Sensor Deployments on Virtualized and Cloud (AWS) Environments

PROXMOX

AWS

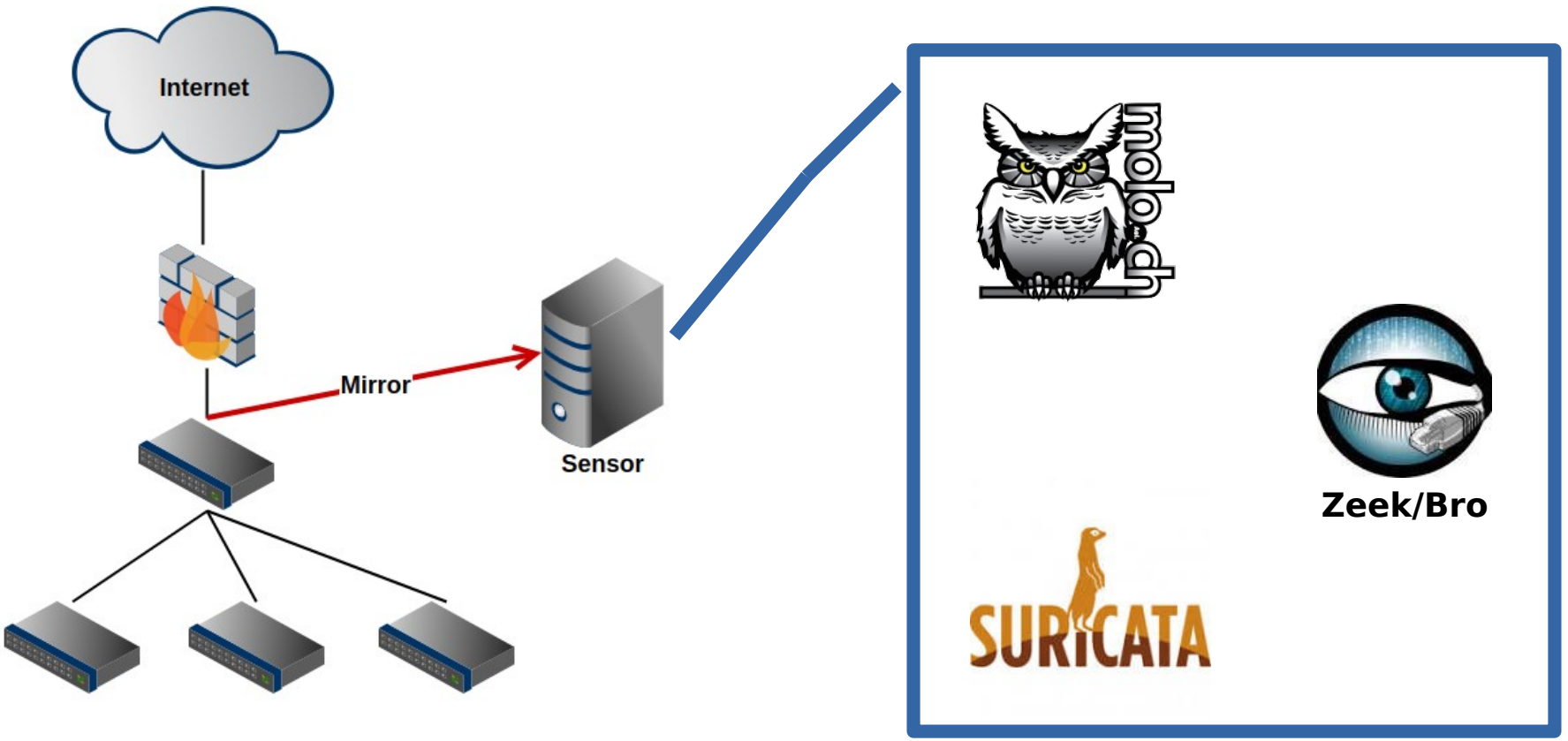
SURICATA

About Me

Srinath Mantripragada

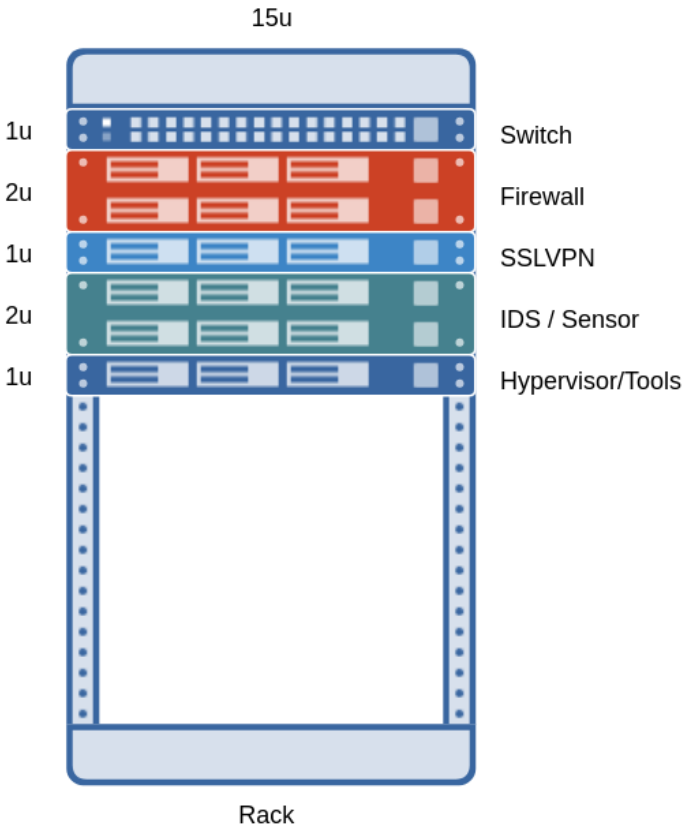
- Sysadmin and more recently DevOps.
- Free and Open Source based solutions.

Logical Diagram / Scope

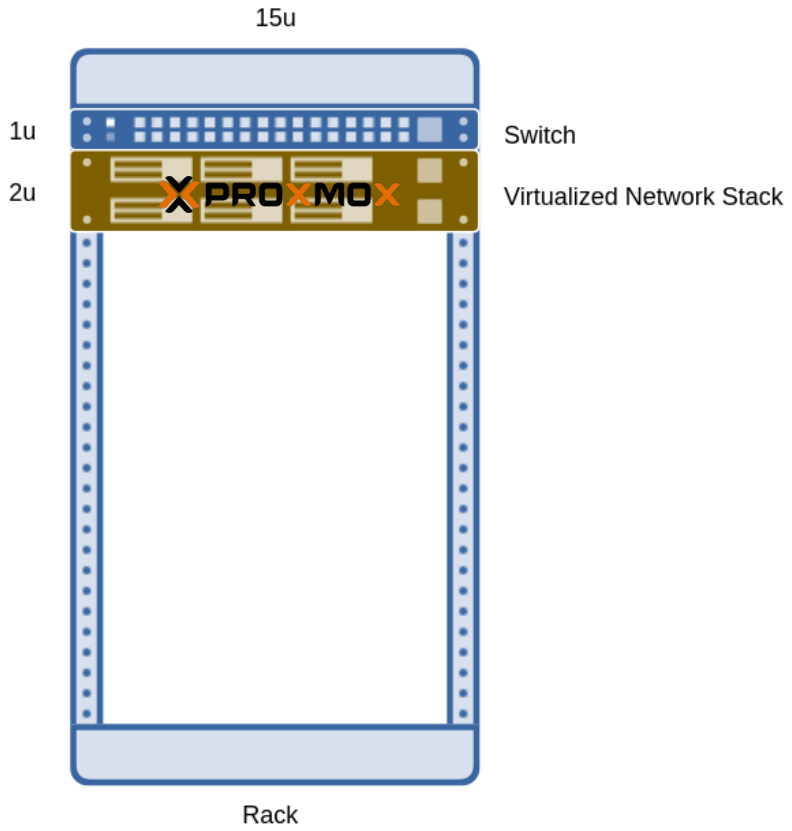


Sensor VM under Proxmox

Large / Medium Sites



Small Sites

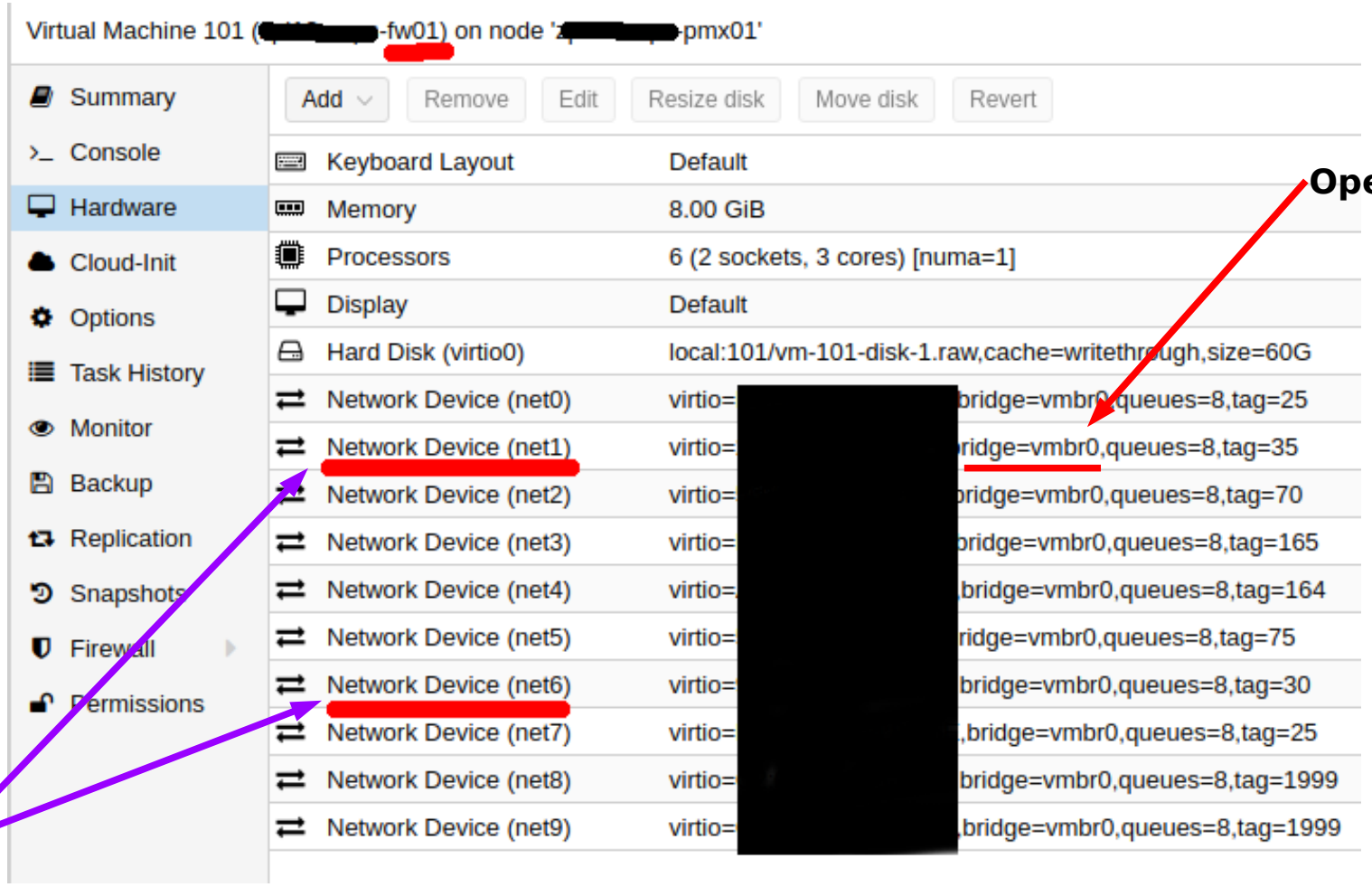


Sensor VM under Proxmox cont.

- Debian based;
- Appliance like install, but Linux-fu available under the hood;
- Since 2008 and frequent releases and feature additions;
- Web UI / Low overhead;
- Paid Support Option / Access to Stable Repository;
- **(OvS) Open vSwitch under the hood ;**
- **KernelCare (Kernel Live Patch)** (fewer reboots) ;

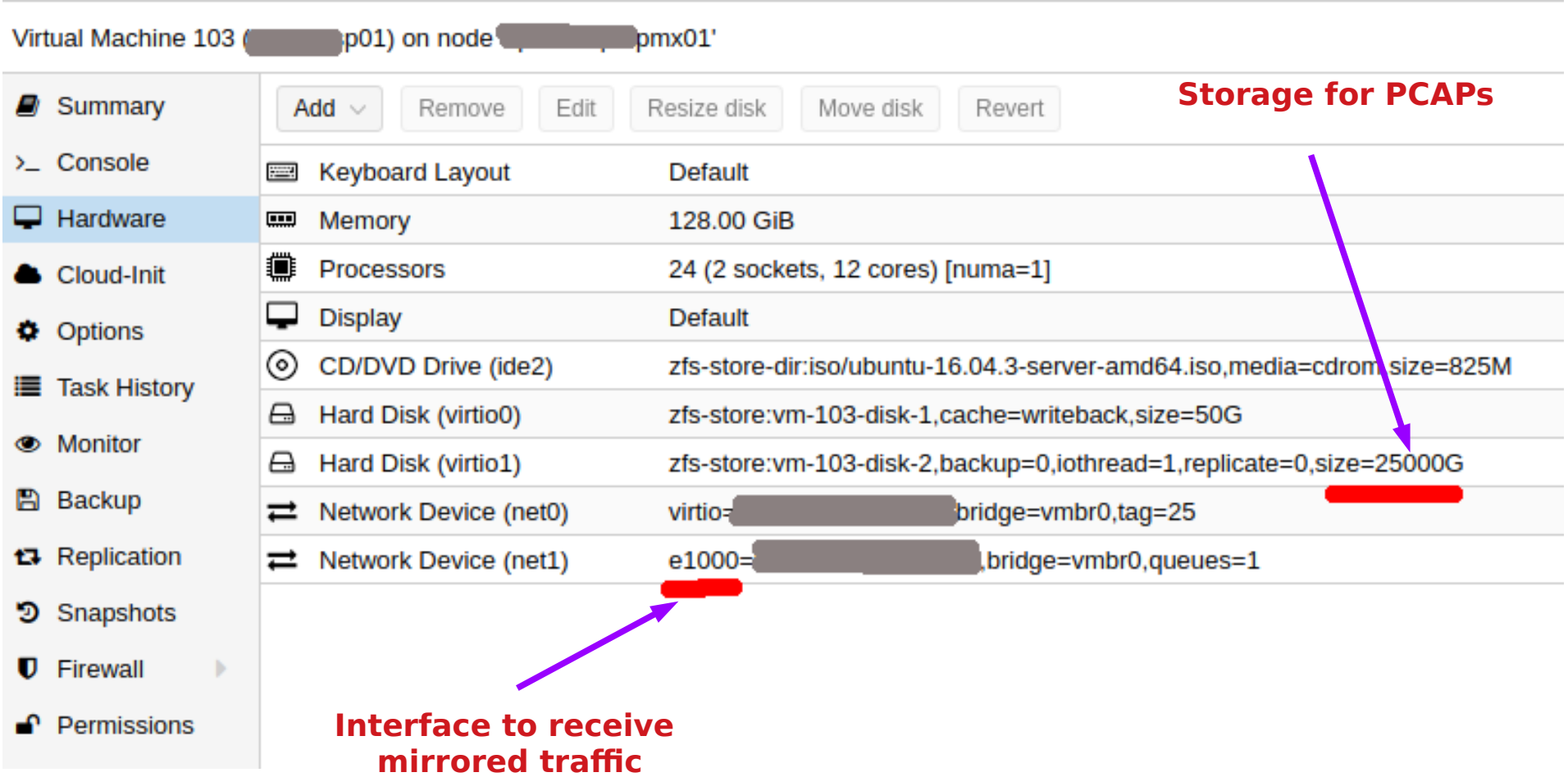
Sensor VM under Proxmox cont.

Firewall VM



Sensor VM under Proxmox cont.

Sensor VM



Sensor VM under Proxmox cont.

Mirror Traffic to Sensor

```
  ovs-vsctl \
      -- --id=@m create mirror name=mirror0 \
      -- add bridge vmbr0 mirrors @m \
      -- set mirror mirror0
select_src_port=tap101i1,tap101i6
select_dst_port=tap101i1,tap101i6 \
      -- set mirror mirror0 output-
port=tap103i1
```

MOLOCH

Create Mirror cont.

```
# ovs-vsctl list mirror
_uuid                : 45640cc4-b38e-4807-8bdf-45ee1967be8e
external_ids         : {}
name                 : "mirror0"
output_port          : 2d480ede-4047-49f5-841d-93c797e2688d
output_vlan          : []
select_all           : false
select_dst_port      : [00d85eb7-c47c-4cc7-9fa6-092cd1853711,
20eb6ca0-48a8-47db-8ed1-2fdaea2f98b9]
select_src_port      : [00d85eb7-c47c-4cc7-9fa6-092cd1853711,
20eb6ca0-48a8-47db-8ed1-2fdaea2f98b9]
select_vlan          : []
snaplen              : []
statistics           : {tx_bytes=2582237631,tx_packets=32296198}
```

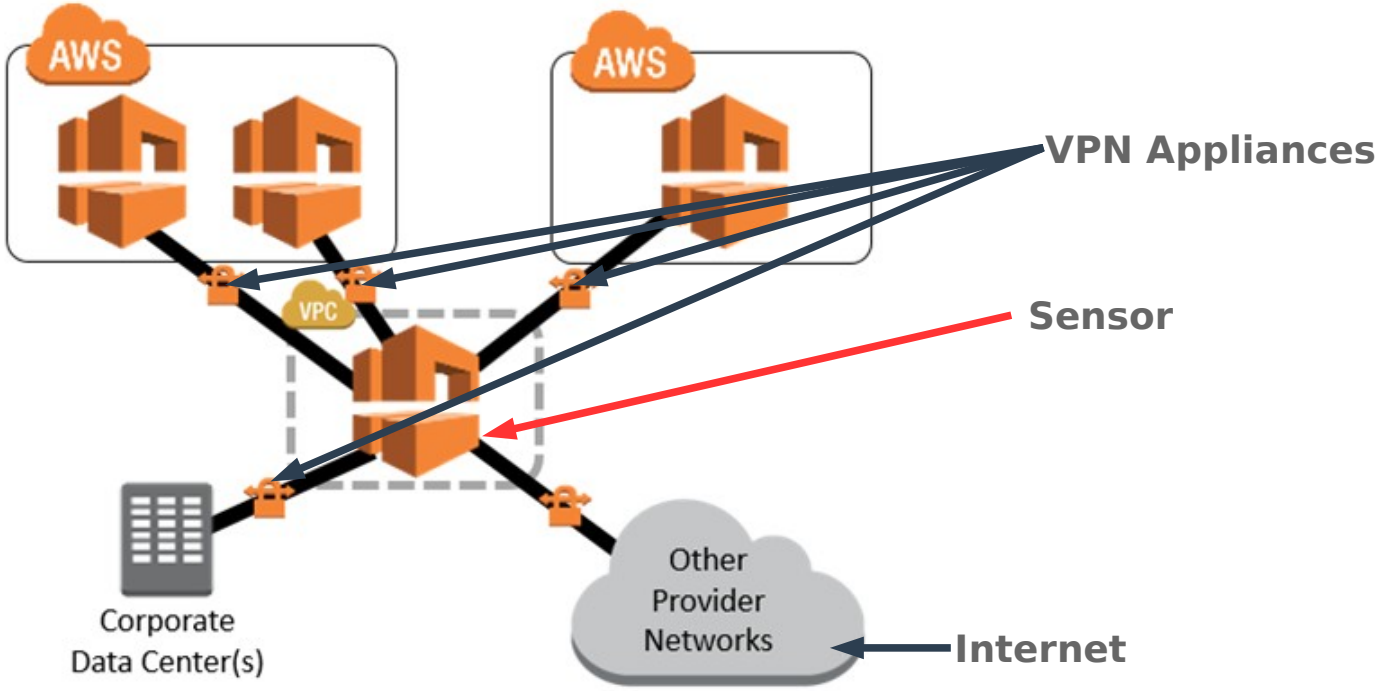
Create Mirror cont.

Disadvantages:

- Mirror setting is not part of Proxmox (will not appear anywhere in the UI)
  - Solution: Documentation
- **VM TAP interfaces are ephemeral**. If source or destination VM is stopped, the mirror setting will be invalidated an will disappear and will not be re-created when the VM is started.
  - Solution: Create a cronjob to re-create the mirror if necessary.
- **Limited Performance**: Network and Disk peformance limitations, good enough for lower bandwidth (~300Mbps)

Sensor on AWS Transit VPC (Hub-and-Spoke)

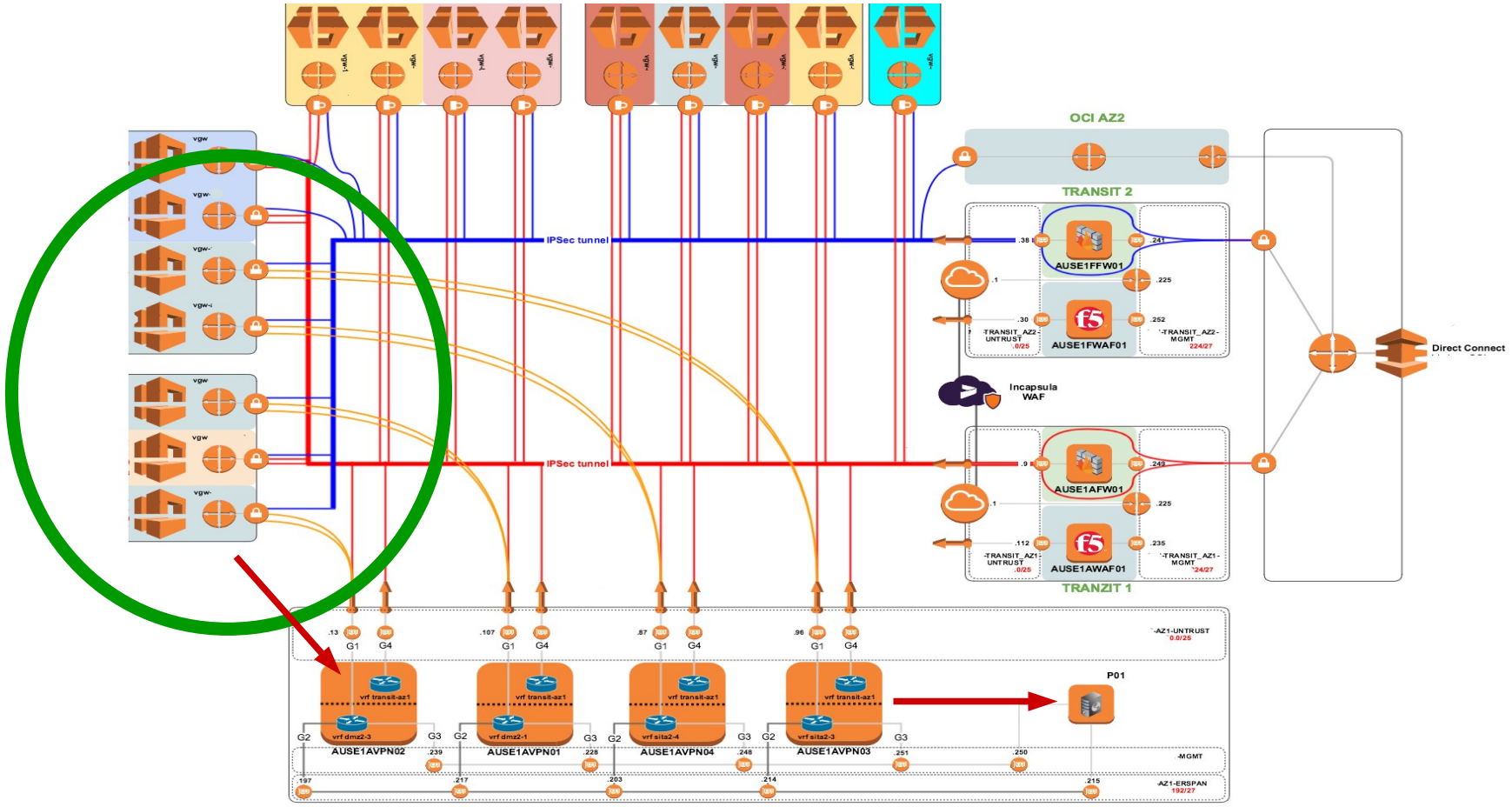
- Routes all traffic through a network **transit center** (a transit VPC)



* *https://aws.amazon.com/answers/networking/aws-global-transit-network/*

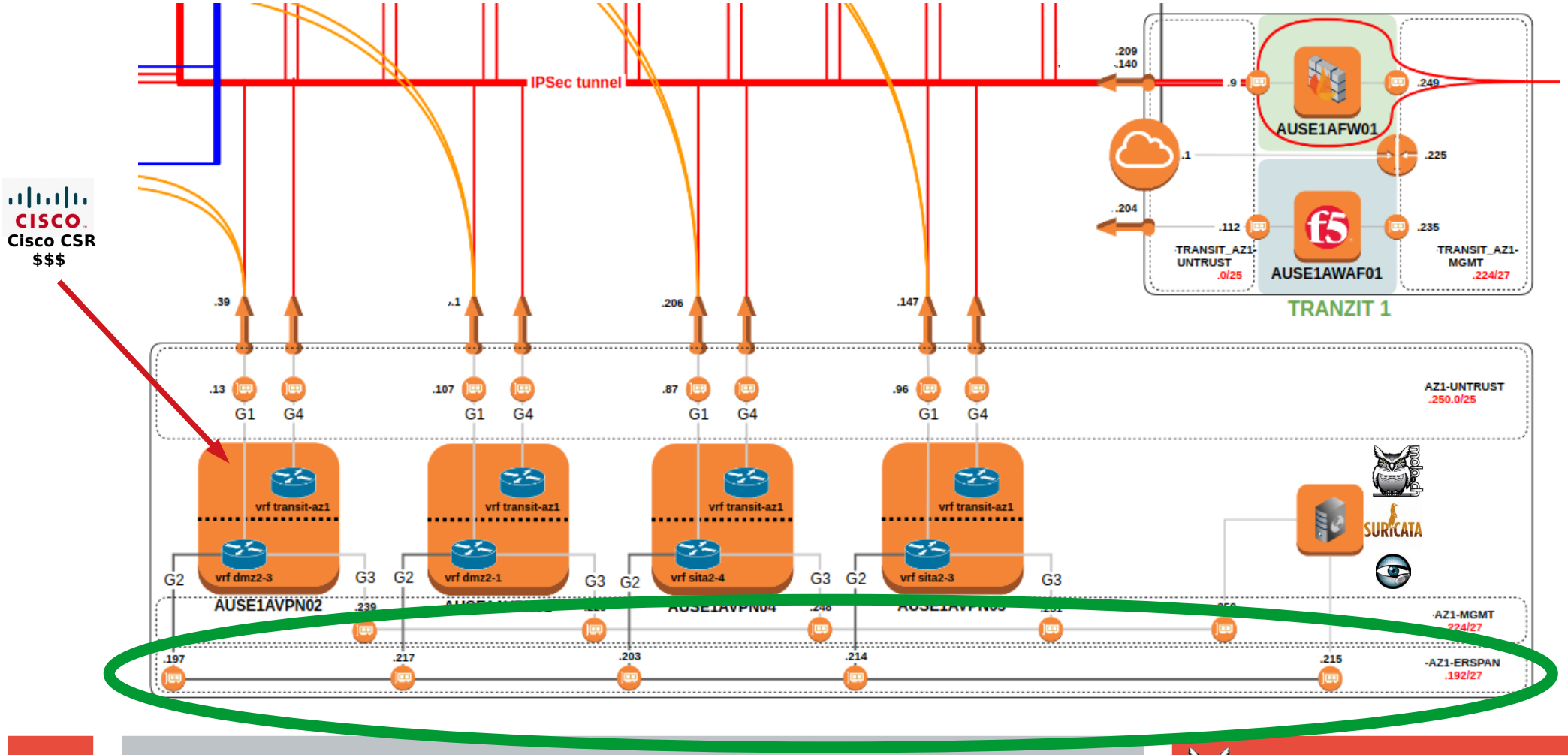
Sensor on AWS Transit VPC (Hub-and-Spoke)

- **How the actual environment looks like**



Sensor on AWS Transit VPC (Hub-and-Spoke)

- **Looking closer at the Sensor VPC**



Sensor on AWS Transit VPC (Hub-and-Spoke)

- **Cisco CSR Configuration**

```
monitor session 1 type erspan-source
 description ERSPAN DMZ2-1
 source interface Tu10 , Tu20
# Tunnel interfaces going to AWS VPC and Transit Firewall
 destination
  erspan-id 1
  mtu 1464
  ip address x.x.x.215
  origin ip address x.x.x.217
```

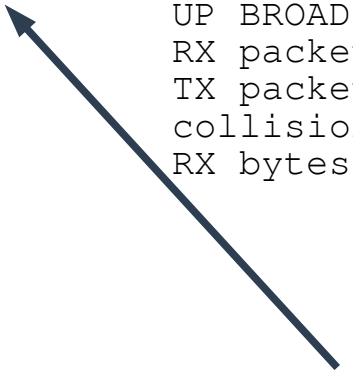
Sensor on AWS Transit VPC (Hub-and-Spoke)

Packet Capture Receiver / RCDCAP

```
ens4      Link encap:Ethernet  HWaddr xx:xx:xx:xx:xx:xx
          inet addr: 10.x.y.215  Bcast:10.X.y.223  Mask:255.255.255.224
          UP BROADCAST RUNNING MULTICAST  MTU:9000  Metric:1
          RX packets:25146086793 errors:0 dropped:0 overruns:0 frame:0
          TX packets:124662 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:13253959739761 (13.2 TB)  TX bytes:7898276 (7.8 MB)


# /usr/bin/rcdcap -i ens4 --erspan --tap-persist --tap-device mon0

mon0      Link encap:Ethernet  HWaddr xx:xx:xx:xx:xx:xx
          UP BROADCAST RUNNING  MTU:1500  Metric:1
          RX packets:25146201487 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:12097246297608 (12.0 TB)  TX bytes:0 (0.0 B)
```



Interface used by tools

Sensor on AWS Transit VPC (Hub-and-Spoke)

Disadvantages:

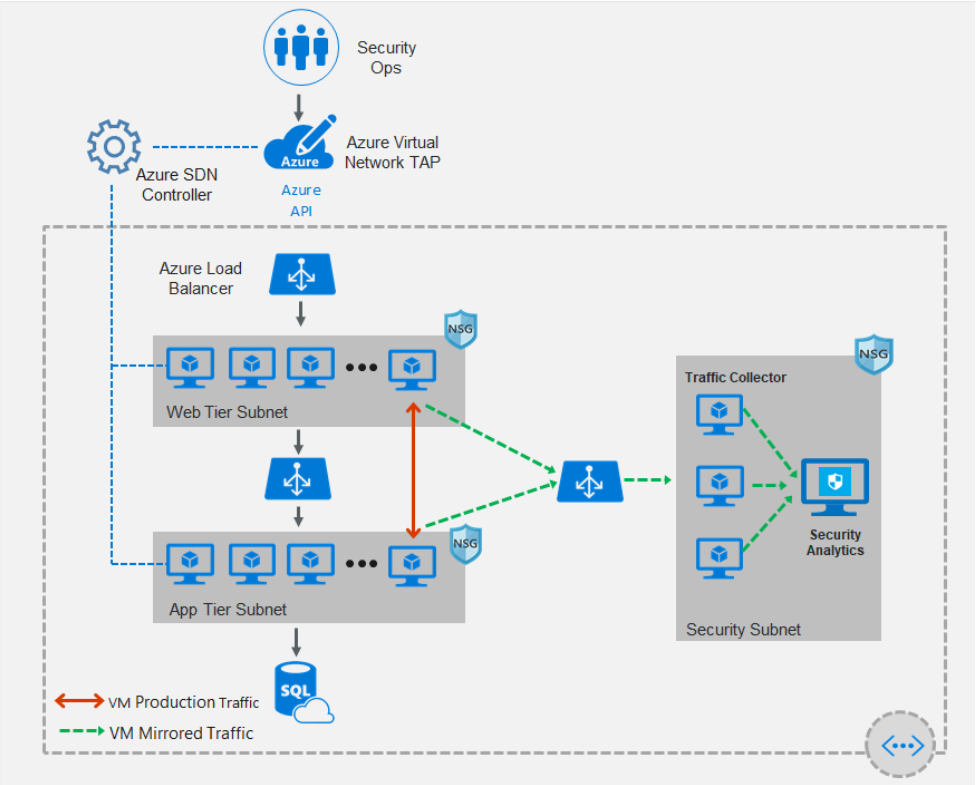
- Complex Infrastructure involving multiple VPC and tunnels.
- Start price at $9000 / year (Software + EC2)
- Not suitable for simpler/smaller environments

Alternatives:

- Use a Linux instance as VPN and fluxcap.
- Run capture inline on the router instance
  - OVS Mirror or IPTables “tee”
- Other commercial alternatives: GigaSECURE, CloudLens, Flowmon, Big Cloud Fabric.

Azure Cloud Virtual Network TAP

- **Recently announced Virtual Network TAP**
  - https://docs.microsoft.com/en-us/azure/virtual-network/virtual-network-tap-overview
- **Still in “developer preview”**
- **VXLAN Based**



References

- https://aws.amazon.com/answers/networking/aws-global-transit-network/
- https://www.openvswitch.org/
- https://backreference.org/2014/06/17/port-mirroring-with-linux-bridges/
- http://umap.openstreetmap.fr/
- https://docs.microsoft.com/en-us/azure/virtual-network/virtual-network-tap-overview
- https://github.com/troydhanson/fluxcap
- https://www.aptly.info/

Ansible and Automation

- **Initial OS install is done manually**
- **From initial OS installation, everything is configured via Ansible playbooks**
- **Playbook for individual components can be decoupled if necessary (Install Moloch but not Suricata and/or Bro/Zeek)**
- **Suricata, Bro and some helper tools are compiled and .deb packages are created for those**

Ansible and Automation

- **We maintain a custom APT repository that contains all packages not in the default ubuntu repository. So everything is installed and updated via APT.**
- **Custom compiled packages are created via Jenkins and DBuilder and copied over to the repository (aptly).**
- **Other packages are just copied from the original repositories to our custom repository via Jenkins tasks (Moloch, Elastic).**

Ansible and Automation

```
- name: "Setting boot kernel parameters"
        lineinfile:
          path:   "/etc/default/grub"
          regexp: "^GRUB_CMDLINE_LINUX_DEFAULT="
          line:   "GRUB_CMDLINE_LINUX_DEFAULT=\"isolcpus={{ isolcpus }}\""
        when: ( isolcpus is defined )
        notify: "Update grub"

- name:  "Log in to Moloch"
        shell: "curl -s -k --cookie-jar - --location --digest --user 'admin:
{{ moloch_admin_password }}' https://localhost:8005/ | grep MOLOCH-COOKIE | awk '{print $7}'"
        register: m_cookie

- name:  "Get Moloch user list"
        shell: "curl -s -k -XPOST --digest --user 'admin:{{ moloch_admin_password }}'
https://localhost:8005/user/list"
        register: m_userList

- name:  "Create users"
        shell: "{{moloch_install_dir}}/bin/moloch_add_user.sh {{ item }} '{{ all_users_db[item |
lower].user_fullname | default(item) }}' '{{password}}' --webauth --email"
        with_items:  "{{ moloch_users }}"
        when: ( item not in ( m_userList.stdout | from_json | json_query('data[*].id') ) )
```