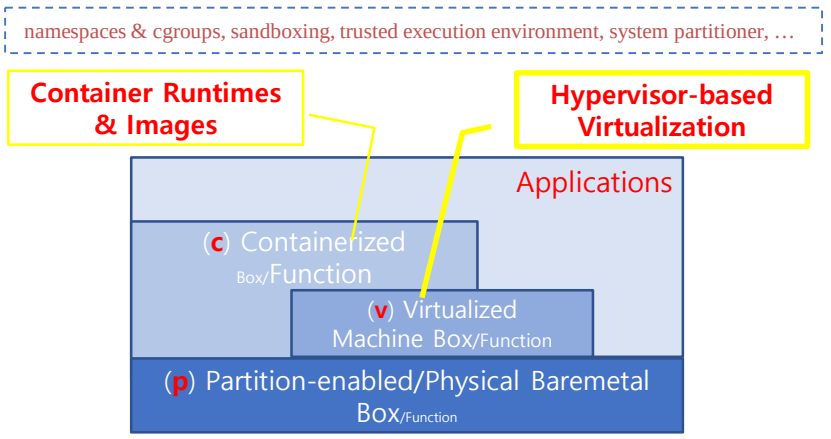
Open Networking Korea 2019 Fall

Cyber Security Defense Cycle

광주과학기술원 NetCS 연구실 – 최영은, 신준식

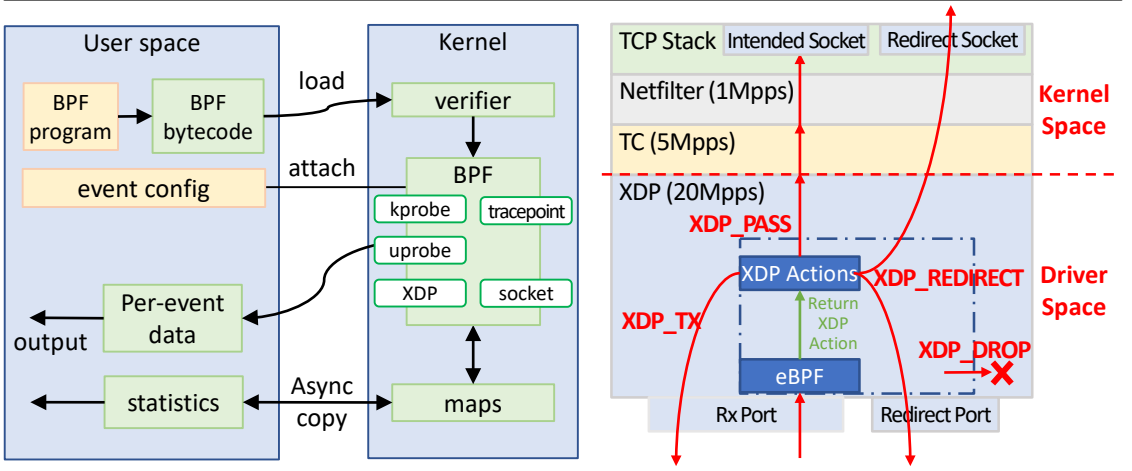
Conceptual Design of eBPF/XDP-based DoS Mitigation for Cloud-native Edge-cloud

P+v+c-harmonization in Cloud-native Edge-clouds



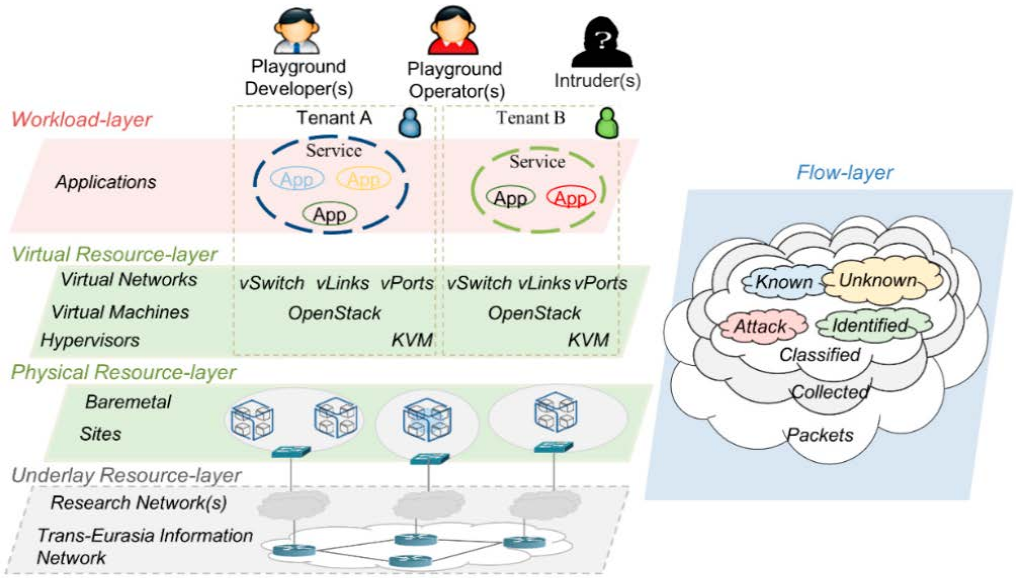
- Cloud-native computing? Containerized / Dynamically orchestrated / Microservices oriented
- Complicated Topology, Broad attack surface, and High-level App QoS in Cloud-native Edge-cloud

Extended Berkeley Packet Filter (eBPF) / Express Data Path (XDP)

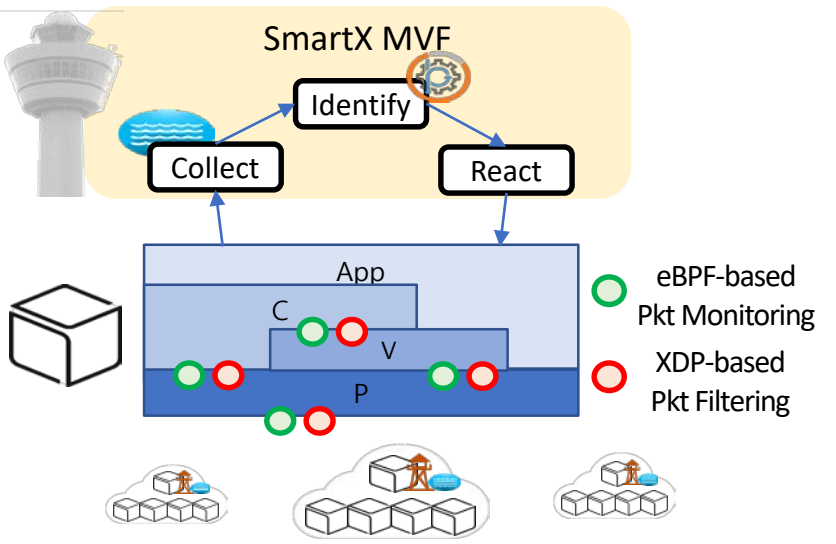


- eBPF enables Dynamic Kernel Code Injection
- eBPF verifier guarantees safe operation of the code in the kernel space
- eBPF/XDP Mode: Offload / Native / Generic

Security Extension of SmartX MultiView Visibility Framework (SmartX MVF)

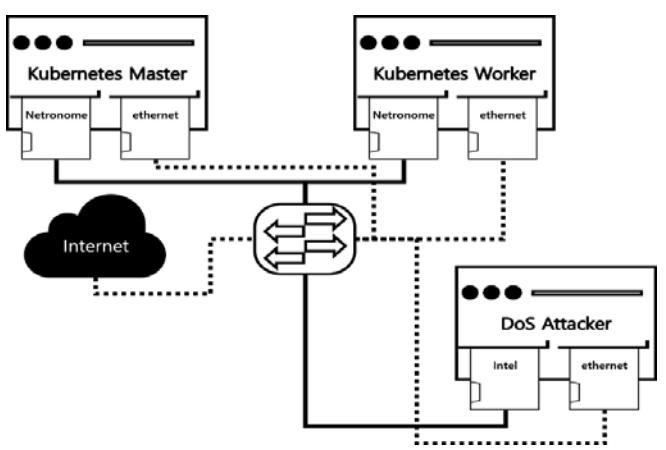


Multi-layered Visibility with an unified Onion-ring Visualization



Concept of eBPF/XDP-based Multi-Defense Security

Conceptual Verification of eBPF/XDP-based Packet Monitoring/Filtering for DoS Mitigation



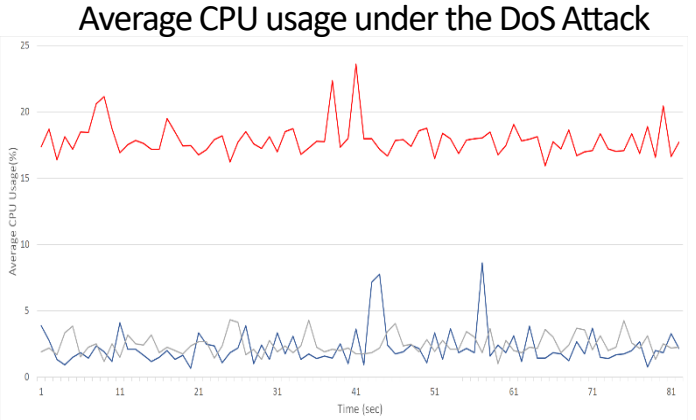
K-ONE Playground Configuration for Verification

- A Kubernetes-based Cloud-native Cluster (p+c)
- 10Gbps networking
- eBPF for Packet Monitoring
- Kafka for Data Transfer
- eBPF/XDP (HW offload mode) for Packet filtering

DoS Attack Simulation

- HTTP get/post flood attack on an IoT service on the cloud-native cluster
- Using Goldeneye, a DoS attack tool
- Traffic generated : 2270Pps & 1.3MBps

Average CPU usage under the DoS Attack



Onion-ring Visualization for DoS Attack

Color	The Incoming Packet Number Domain
Gray	0
Yellow Green	1 – 700
Light Green	701 – 1400
Green	1401 – 2100
Red	2101 - 2800

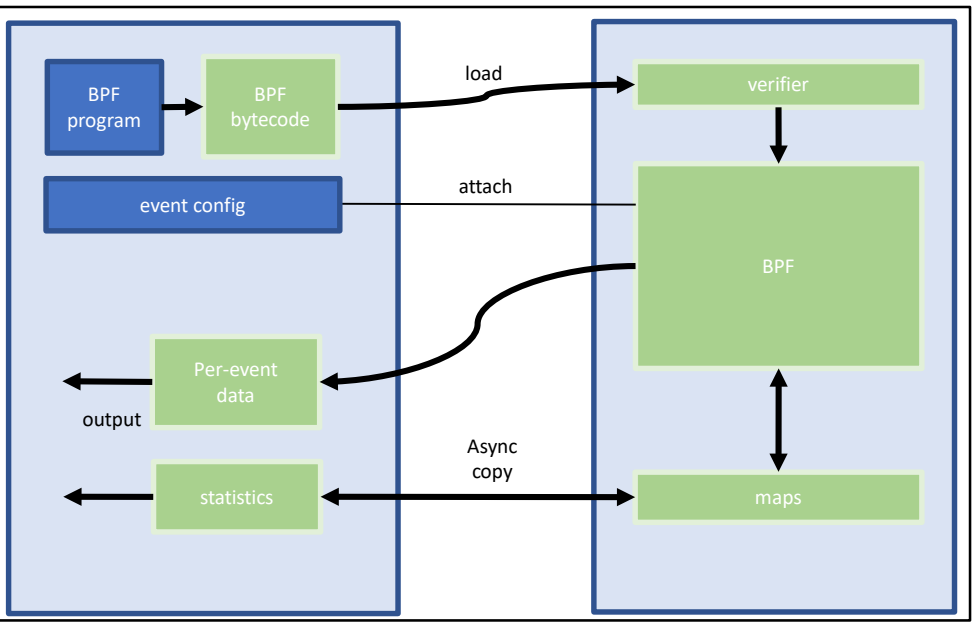
Open Networking Korea 2019 Fall

K-ONE GIST 김종원 교수님

A conceptual design of eBPF/XDP-based DoS mitigation for cloud-native edge-cloud

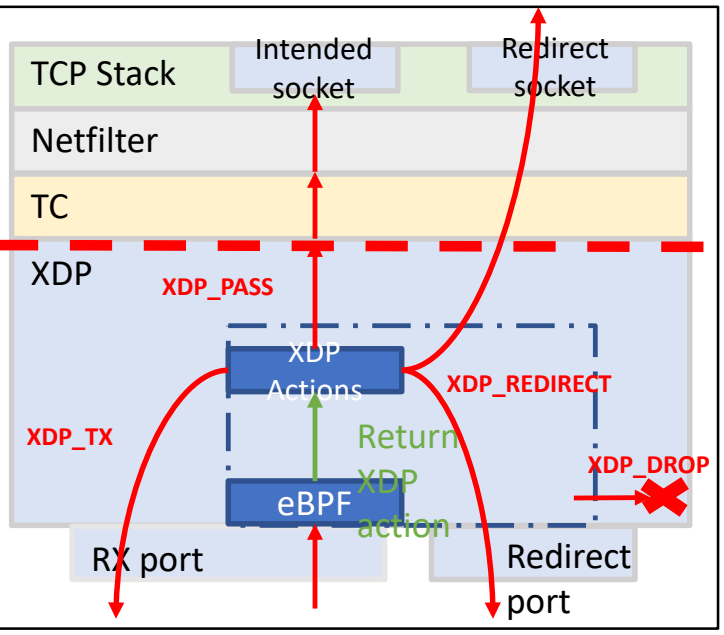
eBPF Based Packet Monitoring

- eBPF Attached to a Network Interface
  - Attach eBPF/XDP program to a network interface
- eBPF Packet Monitoring
  - Parse incoming packets in the kernel space
  - Pass information to the user space using eBPF maps
  - Parse IPv4 source address
  - Count incoming packet numbers
- Parsed Data transmission via Kafka
  - Monitored data is transmitted via Kafka to the visualization program



eBPF/XDP Based Packet Filtering

- eBPF/XDP Packet Filtering Using Hardware Offload Mode
  - eBPF/XDP hardware offload mode requires hardware support
  - Hardware offload does not leverage the host CPU
  - Use parsed information from the eBPF program
  - Checks if the incoming packet's source IPv4 address is in the blocklist
  - Packets dropped using XDP_DROP
  - If the NIC card does not support hardware offload mode, generic mode or native mode is leveraged



Onion-Ring Visualization

- Network Interface Visualization
  - Visualizes network interfaces attached to a physical server(black)
  - Network interface status visualized using different colors based on the incoming packet numbers

Color	The Incoming Packet Number Domain
Gray	0
Yellow Green	1 – 700
Light Green	701 – 1400
Green	1401 – 2100
Red	2101 - 2800

  - IPv4 source address of the incoming packets visualized when the mouse is plac on top of a network interface