Kubernetes 혼자구축해보기

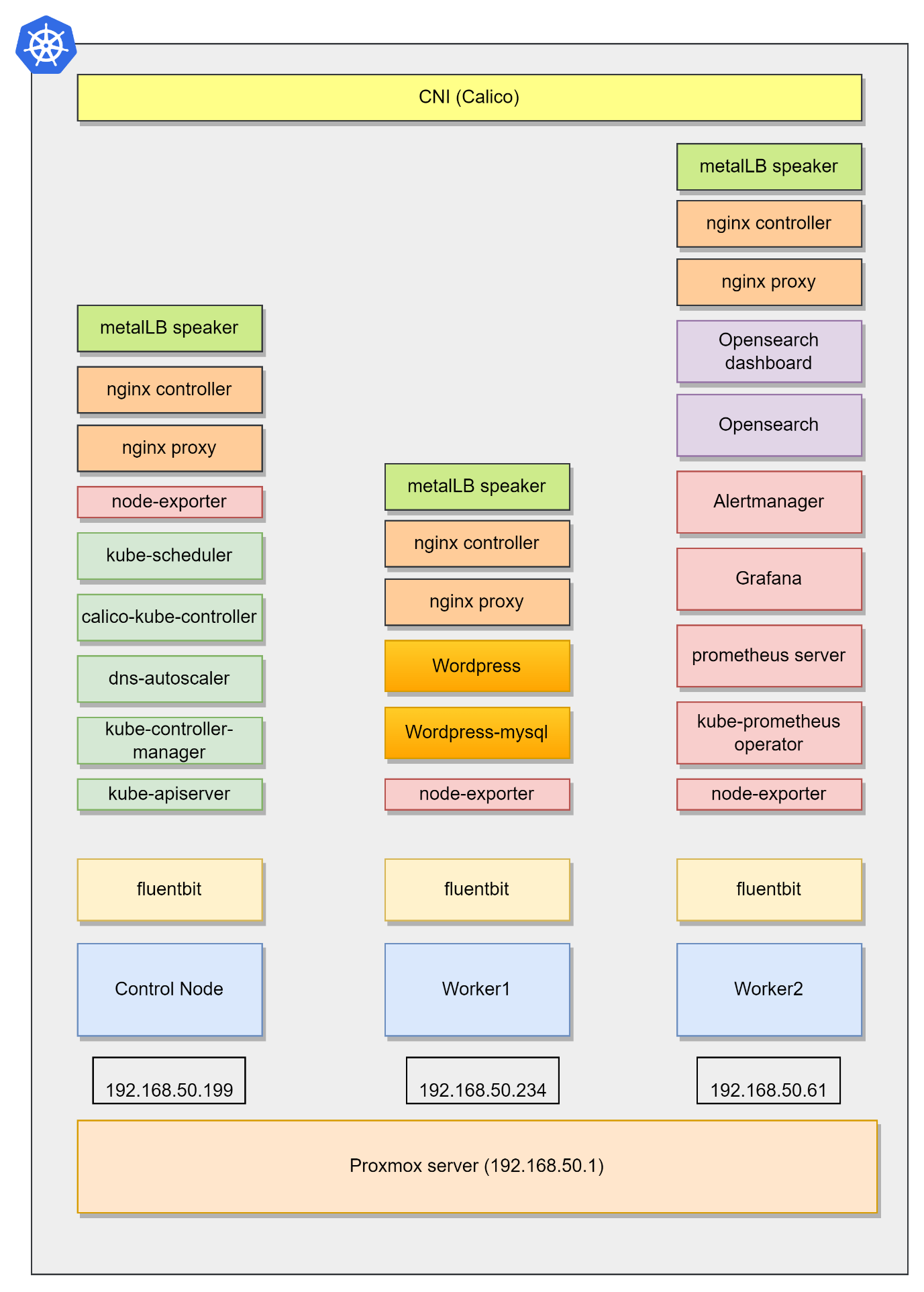
최상원

1. Kubernetes Cluster 구성 정보

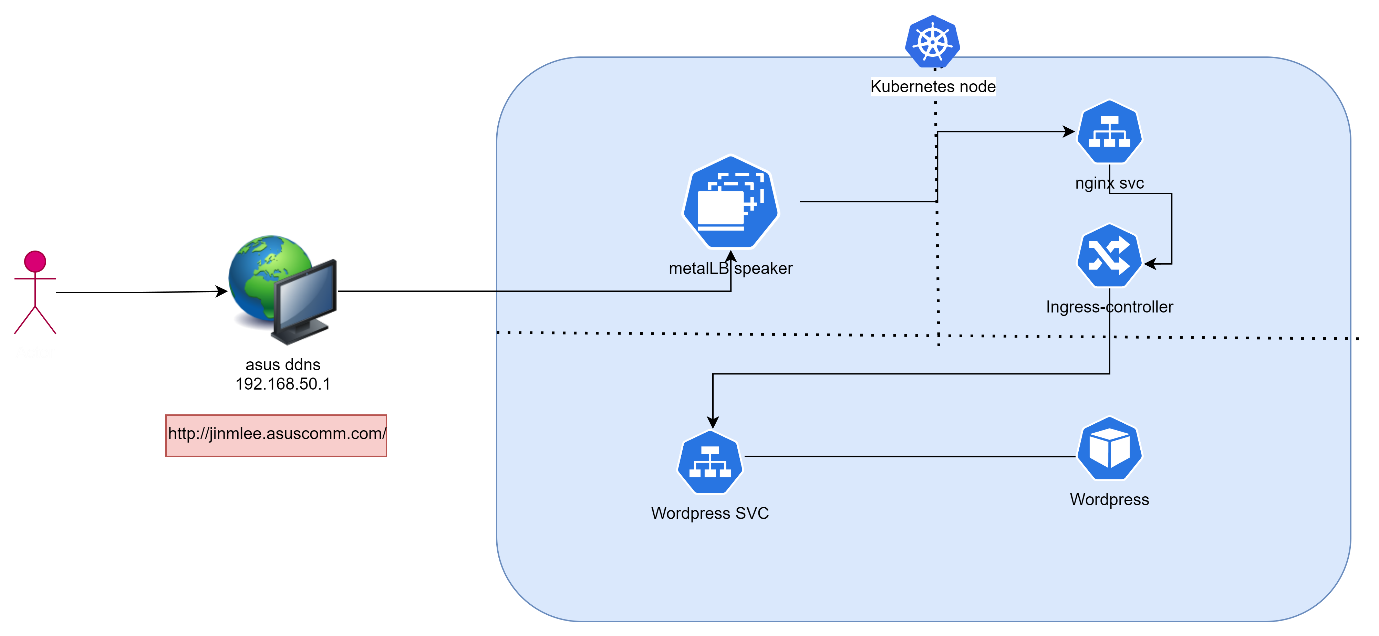
|  |  |  |  |
| --- | --- | --- | --- |
| Server name | IP Address | Port | account |
| master | 192.168.50.199 | 22 | Root/sangwon |
| Worker1 | 192.168.50.234 | 22 | Root/sangwon |
| Worker2 | 192.168.50.61 | 22 | Root/sangwon |

1. Kubernetes 물리 구성

|  |  |  |  |
| --- | --- | --- | --- |
| Node | OS | CPU | Memory |
| Master | Ubuntu 20.04.6 LTS | 4core | 4096MiB |
| Worker1 | Ubuntu 20.04.6 LTS | 4core | 4096MiB |
| Worker2 | Ubuntu 20.04.6 LTS | 4core | 4096MiB |

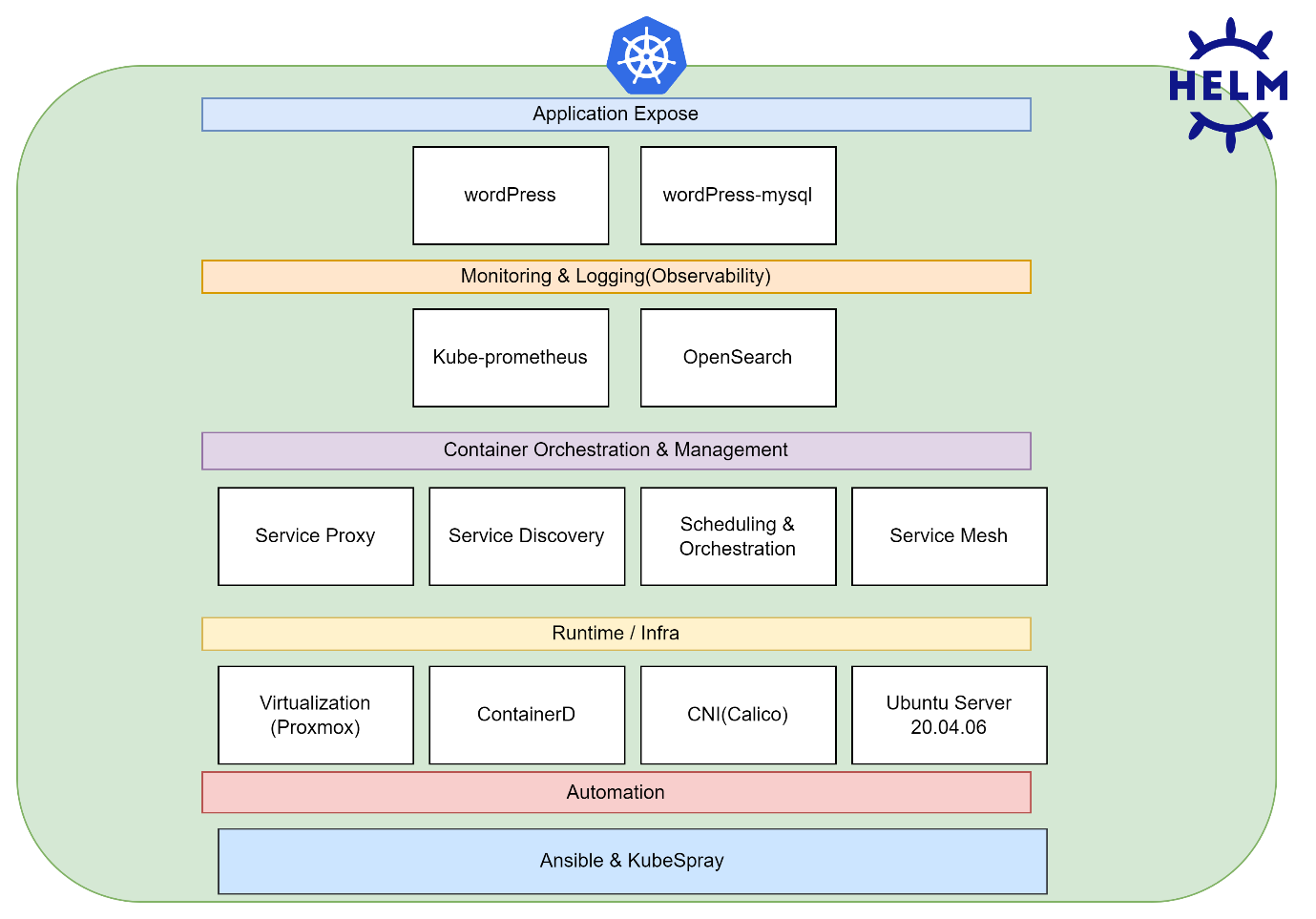


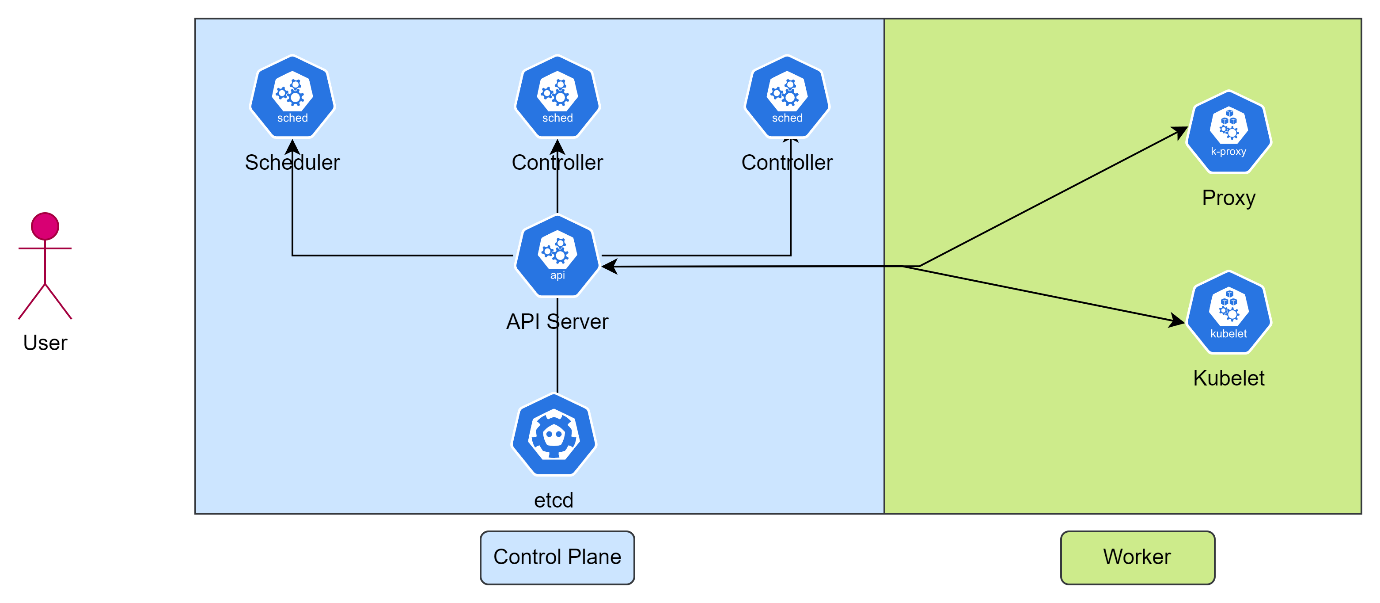
* Kube-spray와 ansible을 이용해 Kubernetes v1.26 환경을 각 노드에 동일하게 배포하였습니다.
* Control node에는 node들을 manage와 control해주는pod들을 배치, worker 1에는 웹사이트를 배포하는 wordpress와 DB인 mysql, worker2에는 모니터링을 담당하는 Kube-prometheus stack과 로깅을 담당하는 Opensearch Stack을 배치하였습니다.
* Public Cloud 환경이 아닌 VM으로 Cluster를 구축했기에 metalLB와 nginx의 reverse proxy로 loadbalancer를 구축하였습니다.



* Wordpress svc를 worker1에서 배포, L7의 nginx를 지나 L4의 metalLB가 내 proxmox server의 공유기인 asus에 도달하게 되고 ddns 서비스를 통해 domain name을 가지고 호스팅하였습니다.

1. Kubernetes 논리 구성





A. Kubernetes의 목적인 Desired State를 구축하기 위해 Control Plane에서 Node별로 할당하고 상태를 체크하는 Controller와 Scheduler를 배포하고 그것을 질의하는 API Server를 두었습니다. Worker의 Proxy와 Kubelet은 API server를 바라보게 만들어 Pod 및 Node상태를 실행, 삭제, 생성하고 상태를 체크하게 구성하였습니다.

B. 동일한 환경 및 쉬운 배포를 하기 위해 Ansible & Kubespray를 사용하였습니다. Inventory를 만들어 ssh 통신만 된다면 Cluster를 아주 쉽게 만들수 있었습니다.

C. Kubernetes에서 핵심인 Orchestration의 역할과 Scheduling, Management 기능들을 구성하였습니다.

D. 제가 구성한 기능들이 Resource가 충분한지, Deployment와 Statefulset, Daemonset들이 멈추지 않는지 혹은 멈춘다면 어떻게 대처하고 로그를 볼 것인지에 대한 기능을 구성하였습니다.

E. VM으로 클러스터를 구현하였고 이를 공유기로 expose하기 위해서 공유기의 ddns라는 기능을 통해 내 컴퓨터의 IP를 연결함과 동시에 domain name을 부여하였습니다.

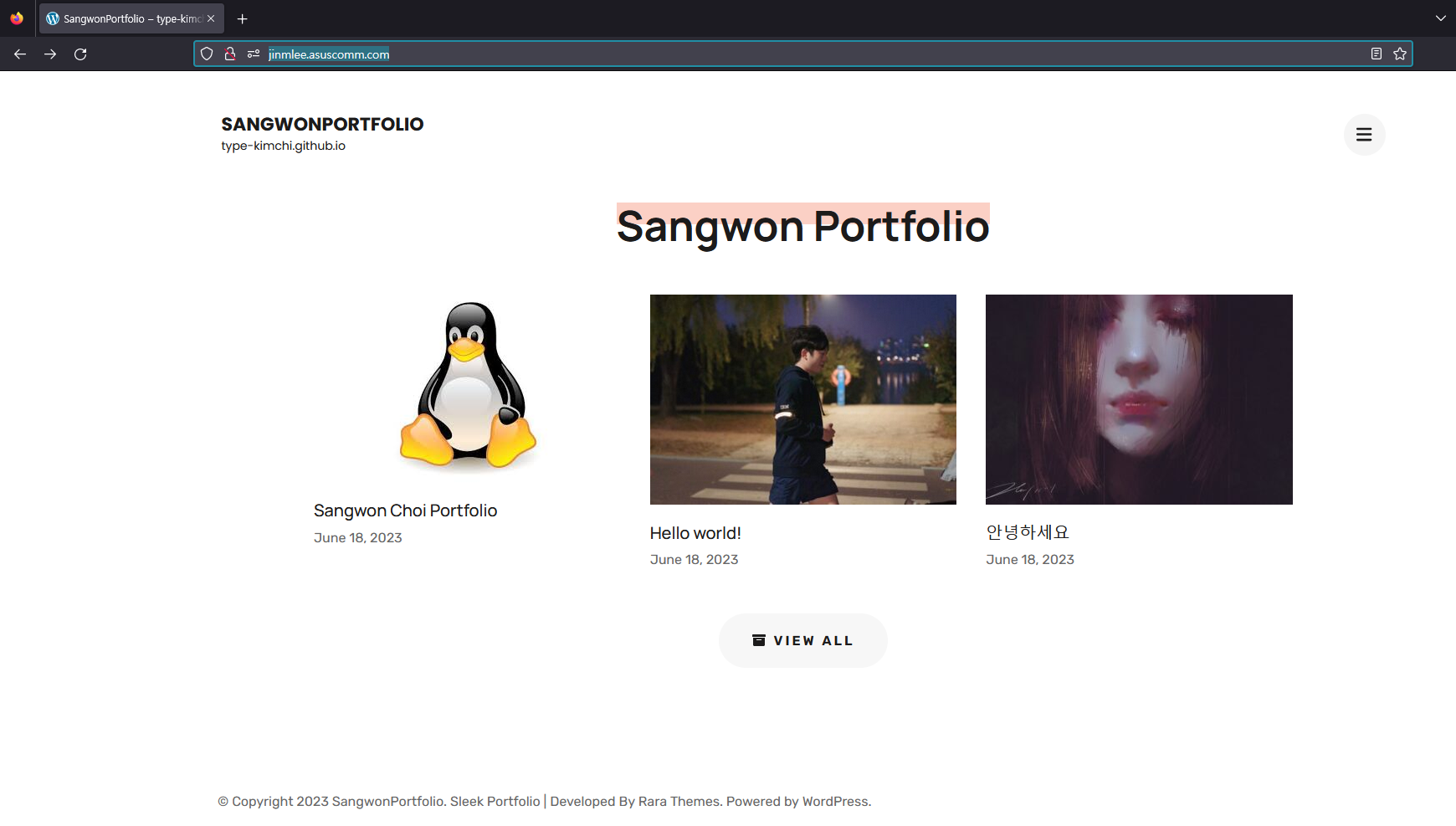
F. Helm Chart를 통해 패키지화 된 image를 value의 customize를 통해 환경에 맞게 배포하였습니다.

3. 서비스 구축가능 여부 확인

Web url : <http://jinmlee.asuscomm.com/>

(firefox 브라우져 추천)

첨부되어 있는 Vpn을 사용하면 모니터링과 로깅 노드에도 접근할 수 있습니다.

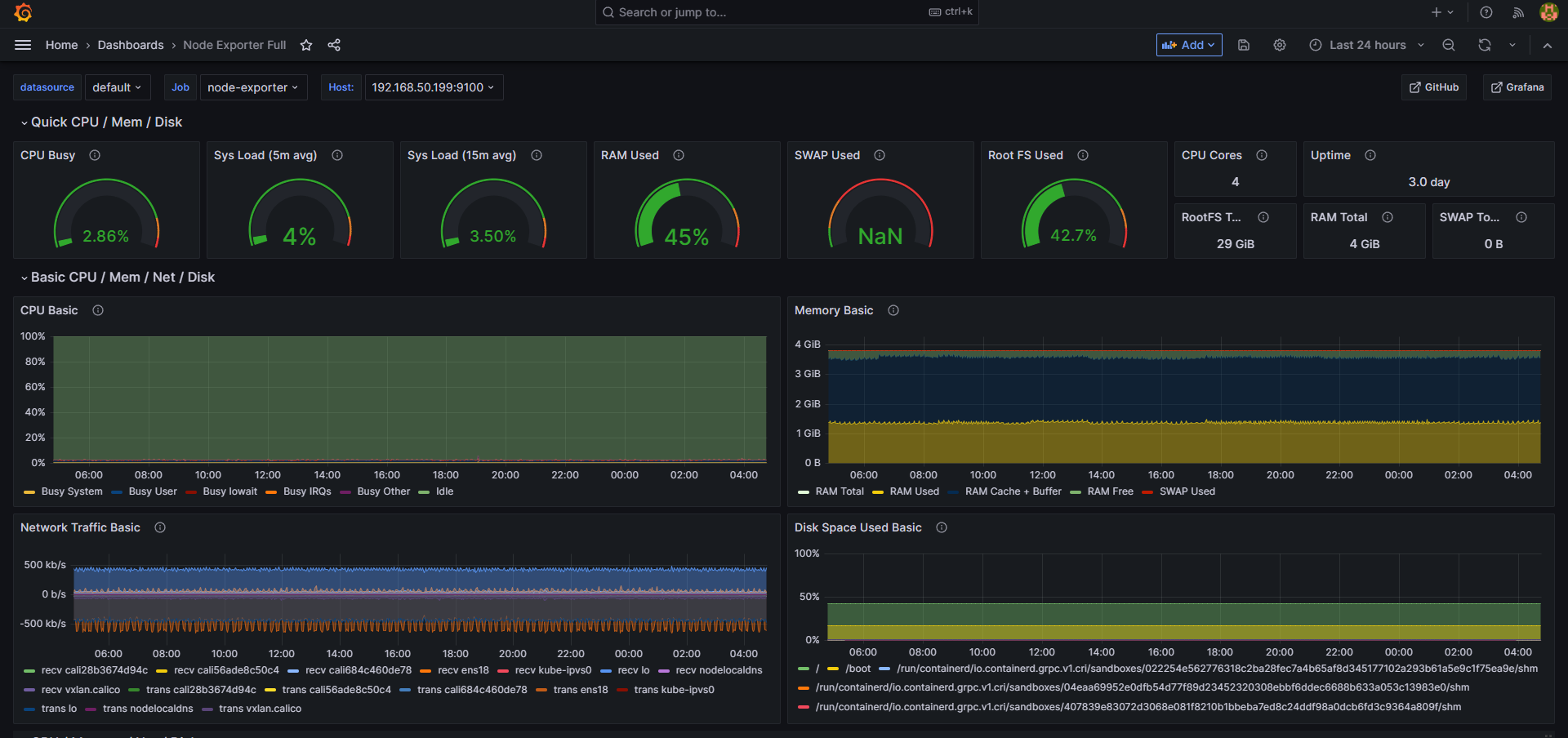


첫번째 게시글인 Sangwon Choi portfolio에 정리를 해놓았습니다.

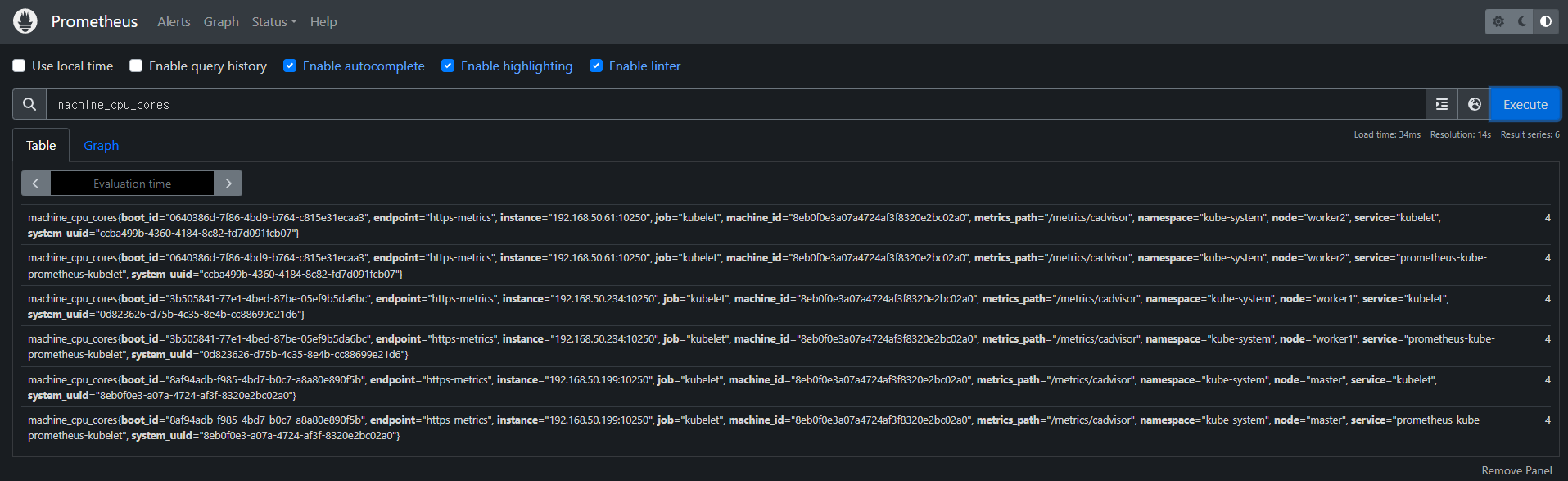
Grafana url : <http://192.168.50.234:32668/>

* 그라파나의 경우는 node exporter로 node의 cpu, disk usage, network, memory 사용량등을 시각적으로 볼 수 있게 설정했습니다.

Id / PW : admin/P@ssw0rd



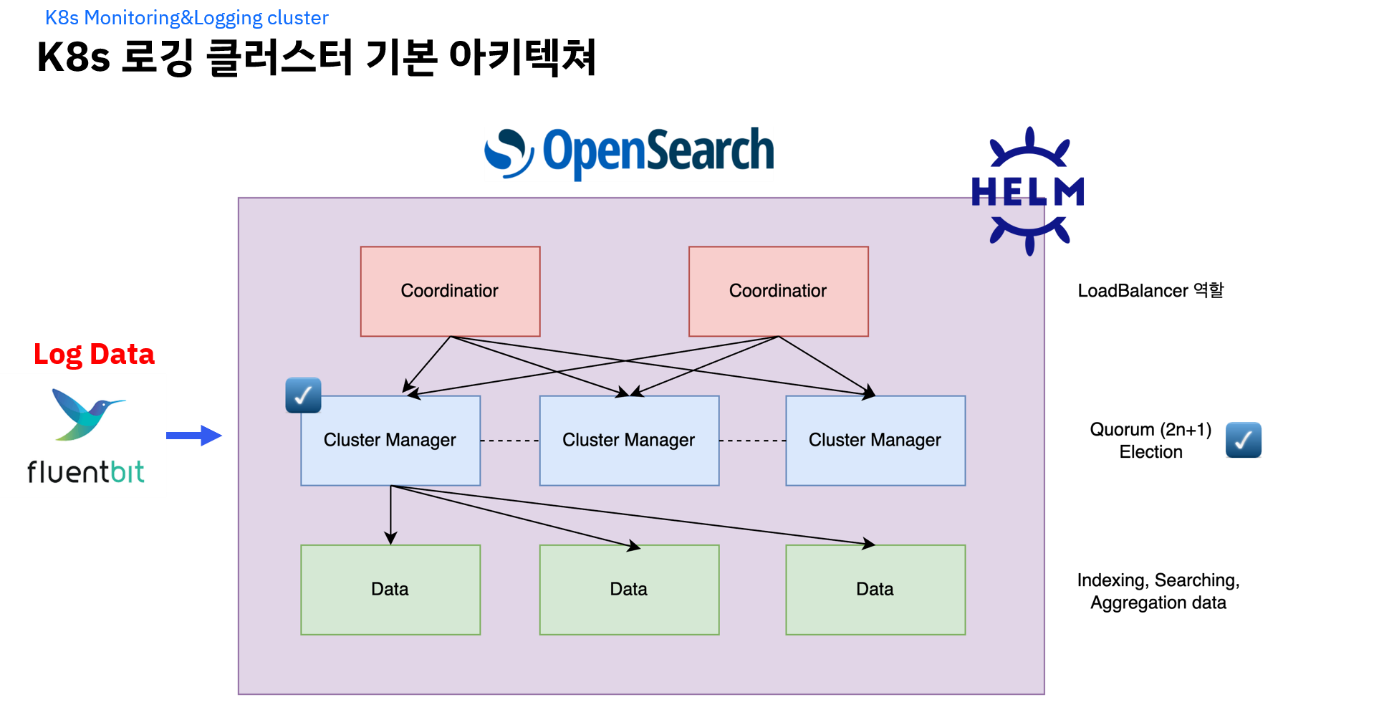
Prometheus url : http://192.168.50.61:30648/

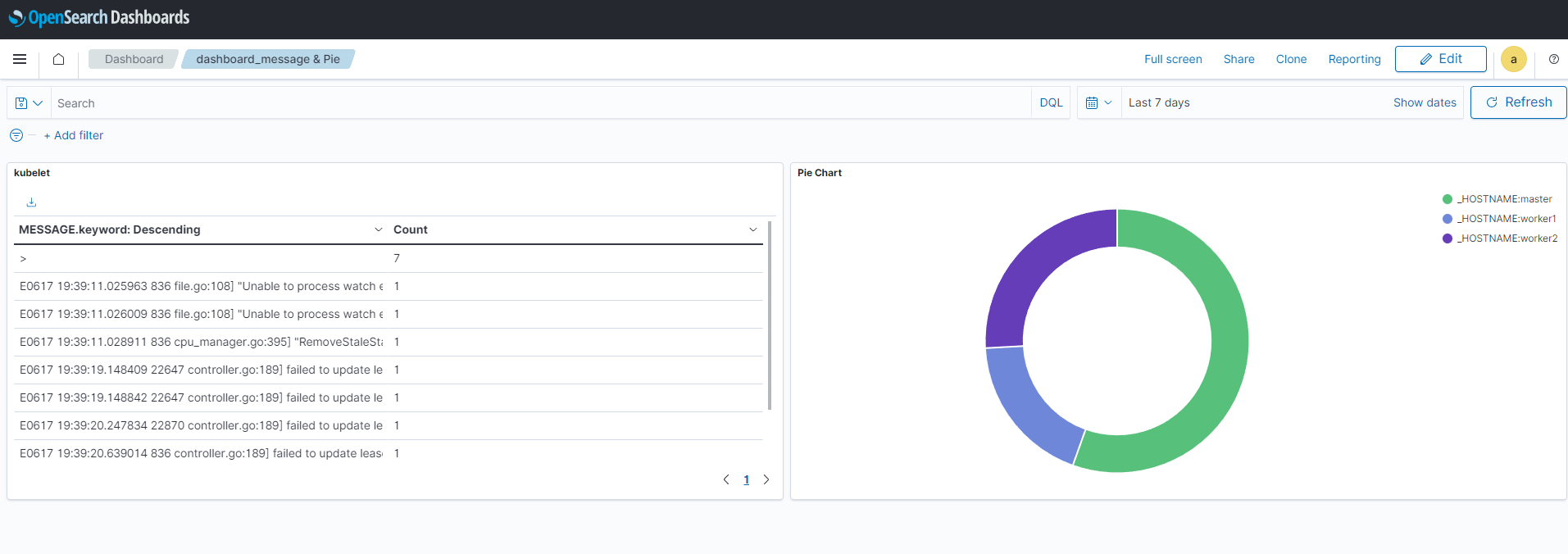


OpenSearch : <http://192.168.50.61:32762/>

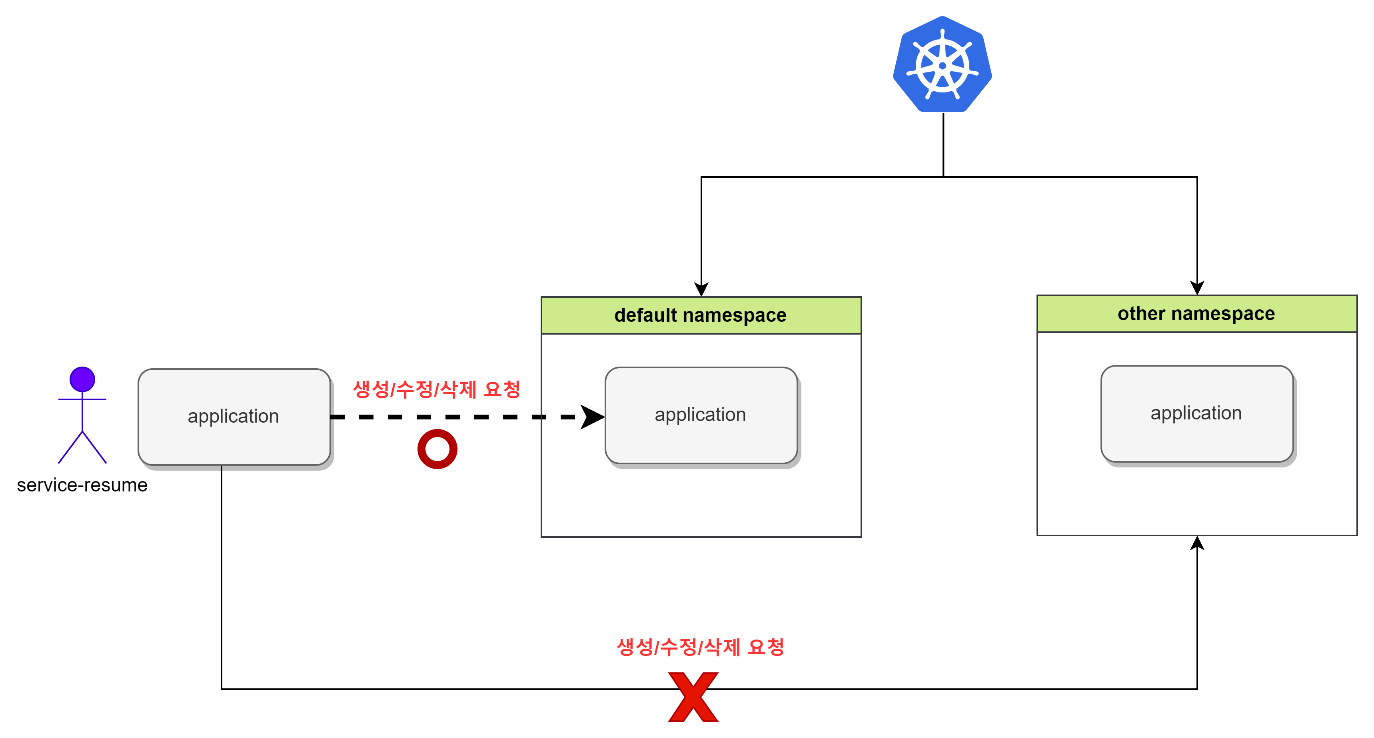
* OpenSearch의 경우에는 fluentbit에서 로그를 가져오도록 설정하였습니다. 간단하게 메시지박스와 piechart로 구성하였습니다.

ID / PW : admin / admin





4. 구축된 service account



- service-resume라는 service account는 default namespace의 pod들에 secret이 mount되어 있습니다.

- 이 service account는 ["create", "delete", "get", "watch", "list"]의 기능을 default namespace에만 구현을 해놓았기에 사용자가 다른 namespace에 접근하는 것을 사전차단 하였습니다.

5. 자신이 구축한 시스템의 특장점과 장애요소를 기술하시오.

특장점

1. Kubespray를 활용하였으므로 클러스터의 규모가 늘어난다 하더라도 쉽게 노드를 추가할 수 있습니다.
2. Monitoring환경을 구성하여 내 시스템의 resource를 효율적으로 관리, 감독할 수 있습니다.
3. Logging 환경을 구성하여 fluentbit이 Container와 Kubelet의 Log를 가져와 문제를 분석하고 장애 및 보안책임을 트래킹 할 수 있습니다.
4. metalLB의 Loadbalancer와 nginx의 reverse proxy로 on-premise 환경에서도 효과적으로 서비스를 expose할 수 있습니다.

장애요소

1. 백업 및 복구 요소를 고려하지 않았기에 고가용성이 부족할 수 있습니다.
2. 보안요소를 고려하지 않았기에 보안에 취약할 수 있습니다. (이미지 취약성 및 보안 컴플라이언스, 통합 보안 관제)
3. 어플리케이션에 mount된 storage를 local storage로 사용했기에 data loss 및 장기보관이 어려울 수 있습니다.
4. 모니터링 metric 및 logging data의 Retention date를 15일로 설정했기에 Long-term 이슈 및 추이 분석이 어려울 수 있습니다.

발전 가능성

1. Wordpress 기반의 블로그를 구성하였지만 추후 resource 및 Plugin에 대한 이해도를 바탕으로 게시판 플랫폼 및 다양한 커뮤니티로의 발전이 가능합니다.
2. 다양한 OpenSource를 활용해 보안, 백업등을 integration 시켜 아키텍쳐를 발전시킬 수 있습니다.