

Table of Contents

Introduction

This guide is intended to help customers or partners deploy vSensors in KVM environments and pair them to your Vectra Brain. It will cover basic background information, connectivity requirements (firewall rules that may be needed in your environment), deployment of the vSensor in KVM, and pairing.

vSensors behave much in the same way that physical Sensors do. One advantage is that there is no cost to deploy a vSensor other than your own costs to provide and maintain the infrastructure they run in. vSensors also allow you to capture and analyze traffic that only exists in the virtual environment. You can even use vSensors in place of physical Sensors to capture physical network traffic.

KVM vSensors can be used in both Respond UX and Quadrant UX deployments. For more detail on Respond UX vs Quadrant UX please see Vectra Analyst User Experiences (Respond vs Quadrant). One of the below guides should be the starting point for your overall Vectra deployment:

- Vectra Respond UX Deployment Guide
- Vectra Quadrant UX Deployment Guide

Either of the above guides cover basic firewall rules needed for the overall deployment and initial platform settings. Virtual Sensor (VMware, Hyper-V, KVM, AWS, Amazon, and GCP) configuration and pairing and covered in their respective guides. Physical appliance pairing is covered in the Vectra Physical Appliance Pairing Guide. Please see the Vectra Product Documentation Index on the Vectra support site for additional documentation including deployment guides for CDR for M365 / IDR for Azure AD and CDR for AWS.

About KVM vSensor Images

The Brain makes an image available in QCOW2 format for download and subsequent use for deploying KVM vSensors. Vectra appliances typically operate with updates enabled. Regular updates ensure that the appliances are running the very latest version. Deployed Sensors and vSensors also update regularly from the Brain. Once a vSensor has been deployed, it will update itself as needed, staying current with its Brain.

- Please note that as your Vectra platform (Brain) is updated, the image for KVM is also updated.
    - If you deploy additional KVM vSensors in the future, always download a fresh copy of the image from an up-to-date Brain to ensure you are working with the latest code.

KVM vSensor Requirements and Throughput

Vectra supports KVM for customers deploying virtual Sensors (vSensors) to capture virtual traffic or physical traffic. Brain or mixed mode deployment is NOT supported in KVM.

KVM vSensor Configurations

VM Type	Capture Ports	Cores	Memory	Storage	Performance *
Standard PC (Q35 + ICH9, 2009)	1	2	8 GB	100 GB	500 / 250 Mbps
Standard PC (Q35 + ICH9, 2009)	1	4	8 GB	150 GB	1 / .5 Gbps
Standard PC (Q35 + ICH9, 2009)	1	8	16GB	150 GB	2 / 1 Gbps
Standard PC (Q35 + ICH9, 2009)	1	16	64 GB	500 GB	5 / 2.5 Gbps

- * 1st number represents NDR/Detect only performance, 2nd number represents performance with Match and/or Suspect Protocol Activity Detections enabled.
- KVM vSensors also require a management port that is separate from the capture port.
- Vectra recommends that Sensors are configured to use storage local to the hypervisor and are not stored on a SAN. Vectra vSensors require extremely high throughput from their disk storage and this throughput cannot normally be sustained by SAN systems without impact to other SAN users.

Connectivity Requirements

The Vectra Respond UX Deployment Guide or Vectra Quadrant UX Deployment Guide detail basic connectivity requirements for initial platform deployment. It also gives guidance on firewall/proxy SSL inspection, Internet access to and from the Brain, and guidance for air-gapped environments. For full detail on all possible firewall rules that might be required in your environment, please see Firewall Requirements for Vectra Appliances. KVM vSensor specific requirements are listed below:

Connectivity Requirements for KVM vSensors

Source	Destination	Protocol/Port	Description
Admin Hosts	vSensors	TCP/22 (SSH)	CLI access to vSensor
Brain	vSensors	TCP/22 (SSH)	Remote management and troubleshooting
vSensors	Brain	TCP/22 (SSH) TCP/443 (HTTPS)	Pairing, metadata transfer, and ongoing communication

Please note:

- vSensors do not communicate with the Vectra Cloud
- All communication sessions with vSensors are initiated from the vSensor to the Brain
- Updates for vSensors are downloaded to the Vectra Brain and the vSensor retrieves them from the Brain
- Command Line access can also be obtained via the console in your hypervisor

Preparing to Deploy KVM vSensors

Some information will need to be gathered or known prior to beginning deployment of a KVM vSensor.

- IP address and subnet mask for the Management interface of the vSensor.
- IP address or hostname of the Vectra Brain that you will be pairing with.
    - This will be built into the downloaded VM image and will point to the Brain that served the download.
    - Whether this points to the IP of the Brain or the hostname depends on what you have set in your UI under *Data Sources > Network > Sensors > Sensor Configuration > Sensor Pairing and Registration* for **"Pair using the Detect Brain"**.
        - Editing this area gives a radio button to choose "**DNS Name**" or **"Management IP Address"**.
- DNS server addresses.
- To configure your vSensor, you will need access to the vSensor Command Line Interface (CLI) either via the

console in your hypervisor or via SSH.
  - DHCP is enabled by default upon vSensor initial boot.
  - You must know the IP that was assigned via DHCP to SSH to the CLI, otherwise you will need to use the hypervisor console.
- For production monitoring, ensure that the vSensor VM is kept running 24x7, and ensure that the hypervisor does not overcommit resources or otherwise misrepresent the resources it is providing to the vSensor.

KVM Requirements, Networking, and Interaction

Requirements

- **Vectra currently only supports KVM on Red Hat or Ubuntu Linux installations.**
- Ensure that libvirt (including libvirt-client) and virt-install are installed locally on the KVM host server.
- You must either be a root user or ensure that the user that will later launch the deployment script is in the libvrt group on the KVM host server.

Networking

- Ensure that the desired KVM networks (that will be used for management and capture) are present, persistent, active, and autostarted using "**`virsh net-list –all`**".
  - The default driver supported for the network interface is virtio-pci.
- Vectra recommends that customers create a new virtual network on the KVM host server from which they want to capture packets and set the interface to promiscuous mode.
  - This is the only configuration that Vectra officially tests and supports.
  - The Vectra vSensor(s) should be the only virtual machine(s) on this network, unless there are other similar virtual machines that you want to provide the same set of traffic into.
- KVM traffic mirroring may be implemented using either Open vSwitch or tc.
  - The steps to create a new mirror from an existing physical network interface to a new virtual one are as follows:

Open vSwitch:

```
# Define the interfaces and the mirror
BRIDGE="br0"
SRC="eth0"
DST="vnet0"
MIRROR="span"
# Create a bridge and set all ports up
ovs-vsctl add-br ${BRIDGE}
ip link set dev ${BRIDGE} up
ip link set dev ${SRC} up
ip link set dev ${DST} up
# Enable promiscuous mode on the source interface
ip link set ${SRC} promisc on
# Add the ports to the bridge and enable a traffic mirror from SRC to DST
ovs-vsctl \
-- add-port ${BRIDGE} ${SRC} \
-- add-port ${BRIDGE} ${DST} \
-- --id=@${SRC} get port ${SRC} \
-- --id=@${DST} get port ${DST} \
-- --id=@${MIRROR} create mirror \
 name=${MIRROR} \
 select-src-port=@${SRC} \
 select-dst-port=@${SRC} \
 output-port=@${DST} \
-- set bridge ${BRIDGE} mirrors=@${MIRROR}
```

tc:

```
SRC_IFACE="eth0"
DST_IFACE="vnet0"
# ingress
tc qdisc add dev $SRC_IFACE ingress
tc filter add dev $SRC_IFACE parent ffff: \
          protocol all \
          u32 match u8 0 0 \
          action mirred egress mirror dev $DST_IFACE
# egress
tc qdisc add dev $SRC_IFACE handle 1: root prio
tc filter add dev $SRC_IFACE parent 1: \
          protocol all \
          u32 match u8 0 0 \
          action mirred egress mirror dev $DST_IFACE
```

Jumbo Frame Support

If your deployment requires jumbo frame support, please see the below additional guidance:

- To process jumbo frames, the MTU needs to be set to 9000 on both the host bridge and the vnet interface used for capture.
- "**virsh net-info <network**" can be used to show the bridge associated with the network you have defined as your capture network. In the below example, "**capture**" is the network name being used for capture. In this example the bridge is "**virbr1**".

```
root@kvm1:~# virsh net-info capture
Name:           capture
UUID:           eb04c023-18d2-4aa5-93fd-de2e6e9f9036
Active:         yes
Persistent:     yes
Autostart:      yes
Bridge:         virbr1
```

- If the bridge is set to 9000 before deploying your Vectra KVM vSensor then the target device will be set to 9000 automatically during the vSensor deployment.
- If you need to set MTU to 9000 after the vSensor has been deployed, "**virsh dumpxml <sensor name>**" will show the bridge and the associated target device that need to be set to 9000. Replace **<sensor name>** with the name of your Vectra KVM vSensor name.
  - Please note that the output of this command is long, and we only care about the interface sections.

```
root@kvm1:~# virsh dumpxml <sensor name>
...
    <interface type='bridge'>
      <mac address='52:54:00:34:63:37'/>
      <source bridge='br0'/>
      <target dev='vnet5'/>
      <model type='virtio'/>
      <alias name='net0'/>
      <address type='pci' domain='0x0000' bus='0x01' slot='0x00' function='0x0'/>
    </interface>
    <interface type='network'>
      <mac address='52:54:00:13:b8:a5'/>
      <source network='capture' portid='4ca15199-5414-4356-b783-2ef1d04d205a' bridge='virbr1'/>
      <target dev='vnet6'/>
      <model type='virtio'/>
      <alias name='net1'/>
      <address type='pci' domain='0x0000' bus='0x06' slot='0x00' function='0x0'/>
    </interface>
...
```

  - In the above (output only showing the necessary sections) example, you can see the bridge and target device (virbr1 and vnet6) are highlighted.

- The command to set the MTU to 9000 on a network (bridge or target device/interface) is "**ip link set dev <network> mtu 9000**". The command has no output as seen in the example below:

```
root@kvm1:~# ip link set dev vnet6 mtu 9000
```

- You can validate the current MTU settings with the "**ip a**" command. Please see abridged example below:

```
root@kvm1:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN group default qlen 1000     link/loop
back 00:00:00:00:00:00 brd 00:00:00:00:00:00     inet 127.0.0.1/8 scope host lo        valid_lft forever
 preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq master br0 state UP group default qlen 1000
    link/ether 2c:ea:7f:78:aa:5b brd ff:ff:ff:ff:ff:ff
…
15: virbr1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9000 qdisc noqueue state UP group default qlen 1000
  link/ether 52:54:00:0d:21:25 brd ff:ff:ff:ff:ff:ff     inet 192.168.100.0/24 brd 192.168.100.255 scope
 global virbr1        valid_lft forever preferred_lft forever 16: virbr1-
…
190: vnet6: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9000 qdisc fq_codel master virbr1 state UNKNOWN group
default qlen 1000     link/ether fe:54:00:13:b8:a5 brd ff:ff:ff:ff:ff:ff
```

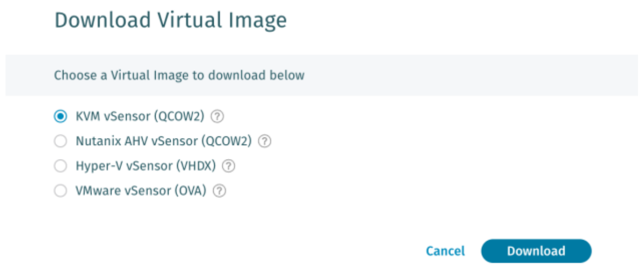
Basic Interaction with KVM Virtual Machines

The easiest way to interact with KVM VMs via the command line is by using the "**virsh**" series of commands. Vectra's deployment script will automatically start and configure the guest you, however these basic commands could prove helpful when diagnosing setup issues:

- View the list of configured virtual machines:
  - **virsh list --all**
- Start - In order to start a given virtual machine for which a configuration already exists:
  - **virsh start <name of vm>**
- Stop and save state:
  - **virsh shutdown <name of vm>**
- Stop quickly and don't save state (used only when you know you are going to purge the vm):
  - **virsh destroy <name of vm>**
- Purge the vm after stopping (needs to be used in tandem with destroy):
  - **virsh undefine <name of vm>**
- View the IP address(es) and MAC address(es) of a given virtual machine and its interfaces:
  - **virsh qemu-agent-command <name of vm> '{"execute":"guest-network-get-interfaces"}' | jq .**

Downloading the KVM vSensor Image

The KVM vSensor image is available under *Data Sources* > *Network* > *Sensors* in your Vectra UI. Navigate to this area, click "Download Virtual Image" at the top right, and select the KVM vSensor (QCOW2) option.

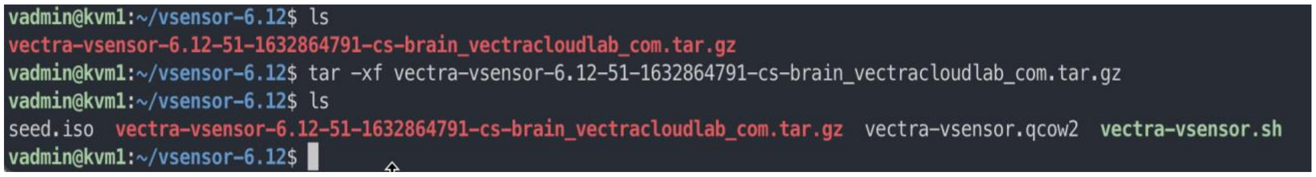

Download Virtual Image

Choose a Virtual Image to download below

[✓ KVM vSensor (QCOW2)]
Nutanix AHV vSensor (QCOW2)
Hyper-V vSensor (VHDX)
VMware vSensor (OVA)

KVM vSensor Deployment

- Once downloaded, the machine image can be placed in the directory of your choice and uncompressed. It is a tar file. On an Ubuntu machine the "**tar -xf**" command will uncompress the file fully. Using other tools may require 2 steps.



```
vadmin@kvm1:~/vsensor-6.12$ ls
vectra-vsensor-6.12-51-1632864791-cs-brain_vectracloudlab_com.tar.gz
vadmin@kvm1:~/vsensor-6.12$ tar -xf vectra-vsensor-6.12-51-1632864791-cs-brain_vectracloudlab_com.tar.gz
vadmin@kvm1:~/vsensor-6.12$ ls
seed.iso  vectra-vsensor-6.12-51-1632864791-cs-brain_vectracloudlab_com.tar.gz  vectra-vsensor.qcow2  vectra-vsensor.sh
vadmin@kvm1:~/vsensor-6.12$
```

- The script to start the deployment is "**vectra-vsensor.sh**" which has help that can be displayed as such:

```
vadmin@kvm1:~/vsensor-6.12$ ./vectra-vsensor.sh -h
6.12.0-14-11
Vectra Networks
KVM deployment script for Vectra virtual appliances.

USAGE: ./vectra-vsensor.sh [NAME] [DISK] [DEST_PATH] [MGT_NETWORK] [CAPTURE_NETWORK] [CONFIG_ID]

All arguments may be optionally provided as environment variables.

ARGS:
NAME              The name of the VM to be created.
                      default: vectra-vsensor
DISK              The path to the qcow2 disk image to either attach directly or clone.
                      default: /home/vadmin/vsensor-6.12/vectra-vsensor.qcow2
DEST_PATH         The storage path for the new VM. If this is not the same directory as the disk image,
                    then the image will be copied to the path before attachment.
                      default: /home/vadmin/vsensor-6.12/
MGT_NETWORK       The KVM network name to be used for management.
                      default: User selection
CAPTURE_NETWORK   The KVM network name to be used for traffic capture.
                  Not applicable for brain and stream modes.
                      default: User selection
CONFIG_ID         The index of the VM config to use. Run without args to view the options
                      default: User selection

CONFIGURATIONS:
ID        Flavor    CPUs      Memory    Disk
0         2core     2         8192      100G
1         4core     4         8192      150G
2         8core     8         16384     150G
3         16core    16        65536     500G

NETWORKS:
ID   Name
0    capture
1    default
```

- Execute the script to begin the deployment:

```
vadmin@kvm1:~/vsensor-6.12$ ./vectra-vsensor.sh tbilen-sensor
[Thu Oct 14 19:14:25 UTC 2021] - Setting VM configuration
ID        Flavor    CPUs      Memory    Disk
0         2core     2         8192      100G
1         4core     4         8192      150G
2         8core     8         16384     150G
3         16core    16        65536     500G
Select a configuration [0]: 0

[Thu Oct 14 19:14:30 UTC 2021] - Setting management NIC
```

```
ID   Name
0    capture
1    default
Select a management network [0]: 1

[Thu Oct 14 19:14:35 UTC 2021] - Setting capture NIC
ID   Name
0    capture
1    default
Select a capture network [0]: 0

[Thu Oct 14 19:14:41 UTC 2021] - Resizing disk
[Thu Oct 14 19:14:41 UTC 2021] - Creating VM: tbilen-sensor
[Thu Oct 14 19:14:42 UTC 2021] - Attaching brain data ISO
[Thu Oct 14 19:14:42 UTC 2021] - Adding capture NIC
[Thu Oct 14 19:14:42 UTC 2021] - Starting VM: tbilen-sensor
```

- As you can see in the above output, you will need to select some options:
  - **VM configuration** - See the KVM vSensor sizing in the earlier section of this document for details on the expected performance and resource requirements for each configuration.
  - **Management NIC** – Which NIC to use for the management interface of the vSensor.
  - **Capture NIC** – Which NIC to use for the capture interface of the vSensor.
- Your KVM vSensor will start automatically once the deployment script has finished.

Initial vSensor Configuration at the CLI

If you are not using DHCP or would like to set a static address, you will need to login to the CLI of the vSensor to set a static interface assignment. DNS for vSensors must also be configured at the CLI.

vSensor login at the CLI is very similar to logging in to physical Sensors. The primary difference is that there is no physical serial console, IPMI/iDRAC, or other ports to log in to. Logging in can be done via your hypervisor console function or using SSH to the management port if it was preconfigured with DHCP.

- Connect to your vSensor CLI using your hypervisor console or "**`ssh vectra@<IP or Hostname>`**" if you use DHCP and already know the address or hostname.
  - If the vSensor has shown up in the Brain UI then you should be able to see its IP address in *Data Sources* > *Network* > *Sensors* as well.
- Once logged in to the appliance you can view command syntax for the "**`set interface`**" command.

```
set interface -h
Usage: set interface [OPTIONS] [mgt1] [dhcp|static] [IP] [SUBNET_MASK]
[GATEWAY_ADDRESS]

Sets mgt1 to either dhcp or static ip configuration

Options:
-h, --help Show this message and exit.
```

- Setting the IP address statically:
  - In v8.5 and higher of Vectra software, IPv6 is supported for the MGT1 interface. For full details, including information regarding dual stack support, please IPv6 Management Support for Vectra Appliances on the Vectra support portal. Below we will show how to enable IPv6 support (its off by default) and the syntax to use when setting an IPv4 or IPv6 address.

- To enable/disable IPv6 support

```
# show ipv6 enabled
IPv6 is disabled

# set ipv6 enabled
Response: ok

# show ipv6 enabled
IPv6 is enabled

# set ipv6 disabled
Response: ok
```

- Setting IPv4 and IPv6 syntax examples:

  Execute the following command to set the MGT1 interface to the desired static IP address:

```
IPv4 Syntax:
set interface mgt1 static x.x.x.x y.y.y.y z.z.z.z

Where:
x.x.x.x is the desired interface IP address
y.y.y.y is the desired interface network mask
z.z.z.z is the desired gateway

IPv6 Syntax:
set interface mgt1 static [IPv6 IP] [Subnet Mask] [Gateway]

Example:
set interface mgt1 static 2001:0db8:0:f101::25 64 2001:0db8:0:f101::1
```

- To change back to DHCP (default):

```
set interface eth0 dhcp
```

- Configure DNS for the appliance:

  Command syntax to set DNS (up to 3 nameservers are supported):

```
set dns [nameserver1 <ip>] [nameserver2 <ip>] [nameserver3 <ip>]
```

  Example:

```
set dns 10.50.10.101 10.50.10.102
```

  Verifying DNS Configuration:

```
show dns
```

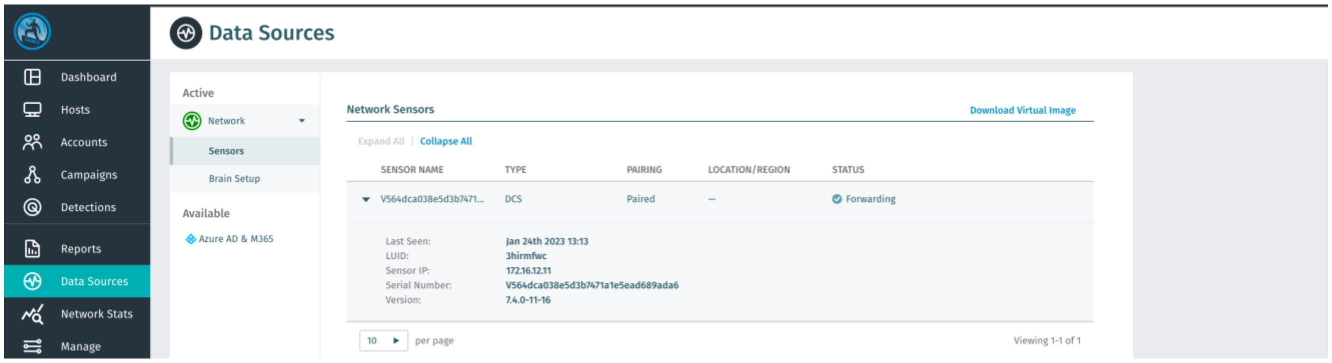
- Once you have set an IP and DNS, please use the "**`set password`**" command to change the password or you may wait and change all paired Sensor passwords en masse in the Brain UI later at *Data Sources > Network > Sensors > Sensor Configuration > CLI Password (Sensors)* if you wish to keep them in sync.

Example:

```
vscli > set interface mgt1 static 172.16.12.11 255.255.255.0 172.16.12.1
Interfaces updated successfully
vscli > set dns 10.50.10.101
DNS Set: success
vscli > show interface
mgt1:
    Running:
        Gateway: 172.16.12.1,
        Ip: 172.16.12.11,
        Link Speed: 10Gbps,
        Link State: up,
        Mac: 00:0c:29:89:ad:a6,
        Mode: static,
        Netmask: 255.255.255.0
vscli > show dns
Id|Server      |Description
1  10.50.10.101 Configured DNS nameserver
```

Pairing KVM vSensors

- vSensors do not offer a web UI.
    - The VectraPlatform (Brain) has the GUI for vSensor management at *Data Sources* > *Network* > *Sensors*.
    - Some configuration of the vSensor can be done at the CLI of the vSensor using the **"vectra"** user.
- vSensor images are already associated with the Brain they were downloaded from and will appear in the UI at *Data Sources* > *Network* > *Sensors* page when booted for the first time.
- vSensors are unique to the Brain that the image was downloaded from and cannot normally be paired to other Brains in your environment.
    - If a **"Sensor Registration Token"** is used, a deployed vSensor can be paired to a Brain other than the one the image was originally downloaded from.
    - This is covered in the below Additional Pairing Guidance section below.
- As per the Vectra Respond UX Deployment Guide or Vectra Quadrant UX Deployment Guide, Sensors, including vSensors, support pairing by IP or by hostname.
    - Pairing by hostname is preferred in failover scenarios. See guide above for more details.
- Once the vSensor is powered on and the interface configuration is set, the vSensor will announce itself to the Brain.
    - This can take a couple of minutes, check firewall rules if there is an issue.
    - If the vSensor appears in the Brain *Data Sources* > *Network* > *Sensors* page, then it has made a successful HTTPS connection to the Brain.
    - If the vSensor does not appear in the Brain *Data Sources* > *Network* > *Sensors* page, check that the vSensor has IP connectivity and that TCP port 443 (HTTPS) is permitted through your firewall.
    - If the vSensor is unable to pair with the Brain, complete its initial update or forward metadata to the Brain check that TCP port 22 (SSH) is permitted through your firewall.
- If the announce is successful, the vSensor will appear at the *Data Sources* > *Network* > *Sensors* page.
- If **"Auto Pairing"** is enabled in *Data Sources* > *Network* > *Sensors* > *Sensor Configuration* > *Sensor Pairing and Registration*, the pairing process will begin automatically.
    - Enabling **"Auto Pairing"** is a best practice during rollout.
- If **"Auto Pairing"** is not enabled, the vSensor must be manually paired by clicking on the "Pair" icon .
    - You will then be presented with a dialog box where you can start the pairing process.
- Initially you will see the **"Pairing Status"** as **"Pairing"** once the vSensor has successfully announced itself to the Brain.
    - Once pairing is complete, the **"Pairing Status"** will change to **"Paired"**, and the **"Status"** should change to **"Forwarding"** once traffic is successfully being forwarded from the vSensor to the Brain.



Network Sensors

SENSOR NAME	TYPE	PAIRING	LOCATION/REGION	STATUS
V564dca038e5d3b7471...	DCS	Paired	—	Forwarding

Last Seen: Jan 24th 2023 13:13
LUID: 3hirmfwc
Sensor IP: 172.16.12.11
Serial Number: V564dca038e5d3b7471a1e5ead689ada6
Version: 7.4.0-11-16

- **Please note:**
  - vSensors, like physical Sensors, will update themselves to stay current with their Brain.
  - After pairing, the vSensor will update by receiving an update from the Brain.
    - This process is automatic, and no input is required.
  - Certain vSensor CLI functions and traffic functions will become available only after the vSensor has fully updated.
  - Depending on the specific version of the vSensor, you may see errors or warnings when running CLI functions during the period of time when the vSensor is still updating.
- The vSensor can be renamed or have its location labeled as desired by clicking on the pencil icon on the right of the vSensor.

Additional Pairing Guidance

Pairing with new or changed Brains:

- If you have a backup of your Brain and restore it to a new Brain with the same configuration (IP or hostname), previously paired Sensors (including vSensors) will connect to the new Brain automatically as the Sensor state is saved in the backup.
- If the Brain IP has changed but otherwise remains the same, the vSensors may be updated to the new IP address using the "`set brain`" command.
- Existing tunnels have to terminate to re-establish connection to a new Brain. This can be accomplished a few different ways.
  - Naturally, because the original Brain is no longer reachable due to firewall change, hardware or software failure, etc.
  - Unpairing the vSensor from the original Brain and having the vSensor attempting communication to the original Brain.
  - Using the "`set brain`" command at the CLI will terminate an existing tunnel and attempt to start pairing with a new Brain.
- If you have a Brain that will not be restored from backup that you wish to pair an existing vSensor to, this is possible via the use of the "Sensor Registration Token".
  - Retrieve or generate a current Sensor Registration Token from *Data Sources > Network > Sensors > Sensor Configuration > Sensor Pairing and Registration* in the Brain GUI.
  - Perform the "`set registration-token <token>`" command at the Sensor CLI.
  - Finally perform the "`set brain <IP or Hostname>`" command at the Sensor CLI (depending on if you have selected to pair via the management IP or DNS name in *Data Sources > Network > Sensors > Sensor Configuration > Sensor Pairing and Registration).*

vSensors and Pairing by Hostname vs IP

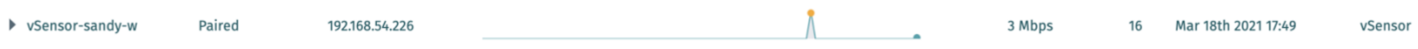
- The vSensor image downloaded from the Brain will use, by default, the Brain's IP address for pairing.
- You will need to set the *Data Sources > Network > Sensors > Sensor Configuration > Sensor Pairing and Registration* **"Pair using DNS name"** option to generate the virtual machine image that points at a hostname.
- When this setting is changed, it does not affect any previously paired (either by IP or Hostname) vSensors.

Traffic Validation

Please see the following Vectra support article for recommendations on network traffic that should be examined and excluded from analysis:

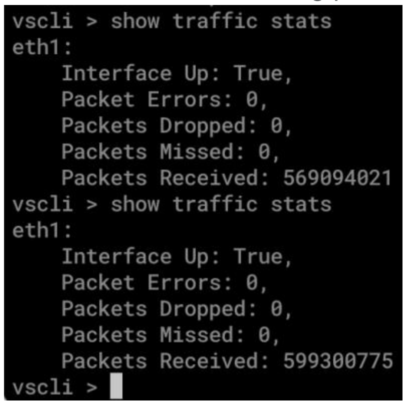
- Vectra Platform Network Traffic Recommendations

For a quick spot check to see that you are receiving any traffic at all via the vSensor you many want to check the GUI and/or CLI for statistics. If the vSensor is seeing more than 1 Mbps of traffic, this will show in the GUI under *Network Stats > Ingested Traffic* after a few minutes.



vSensor-sandy-w	Paired	192.168.54.226	3 Mbps	16	Mar 18th 2021 17:49	vSensor

- You can see traffic flow immediately at the CLI of the Sensor using the "**`show traffic stats`**" command.
- Execute this command a few times in a row to see increasing packet counts.



```
vscli > show traffic stats
eth1:
    Interface Up: True,
    Packet Errors: 0,
    Packets Dropped: 0,
    Packets Missed: 0,
    Packets Received: 569094021
vscli > show traffic stats
eth1:
    Interface Up: True,
    Packet Errors: 0,
    Packets Dropped: 0,
    Packets Missed: 0,
    Packets Received: 599300775
vscli >
```

After sending traffic to your Sensors, it is a best practice to validate that the traffic observed meets quality standards required for accurate detection and processing. Vectra's Enhanced Network Traffic Validation feature provides alarms and metrics that can be used to validate the quality of your traffic. Please see the following Vectra support article for details:

- Enhanced Network Traffic Validation (CLI)

Worldwide Support Contact Information

- Support portal: https://support.vectra.ai/
- Email: support@vectra.ai (preferred contact method)
- Additional information: https://www.vectra.ai/support