Efficient data processing using eBPF

Team:

- Swaminathan Sriram
- Sreeram Ganesan
- Gopala Krishna Vasanth K

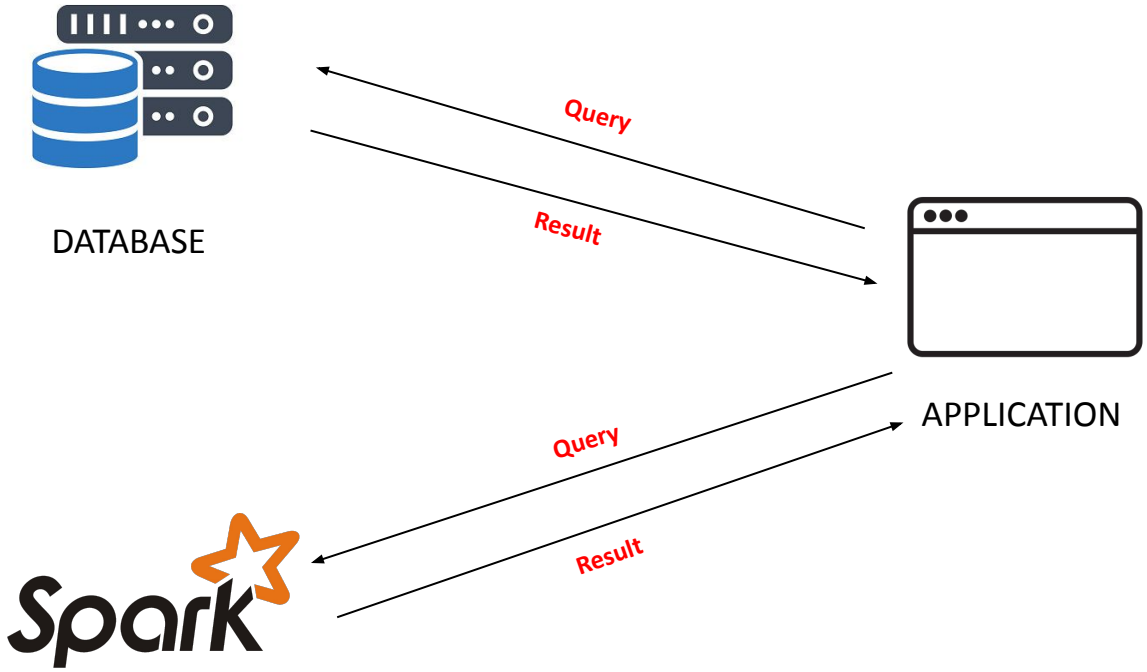
1987 -> 2021

BIG DATA

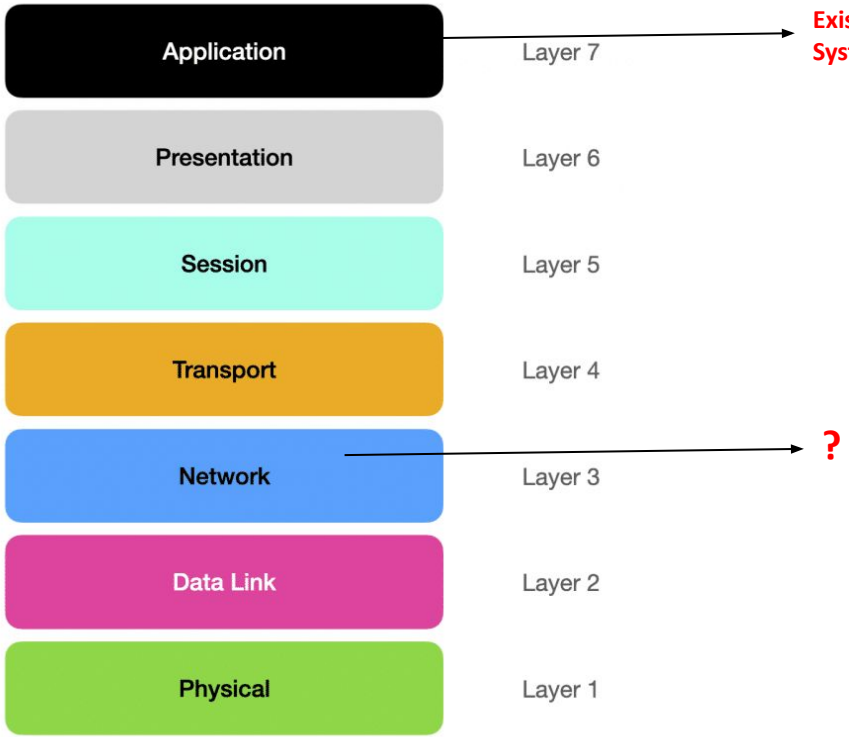
1 Zettabyte = 1 Million Exabytes

1 Exabyte= 1 Million Terabytes

Existing Data-Processing System



The OSI Model



Understanding eBPF & XDP

What is eBPF? (Apple Pen)

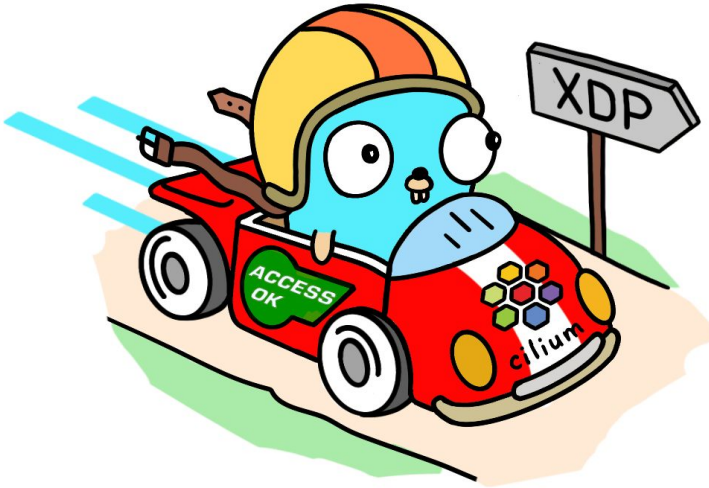
- Berkeley Packet Filter
- in-kernel execution engine to process virtual instruction sets

What is XDP? (Pineapple Pen)

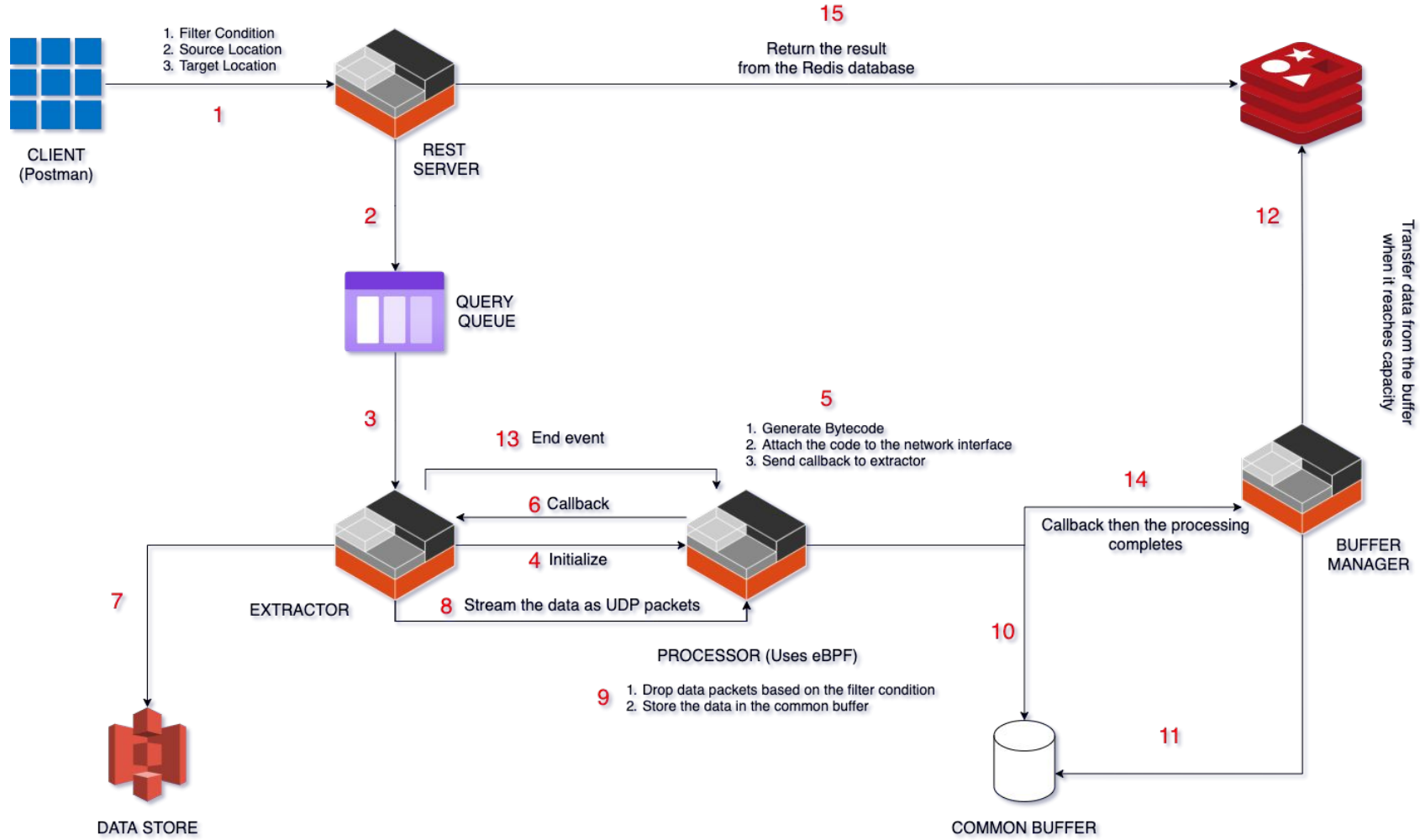
- Programmable network layer

What did we do? (Pen-Pineapple-Apple-Pen)

- Append user-filters to XDP
- Filter data packets at the network layer



Architecture Diagram


15
1. Filter Condition
2. Source Location
3. Target Location
1
Return the result
from the Redis database
CLIENT
(Postman)
REST
SERVER
2
12
Transfer data from the buffer
when it reaches capacity
QUERY
QUEUE
3
5
1. Generate Bytecode
2. Attach the code to the network interface
3. Send callback to extractor
13 End event
6 Callback
14
Callback then the processing
completes
BUFFER
MANAGER
4 Initialize
7
EXTRACTOR
8 Stream the data as UDP packets
10
PROCESSOR (Uses eBPF)
9
1. Drop data packets based on the filter condition
2. Store the data in the common buffer
11
DATA STORE
COMMON BUFFER

Components

- Kubernetes - service deployment and scaling
- eBPF - packet filtering
- rabbitMQ - messaging queue
- Flask REST server - API interface
- Redis - data store

Debugging

- BPF helpers
- K8s system logs
- rabbitMQ for gathering logs

Learnings

- eBPF & XDP - Network Programming
- Kubernetes - Microservice Orchestration

Future Scope

- Handling common memory buffer
- Handling TCP packets

Thank You!

Presentation Outline

- Project Goals
- Software and hardware components
- Architectural diagram – description of interactions
- How did you debug?
- What did you learn and what would you do differently?