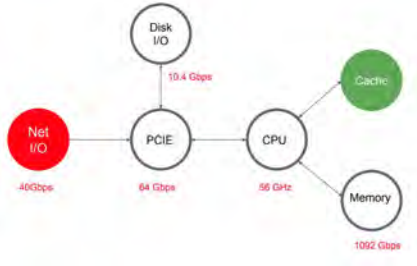
Enabling Suricata in the Cloud at Scale Using DPDK

Suricon, Boston, 10/22/2021

Troy Hanson, Shend Saliaga, Sruthi Yellamraju, James Ezick, Alison Ryan, Jordi Ros-Giralt

Reservoir Labs

632 Broadway 803, New York,
NY 10012,
United States



Roadmap

- R-Core Project: DPDK-based packet acquisition for Suricata and Zeek
  - High-performance features
  - Base architecture
  - Hardware filtering
  - R-Core shared library and plugin layer
- Deploying Suricata in the cloud using R-Core/DPDK
  - Existing solutions
  - Concepts: network load balancer and target groups
  - Health check
  - Deployment steps and data flow
  - Scaling

R-Core

DPDK-based Packet Forwarding Engine

R-Core Project

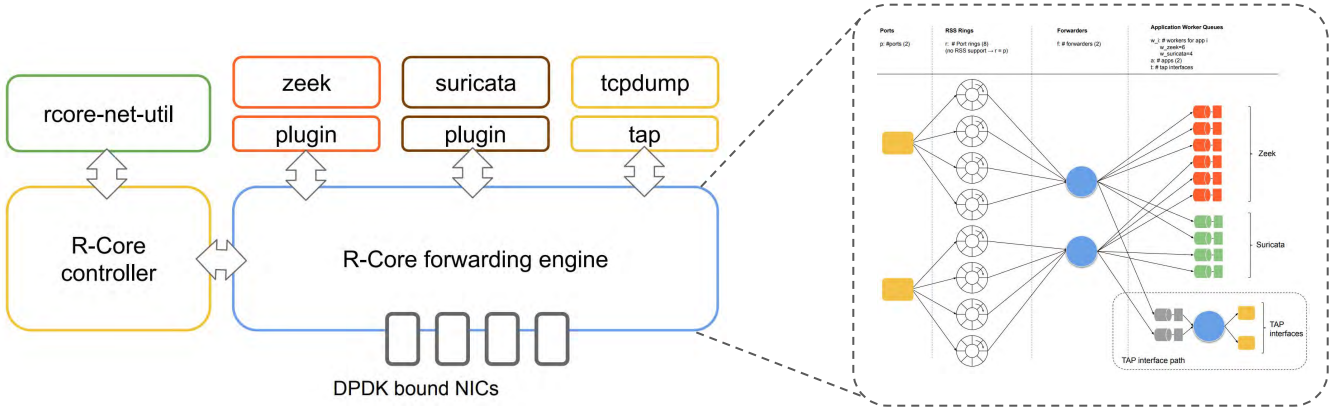
- A vendor independent high-performance packet forwarding engine
- Runs on DPDK-capable NICs, leverages DPDK high-performance features
- Forwards packets from the NIC to $N parallel applications, including Suricata, Zeek and Tcpdump
- Suricata integration:
  - Utilizes multi-threaded workers (*workers* runmode) for scalability
  - Easy configuration through suricata.yaml
  - Integrated with Suricata stats counters (packets, bytes, drops)

High-Performance Optimizations

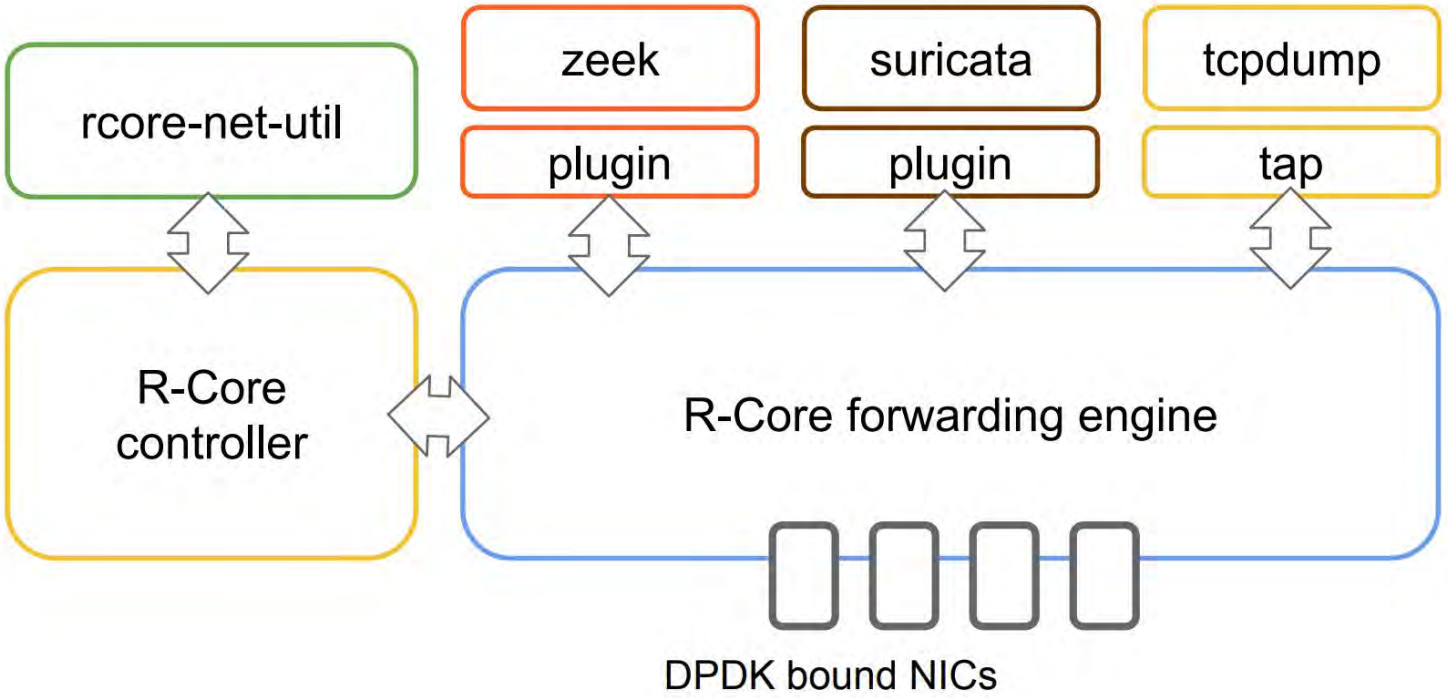
- Kernel bypass
- Zero packet copy
- Lockless data structures
- Intelligent packet shunting
- NUMA affinity
- CPU/core affinity/pinning
- Multi-core elastic scalability
- Optimizations described in:
  - "High-performance many-core networking: design and implementation" [https://dl.acm.org/doi/10.1145/2830318.2830319]
  - "Algorithms and Data Structures to Accelerate Network Analysis" [https://bit.ly/3E4h2V5].

Base Architecture

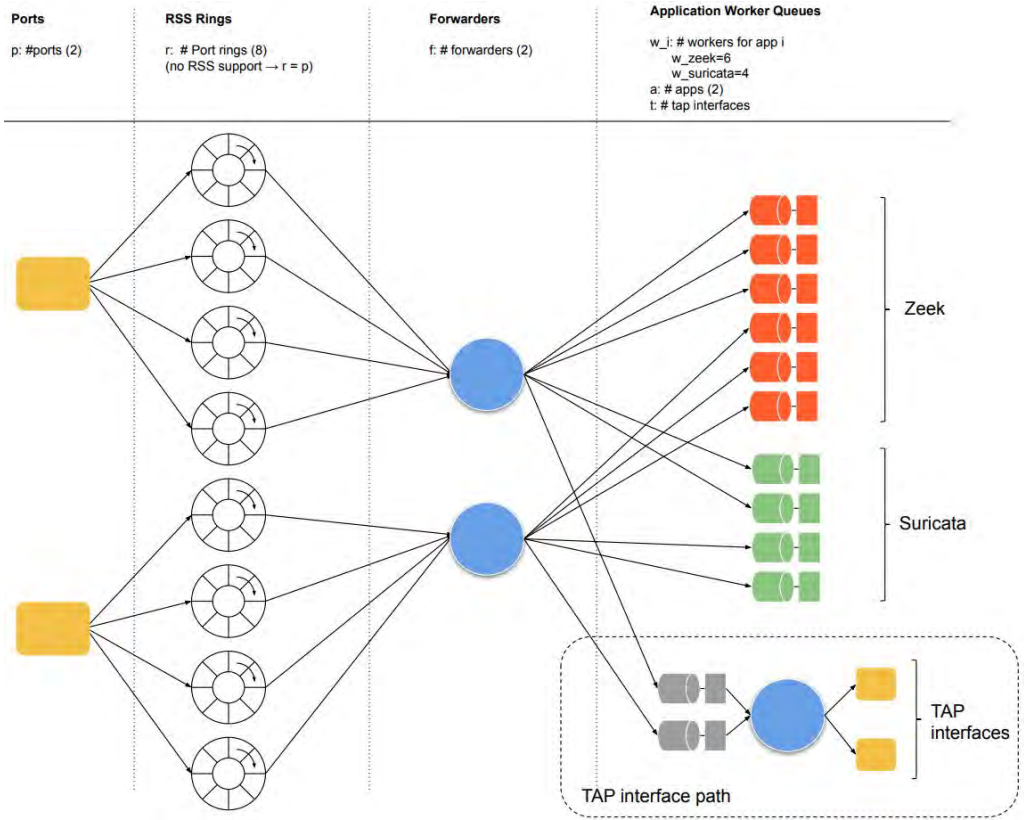
- R-Core controller: Manages the forwarding engine
- rcore-net-util: User CLI
- Plugins: Support for Zeek, Suricata, tcpdump and TAP
- Can filter traffic using DPDK rte_flow and BPF



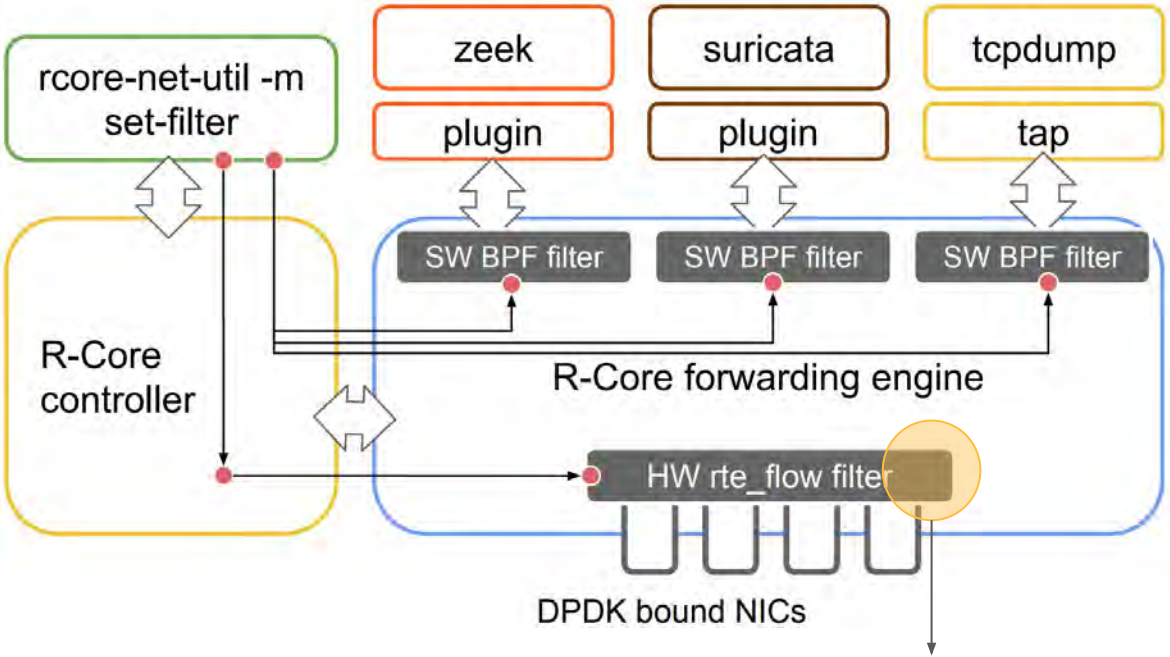
Base Architecture



Base Architecture



High-Performance Hardware Filtering

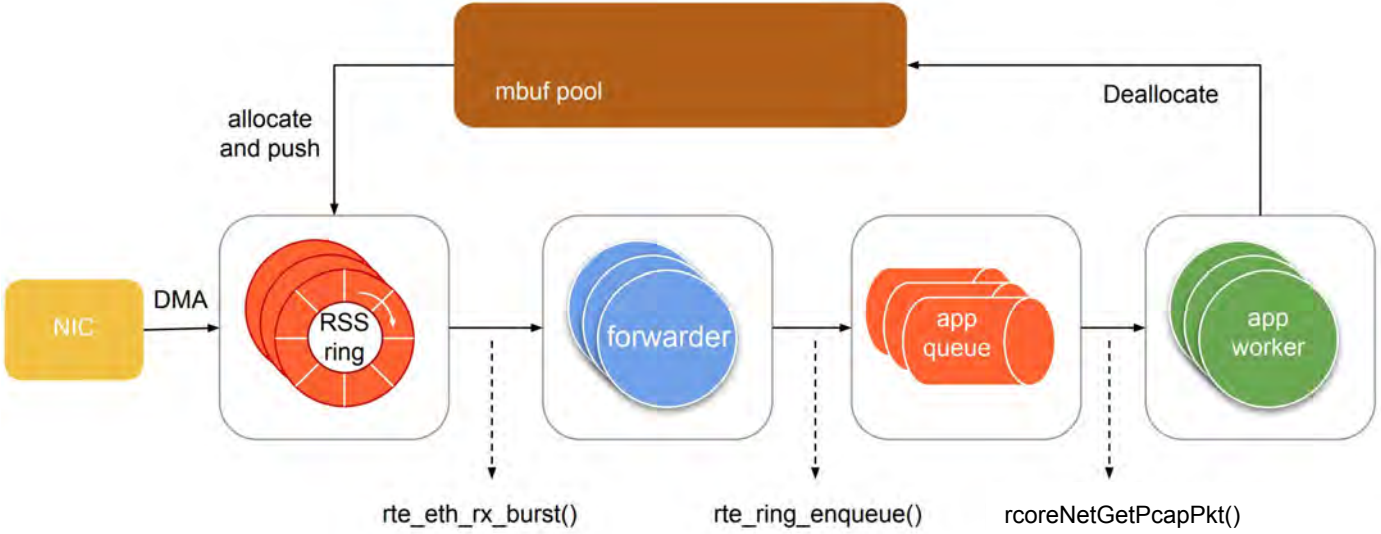


Examples:

- Flow shunting
- Flow forwarding
- Tunnel decapsulation

Based on DPDK RTE Flow:
https://doc.dpdk.org/guides/prog_guide/rte_flow.html

Life-cycle of a Packet



R-Core YAML Configuration

```
verbose_mode: 0             # Enable debug of mcore non-critical paths (Optional, default: 0)
mcore:                      # mCore Instance name
  - name: "amz1"            # Name of this mCore instance
    controller_core: 0      # Core where the controller is to be pinned (Optional, default: 0)
    forwarder_count: 4      # Number of forwarders to launch
    ring_count: 8           # Number of RSS rings to allocate in total
    ring_size: 1024         # Size in mbufs of each RSS ring
    worker_queue_size: 2048 # Size in mbufs of each application worker queue
    num_channels: 4         # Number of memory channels per socket (Optional, default: 2)
    socket_mem: 32          # Pre-allocated memory per socket in MB (Optional, default: 16)
    max_pkt_size: 9000      # Max packet size; used for jumbo frames (Optional, default: 9038)
    rss_offload_disable: 0  # Disable RSS offload to H/w
    rss_type: “ip_port”     # RSS hash type. Support values: “ip”, “ip_port”
    port:
      - id: "08:00.0"       # PCI ID of the port
        forwarder_core:     # Cores to pin this port’s forwarders on
          - 1
          - 2
      - id: "82:00.0"
        forwarder_core:
          - 14
          - 15
    application:
      - name: "zeek"        # ID name of an application
        worker_count: 8     # Number of app workers queues to allocate
      - name: "suri"
        worker_count: 4
      - name: "monitor"
        worker_count: 4
        type: "tunnel"      # This is a tap interface (Optional, default: no tunnel)
```

R-Core YAML Configuration

```
verbose_mode: 0              # Enable debug of mcore non-critical paths (Optional, default: 0)
mcore:                       # mCore Instance name
  - name: "amz1"             # Name of this mCore instance
    controller_core: 0       # Core where the controller is to be pinned (Optional, default: 0)
    forwarder_count: 4       # Number of forwarders to launch
    ring_count: 8            # Number of RSS rings to allocate in total
    ring_size: 1024          # Size in mbufs of each RSS ring
    worker_queue_size: 2048  # Size in mbufs of each application worker queue
    num_channels: 4          # Number of memory channels per socket (Optional, default: 2)
    socket_mem: 32           # Pre-allocated memory per socket in MB (Optional, default: 16)
    max_pkt_size: 9000       # Max packet size; used for jumbo frames (Optional, default: 9038)
    rss_offload_disable: 0   # Disable RSS offload to H/w
    rss_type: “ip_port”      # RSS hash type. Support values: “ip”, “ip_port”
    port:
      - id: "08:00.0"        # PCI ID of the port
        forwarder_core:      # Cores to pin this port’s forwarders on
          - 1
          - 2
      - id: "82:00.0"
        forwarder_core:
          - 14
          - 15
    application:
      - name: "suri"         # ID name of an application
        worker_count: 4      # Number of app workers queues to allocate
      - name: "zeek"
        worker_count: 8
      - name: "monitor"
        worker_count: 4
        type: "tunnel"       # This is a tap interface (Optional, default: no tunnel)
```

Recommended Pinning

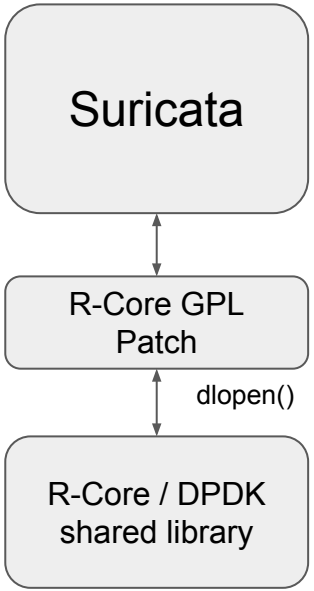
```
$ rcore-net-util -k key -m get-info

[clipped]

INFO,Controller: Recommended application affinities:
INFO,Controller:  [suri:1] -> node 0/core 7
INFO,Controller:  [suri:2] -> node 0/core 8
INFO,Controller:  [suri:3] -> node 1/core 20
INFO,Controller:  [suri:4] -> node 1/core 21
INFO,Controller:  [zeek:1] -> node 0/core 3
INFO,Controller:  [zeek:2] -> node 0/core 4
INFO,Controller:  [zeek:3] -> node 0/core 5
INFO,Controller:  [zeek:4] -> node 0/core 6
INFO,Controller:  [zeek:5] -> node 1/core 16
INFO,Controller:  [zeek:6] -> node 1/core 17
INFO,Controller:  [zeek:7] -> node 1/core 18
INFO,Controller:  [zeek:8] -> node 1/core 19

[clipped]
```

R-Core Shared Library and Plugin Layer



- Suricata now supports dynamically loadable plugins:
  - https://redmine.openinfosecfoundation.org/issues/3332
  - https://github.com/OISF/suricata/pull/5267
- Also based on dlopen():

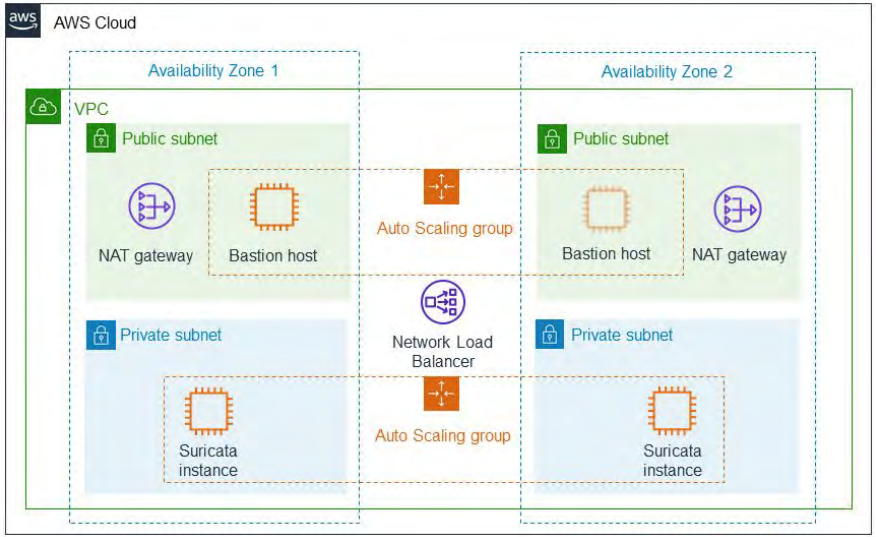
```
31  + static void InitPlugin(char *path)
32  + {
33  +     void *lib = dlopen(path, RTLD_NOW);
```

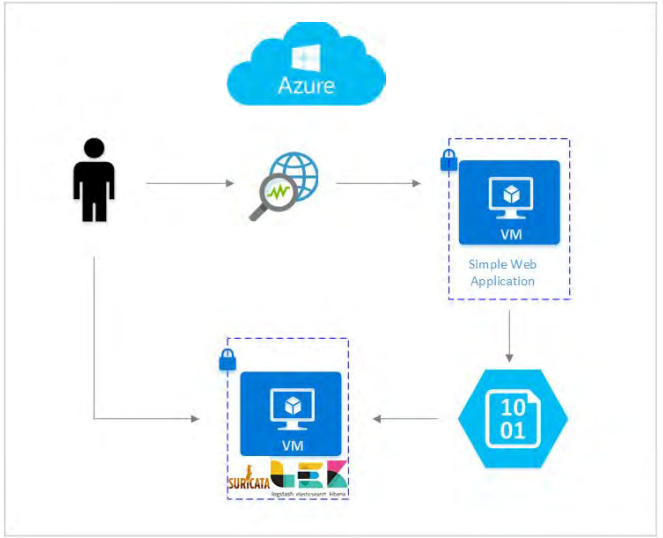
DPDK for Suricata

- Vipin Varghese initiated work to support DPDK for Suricata:
  - https://forum.suricata.io/t/discussion-for-dpdk-api-support-into-suricata/202
- Merge versus split mode discussion:
  - R-Core is a split mode implementation
- Pull request submitted (08/30/2021): https://github.com/OISF/suricata/pull/6317

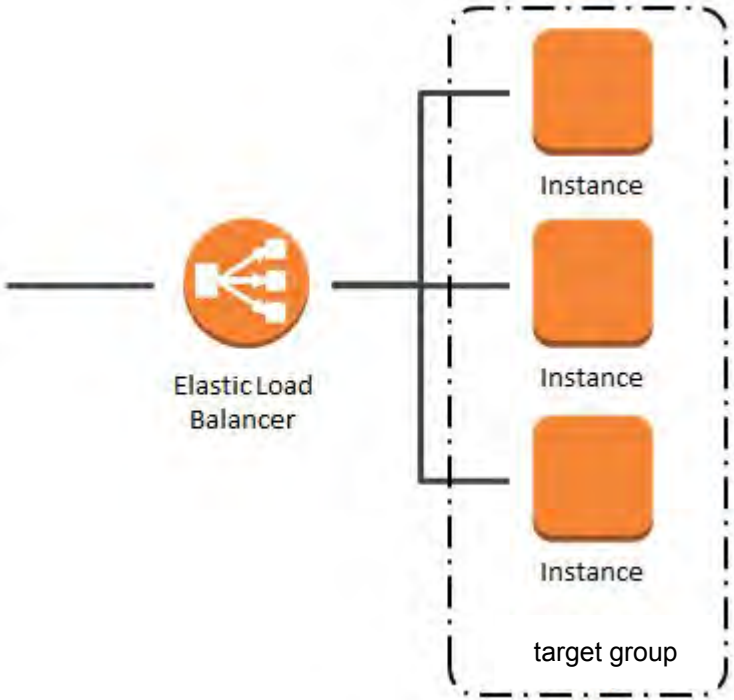
Suricata in the Cloud Using DPDK and Network Load Balancing

"Out of the Box" Suricata Cloud Deployments

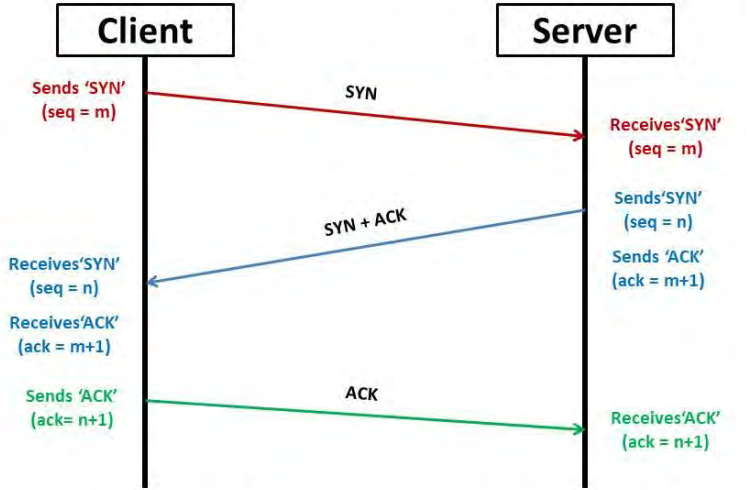




Concepts: Elastic Load Balancer and Target Group



Responding to the Load Balancer's Health Check Probes



- Suricata is listening from a NIC that is not bound to any TCP/IP stack, so we can't use the Linux TCP/IP stack
- R-Core implements the TCP 3-way handshake and uses DPDK transmit path (supports also ICMP probing)

Health-Check R-Core YAML Configuration

```
hcm_mode: 0                 # Reply to NLB health probes (0=disabled/default, 1=TCP/ICMP; 2=ICMP)
disable_auto_hcm: 0         # Whether or not to disable automated health changes (Optional)
hcm_preupgrade_wait: 60     # Seconds to wait after recovering health before reporting healthy
hcm_predowngrade_wait: 60 # Seconds to wait after losing health before reporting unhealthy
probe_tcp_port: 80          # Port on which to listen for health probes (Optional)
nlb_ip_address: 1.2.3.4     # IP address of the NLB from which to recognize probes (Optional)
propagate_probes: 0         # Propagate health probes/replies to analysis applications (Optional)
start_unhealthy: 0          # Ignore health probes until operator invokes set-healthy (Optional)
```

```
$ rcore-net-util -k amz1 -m get-stats

INFO,Application: eal_args: rcore --proc-type secondary -l 12   -w 08:00.0 -w 82:00.0
EAL: Detected 56 lcore(s)
EAL: Detected 2 NUMA nodes
EAL: Multi-process socket /var/run/dpdk/rte/mp_socket_7217_33a6315f9e9e2
EAL: Probing VFIO support...
EAL: VFIO support initialized
EAL: PCI device 0000:08:00.0 on NUMA socket 0
EAL:   probe driver: 8086:1572 net_i40e
EAL:   using IOMMU type 1 (Type 1)
EAL: PCI device 0000:82:00.0 on NUMA socket 1
EAL:   probe driver: 8086:1572 net_i40e

################################################################################
#                                RCORE STATUS                                  #
################################################################################
  started          2019-06-09 10:43:11 -0400
  updated          2019-06-09 19:04:06 -0400
  cores            0,1,2,14,15,3
  pid              45714
  mbuf-outages     0
  lcore-restarts   0
  probes-received  10
  probes-ackd      10
  foreign-probes   0
  man-hcm-changes  0
  auto-hcm-changes 0
  pkts-inspected   10

################################################################################
#                             APPLICATION METRICS                              #
# application      packets          bytes            packet-drops             #
################################################################################
  suri             22110989061      15345599095254   0

[clipped]
```

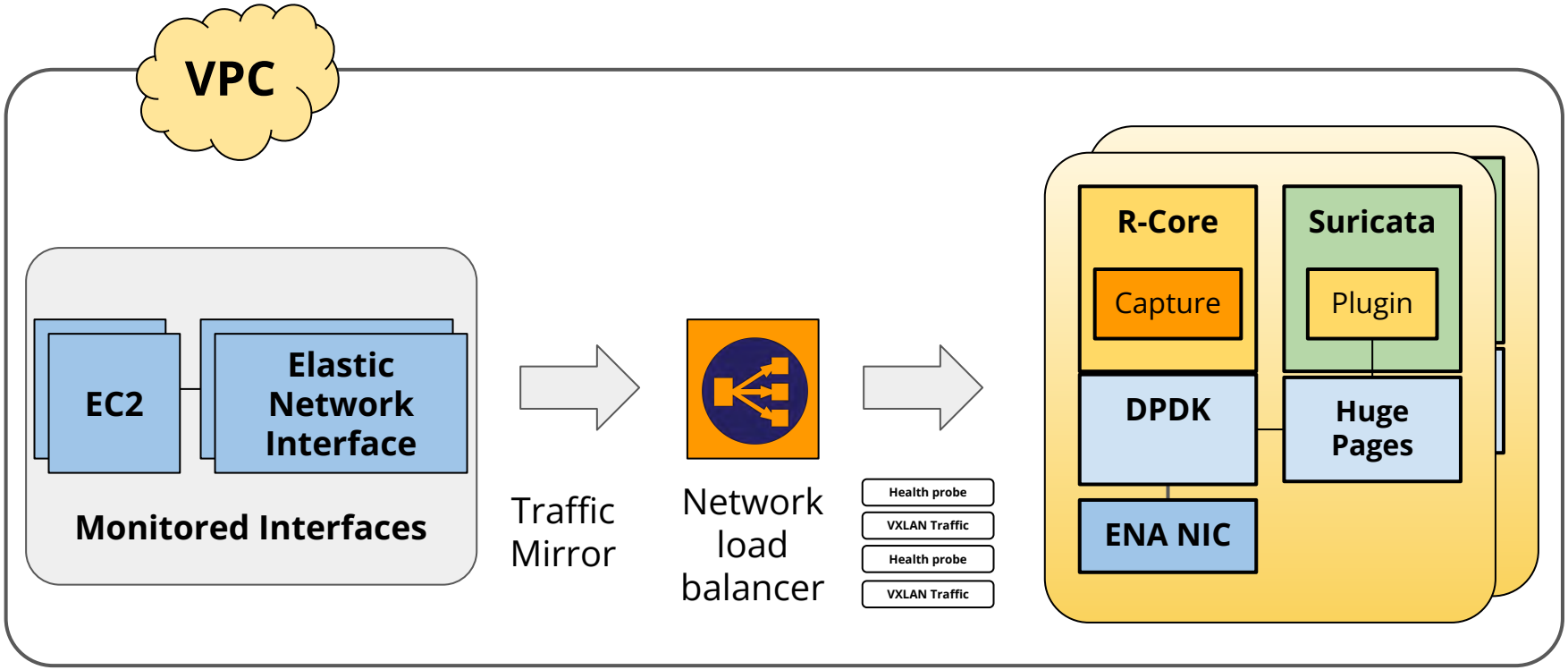
Approaches to Building a Suricata Deployment (AWS Case)

- AWS management console
- AWS command line interface
- AWS CloudFormation
- CDK

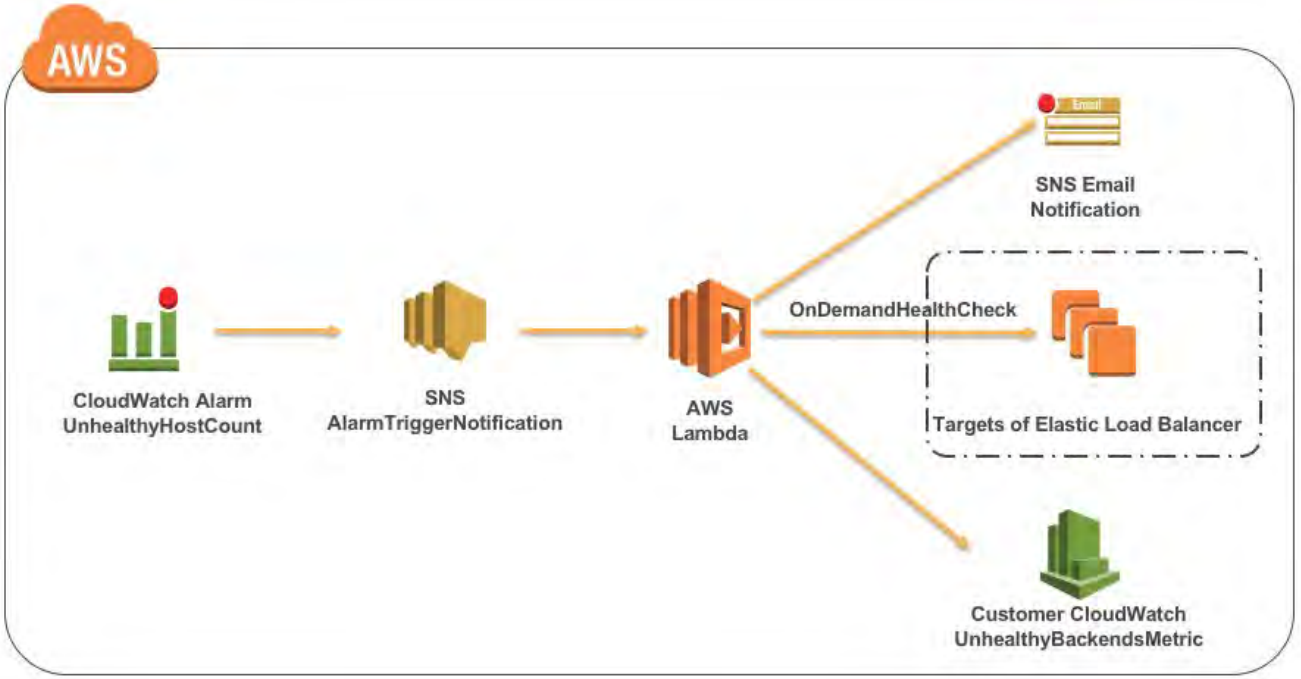
Deployment Steps in AWS (simplified)

- Start with a VPC having traffic to observe
- Create R-Core/Suricata instances (instantiate AMI)
- Create a network load balancer (NLB), a target group, and register targets (R-Core/Suricata)
- Create a traffic mirror session to capture the VPC interfaces of interest
- Create a traffic mirror target (designating the NLB as target)
- VXLAN-encapsulated mirrored traffic will now flow from Traffic Mirror to NLB to R-Core/Suricata
- Health Probes will also be sent from NLB, and answered by R-Core/Suricata instance

End-to-End Data Flow



Deployment Scaling



Q&A

Thank you

Reservoir Labs

632 Broadway 803, New York, NY 10012, United States