Dynamic Network Traffic Monitoring and Scaling Using eBPF

Team Members:

Faiaz Halim (202383578)
Anil Manyam (202483873)
S. M. Yeamin Oni (202293272)

Problem Statement

Traditional monitoring lacks flexibility for microservices

Need for real-time traffic-based scaling

Complex network policy management

Solution

eBPF-based network monitoring

Multi-metric scaling (traffic, CPU, memory)

Policy-driven traffic control

Real-time visualization

Architecture

Components

1. **Core Infrastructure**
   - Kubernetes 1.31 cluster (kind)
   - Cilium 1.16.3 with eBPF
   - Hubble for flow visibility

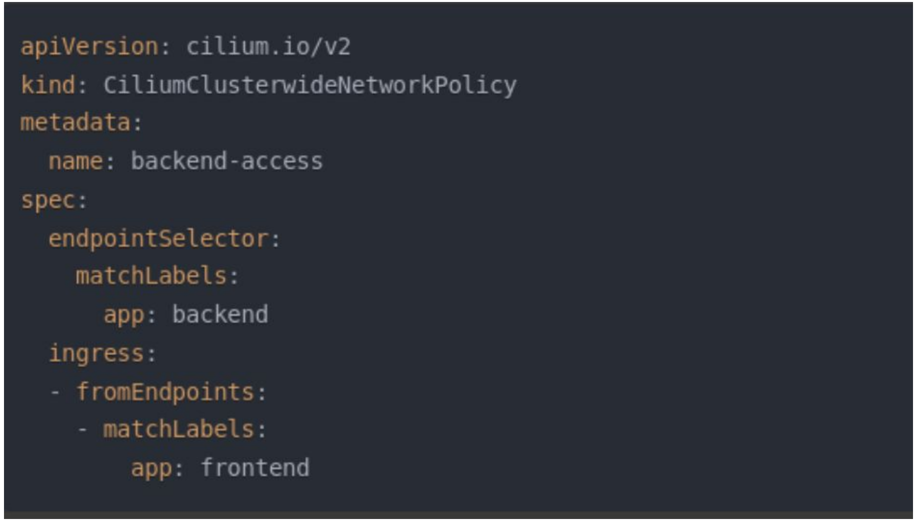
2. **Monitoring Stack**
   - Prometheus for metrics storage
   - Custom Prometheus Adapter
   - Hubble UI and Grafana for visualization

3. **Application Layer**
   - Frontend service
   - Backend service
   - Load testing suite

Implementation

1. Network Monitoring

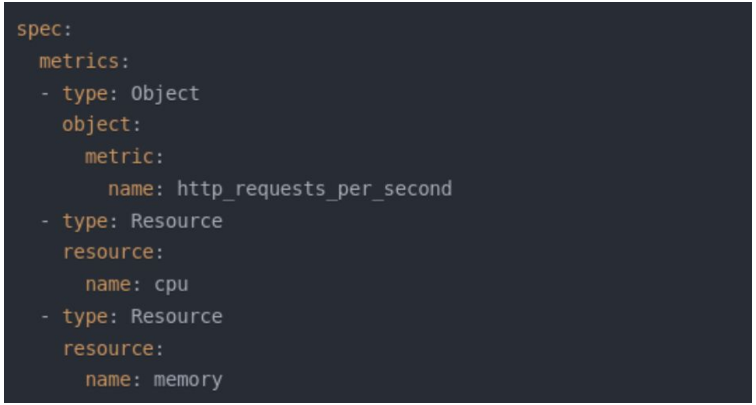
- Cilium eBPF programs for packet inspection
- Hubble flow monitoring



```yaml
apiVersion: cilium.io/v2
kind: CiliumClusterwideNetworkPolicy
metadata:
  name: backend-access
spec:
  endpointSelector:
    matchLabels:
      app: backend
  ingress:
  - fromEndpoints:
    - matchLabels:
        app: frontend
```

2. Dynamic Scaling

- Multiple HPA metrics combined:



```
spec:
  metrics:
  - type: Object
    object:
      metric:
        name: http_requests_per_second
  - type: Resource
    resource:
      name: cpu
  - type: Resource
    resource:
      name: memory
```

3. Load Testing Suite

- Traffic generation
- CPU stress testing
- Real-time monitoring
- Automated reporting

Features

Monitoring

- Network flows
- Resource utilization
- Policy enforcement

Scaling

- Traffic-based scaling
- Resource-based scaling
- Combined metric approach

Security and Control

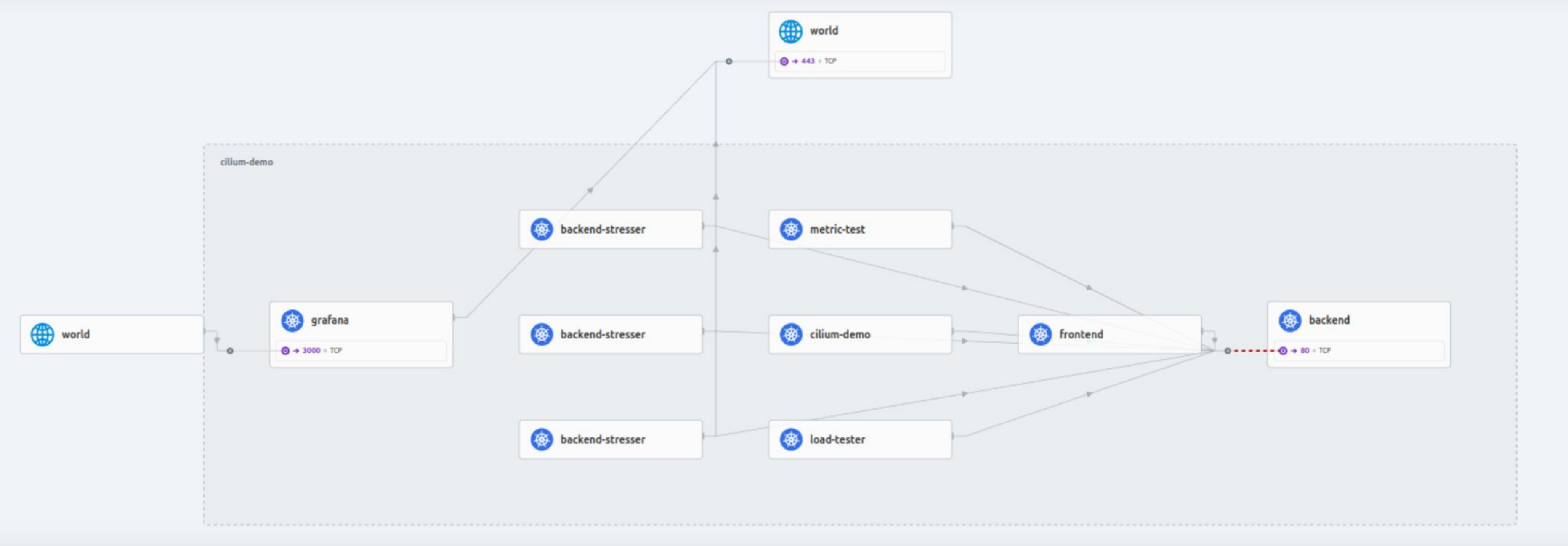
- L3/L4/L7 policies
- Rate limiting
- Flow visibility

Testing and Validation

- Automated load testing
- Multi-metric validation
- Comprehensive reporting

Results

- Hubble UI
- Traffic Monitoring Report



Future Enhancements

Advanced Analytics

- Machine learning for predictive scaling
- Pattern recognition
- Anomaly detection

Enhanced Policies

- Dynamic policy adjustment
- Context-aware rules
- Advanced rate limiting

Optimization

- Performance tuning
- Resource efficiency
- Scaling algorithms

Questions?

References

- S. Magnani, F. Risso, D. Siracusa, “*A Control Plane Enabling Automated and Fully Adaptive Network Traffic Monitoring With eBPF*”, IEEE Access (Volume: 10) (https://ieeexplore.ieee.org/document/9869628), 2022.
- JB. Lee, TH. Yoo, EH. Lee, BH. Hwang, SW. Ahn, CH. Cho, “*High-Performance Software Load Balancer for Cloud-Native Architecture*”, IEEE Access (Volume: 9) (https://ieeexplore.ieee.org/document/9524915), 2021.
- S. Miano, F. Risso, M. V. Bernal, M. Bertrone, Y. Lu, “*A Framework for eBPF-Based Network Functions in an Era of Microservices*”, IEEE Transactions on Network and Service Management (Volume: 18, Issue: 1) (https://ieeexplore.ieee.org/document/9340283), 2021.
- H. Sharaf, I. Ahmad, T. Dimitriou, “*Extended Berkeley Packet Filter: An Application Perspective*”, IEEE Access (Volume: 10) (https://ieeexplore.ieee.org/document/9968265), 2022.
- M. Jadin, Q. D. Coninck, L. Navarre, M. Schapira, O. Bonaventure, “*Leveraging eBPF to Make TCP Path-Aware*”, IEEE Transactions on Network and Service Management (Volume: 19, Issue: 3) (https://ieeexplore.ieee.org/document/9772044), 2022.