고급 구성 옵션

NetApp Solutions

NetApp
September 26, 2024

목차

고급 구성 옵션

로드 밸런서 옵션 탐색

로드 밸런싱 장치 옵션 알아보기: NetApp의 Red Hat OpenShift

대부분의 경우 Red Hat OpenShift는 경로를 통해 외부 세계에 애플리케이션을 제공합니다. 외부에서 연결할 수 있는 호스트 이름을 제공하여 서비스가 노출됩니다. 정의된 라우트와 서비스로 식별되는 엔드포인트는 OpenShift 라우터에서 외부 클라이언트에 명명된 연결을 제공하기 위해 사용될 수 있습니다.

그러나 경우에 따라 적절한 서비스를 제공하기 위해 응용 프로그램에서 사용자 지정 로드 밸런싱 장치를 구축하고 구성해야 하는 경우도 있습니다. 한 가지 예로는 NetApp Astra Control Center가 있습니다. 이러한 요구 사항을 충족하기 위해 다양한 사용자 지정 로드 밸런서 옵션을 평가했습니다. 설치 및 구성에 대한 자세한 내용은 이 섹션을 참조하십시오.

다음 페이지에서는 NetApp OpenShift에서 검증된 로드 밸런서 옵션에 대한 추가 정보를 제공합니다.

- "메탈리스"
- "F5 BIG-IP"

MetalLB 로드 밸런서 설치: NetApp과 Red Hat OpenShift

이 페이지에는 MetalLB 로드 밸런서에 대한 설치 및 구성 지침이 나와 있습니다.

MetalLB는 OpenShift 클러스터에 설치되는 자체 호스팅 네트워크 로드 밸런서로서, 클라우드 공급자에서 실행되지 않는 클러스터에서 유형 로드 밸런서의 OpenShift 서비스를 생성할 수 있습니다. 로드 밸런서 서비스를 지원하기 위해 함께 작동하는 MetalLB의 두 가지 주요 기능은 주소 할당과 외부 안내입니다.

MetalLB 구성 옵션

MetalLB가 OpenShift 클러스터 외부의 로드 밸런서 서비스에 할당된 IP 주소를 알려 주면 두 가지 모드로 작동합니다.

- * Layer 2 모드 * 이 모드에서는 OpenShift 클러스터의 한 노드가 서비스 소유권을 가져와 해당 IP에 대한 ARP 요청에 응답하여 OpenShift 클러스터 외부에서 해당 IP에 연결할 수 있도록 합니다. 노드만 IP를 광고하기 때문에 대역폭 병목 현상 및 느린 페일오버 제한이 있습니다. 자세한 내용은 설명서를 참조하십시오 "여기".
- * BGP mode. * 이 모드에서 OpenShift 클러스터의 모든 노드는 라우터와 BGP 피어링 세션을 설정하고 트래픽을 서비스 IP로 전달하기 위한 경로를 광고합니다. 이를 위해서는 MetalLB를 해당 네트워크의 라우터에 통합해야 합니다. BGP의 해싱 메커니즘으로 인해 서비스에 대한 IP-노드 매핑이 변경될 때 특정 제한이 있습니다. 자세한 내용은 설명서를 참조하십시오 "여기".

ⓘ 이 문서에서는 MetalLB를 Layer-2 모드로 구성합니다.

MetalLB 로드 밸런서 설치

1. MetalLB 리소스를 다운로드합니다.

```
[netapp-user@rhel7 ~]$ wget
https://raw.githubusercontent.com/metallb/metallb/v0.10.2/manifests/name
space.yaml
[netapp-user@rhel7 ~]$ wget
https://raw.githubusercontent.com/metallb/metallb/v0.10.2/manifests/meta
llb.yaml
```

2. 파일 Metallb.YAML을 편집하고 컨트롤러 배포 및 스피커 DemonSet에서 pec.template.spec.securityContext` 파일을 제거합니다.

   삭제할 줄: *

```
securityContext:
  runAsNonRoot: true
  runAsUser: 65534
```

3. 'metallb-system' 네임스페이스를 만듭니다.

```
[netapp-user@rhel7 ~]$ oc create -f namespace.yaml
namespace/metallb-system created
```

4. MetalLB CR을 만듭니다.

```
[netapp-user@rhel7 ~]$ oc create -f metallb.yaml
podsecuritypolicy.policy/controller created
podsecuritypolicy.policy/speaker created
serviceaccount/controller created
serviceaccount/speaker created
clusterrole.rbac.authorization.k8s.io/metallb-system:controller created
clusterrole.rbac.authorization.k8s.io/metallb-system:speaker created
role.rbac.authorization.k8s.io/config-watcher created
role.rbac.authorization.k8s.io/pod-lister created
role.rbac.authorization.k8s.io/controller created
clusterrolebinding.rbac.authorization.k8s.io/metallb-system:controller
created
clusterrolebinding.rbac.authorization.k8s.io/metallb-system:speaker
created
rolebinding.rbac.authorization.k8s.io/config-watcher created
rolebinding.rbac.authorization.k8s.io/pod-lister created
rolebinding.rbac.authorization.k8s.io/controller created
daemonset.apps/speaker created
deployment.apps/controller created
```

5. MetalLB 스피커를 구성하기 전에, 로드 밸런서가 작동하는 데 필요한 네트워킹 구성을 수행할 수 있도록 스피커 DemonSet Elevated 권한을 부여합니다.

```
[netapp-user@rhel7 ~]$ oc adm policy add-scc-to-user privileged -n
metallb-system -z speaker
clusterrole.rbac.authorization.k8s.io/system:openshift:scc:privileged
added: "speaker"
```

6. MetalLB는 metallb-system 네임스페이스에서 ConfigMap을 만들어 구성합니다.

```
[netapp-user@rhel7 ~]$ vim metallb-config.yaml

apiVersion: v1
kind: ConfigMap
metadata:
  namespace: metallb-system
  name: config
data:
  config: |
    address-pools:
    - name: default
      protocol: layer2
      addresses:
      - 10.63.17.10-10.63.17.200

[netapp-user@rhel7 ~]$ oc create -f metallb-config.yaml
configmap/config created
```

7. 이제 로드 밸런서 서비스가 생성되면 MetalLB는 서비스에 외부 IP를 할당하고 ARP 요청에 응답하여 IP 주소를 알립니다.

ⓘ BGP 모드에서 MetalLB를 구성하려면 위의 6단계를 건너뛰고 MetalLB 설명서의 절차를 따르십시오 "여기".

F5 BIG-IP 로드 밸런서 설치

F5 BIG-IP는 광범위한 고급 프로덕션 등급 트래픽 관리 및 L4-L7 로드 밸런싱, SSL/TLS 오프로드, DNS, 방화벽 등의 보안 서비스를 제공하는 ADC(Application Delivery Controller)입니다. 이러한 서비스는 애플리케이션의 가용성, 보안 및 성능을 크게 향상시킵니다.

F5 BIG-IP는 전용 하드웨어, 클라우드 또는 온프레미스 가상 어플라이언스로 다양한 방식으로 구축 및 사용할 수 있습니다. 요구 사항에 따라 F5 BIG-IP를 탐색 및 배포하려면 여기 설명서를 참조하십시오.

F5 BIG-IP 서비스와 Red Hat OpenShift의 효율적인 통합을 위해 F5는 BIG-IP Container Ingress Service(CIS)를 제공합니다. CI는 특정 CRD(Custom Resource Definitions)에 대한 OpenShift API를 감시하고 F5 BIG-IP 시스템

구성을 관리하는 컨트롤러 포드로 설치됩니다. F5 BIG-IP CIS는 OpenShift에서 서비스 유형 로드 밸런서 및 경로를 제어하도록 구성할 수 있습니다.

또한 로드 밸런서를 서비스하기 위한 자동 IP 주소 할당의 경우 F5 IPAM 컨트롤러를 사용할 수 있습니다. F5 IPAM 컨트롤러는 사전 구성된 풀에서 IP 주소를 할당하는 ipamLabel 주석이 있는 loadbalancer 서비스용 OpenShift API를 감시하는 컨트롤러 포드로 설치됩니다.

이 페이지에는 F5 BIG-IP CIS 및 IPAM 컨트롤러에 대한 설치 및 구성 지침이 나와 있습니다. 사전 요구 사항으로 F5 BIG-IP 시스템을 배포하고 라이센스를 받아야 합니다. 또한 빅-IP VE 기본 라이센스와 함께 기본적으로 포함되는 SDN 서비스에 대한 라이센스가 필요합니다.

F5 BIG-IP는 독립 실행형 또는 클러스터 모드로 구축할 수 있습니다. 이러한 검증을 위해 F5 BIG-IP는 독립 실행형 모드로 구축되었지만, 생산 목적상 단일 장애 지점을 방지하기 위해 대규모 IP 클러스터를 사용하는 것이 좋습니다.

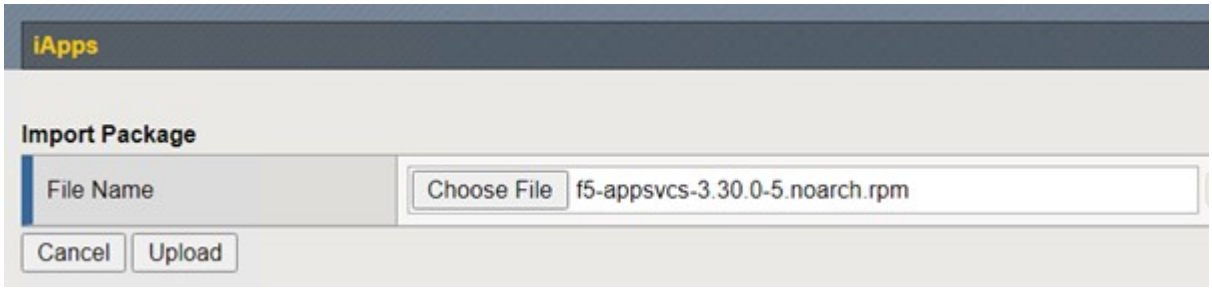
F5 BIG-IP 시스템은 전용 하드웨어, 클라우드 또는 12.x 이상의 버전이 있는 가상 어플라이언스로 구축할 수 있으며 F5 CIS와 통합할 수 있습니다. 이 문서의 목적에 따라 F5 BIG-IP 시스템은 예를 들어 BIG-IP VE 버전을 사용하는 가상 어플라이언스로 검증되었습니다.

검증된 릴리즈

제공합니다	소프트웨어 버전
Red Hat OpenShift	4.6 EUS, 4.7
F5 BIG-IP VE 버전	16.1.0
F5 컨테이너 침투 서비스	2.5.1
F5 IPAM 컨트롤러	0.1.4
F5 AS3	3.30.0

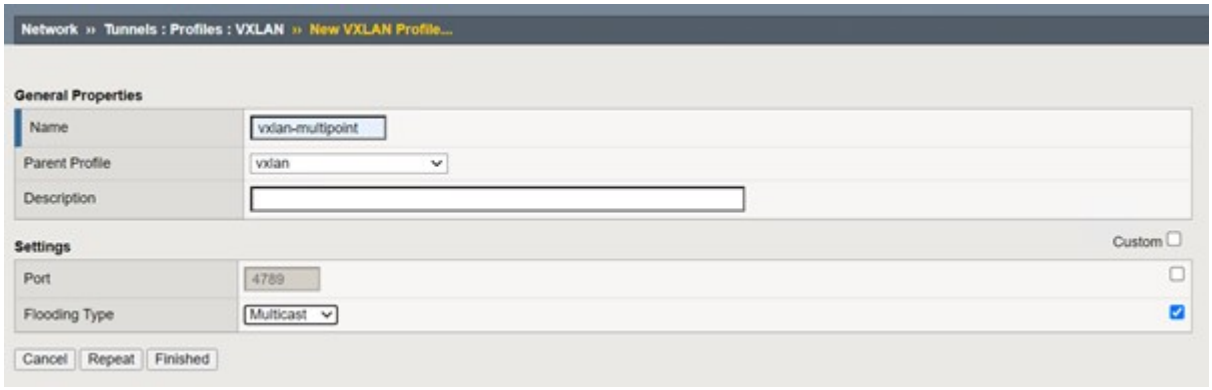
설치

1. 필수 명령 대신 BIG-IP 시스템이 JSON의 구성을 수락할 수 있도록 F5 Application Services 3 확장을 설치합니다. 로 이동합니다 "F5 AS3 GitHub 리포지토리"최신 RPM 파일을 다운로드합니다.
2. F5 BIG-IP 시스템에 로그인하고 iApps > 패키지 관리 LX 로 이동한 다음 가져오기 를 클릭합니다.
3. 파일 선택 을 클릭하고 다운로드한 AS3 RPM 파일을 선택한 다음 확인 을 클릭하고 업로드 를 클릭합니다.

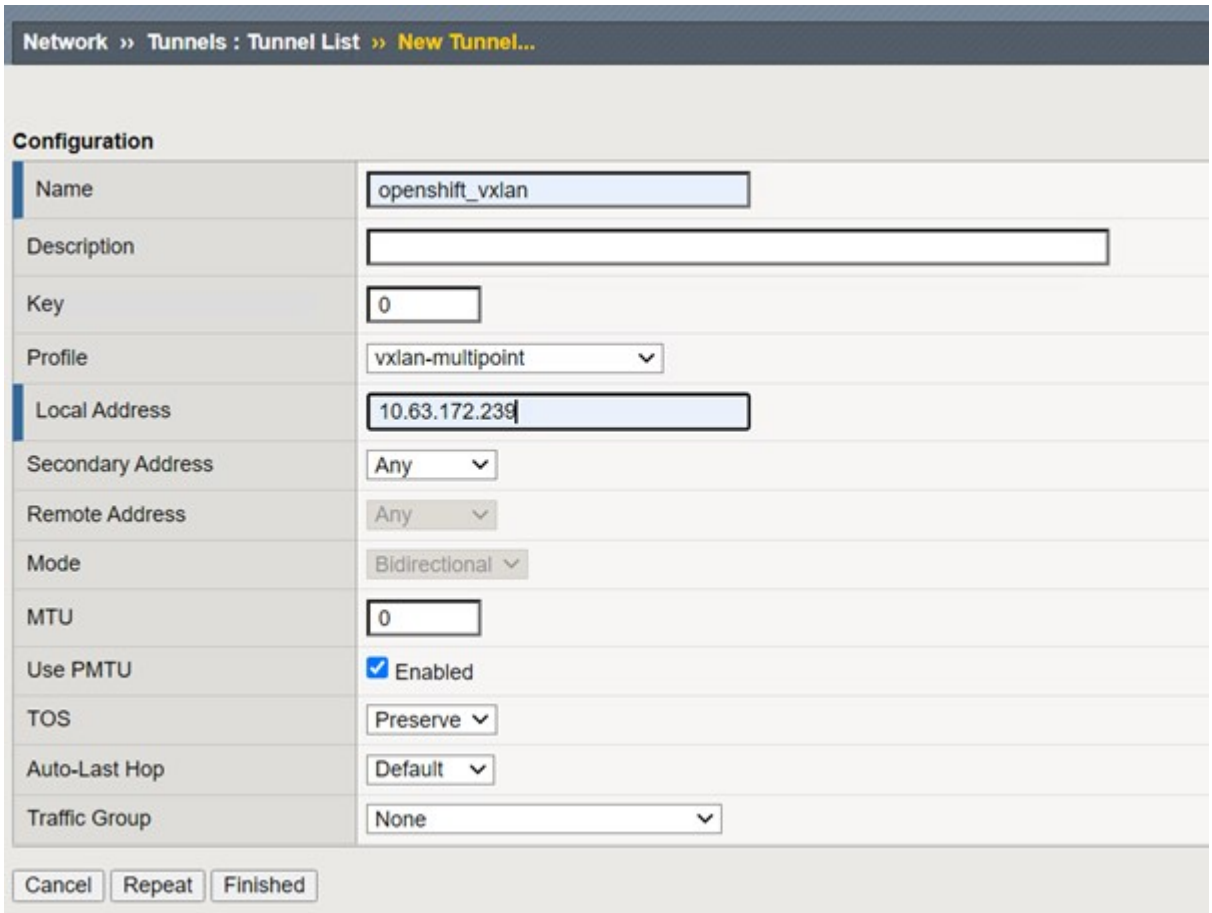


4. AS3 확장 프로그램이 성공적으로 설치되었는지 확인합니다.

5. 그런 다음 OpenShift와 BIG-IP 시스템 간의 통신에 필요한 리소스를 구성합니다. 먼저 OpenShift SDN용 BIG-IP 시스템에서 VXLAN 터널 인터페이스를 생성하여 OpenShift와 BIG-IP 서버 간에 터널을 생성합니다. 네트워크 > 터널 > 프로필 로 이동하고 생성 을 클릭한 다음 부모 프로필을 VXLAN 으로 설정하고 플러딩 유형 을 멀티캐스트로 설정합니다. 프로파일 이름을 입력하고 마침 을 클릭합니다.



6. 네트워크 > 터널 > 터널 목록 으로 이동하고 생성 을 클릭한 다음 터널의 이름과 로컬 IP 주소를 입력합니다. 이전 단계에서 만든 터널 프로필을 선택하고 마침 을 클릭합니다.



7. 클러스터 관리자 권한으로 Red Hat OpenShift 클러스터에 로그인합니다.
8. OpenShift에서 F5 BIG-IP 서버용 hostsubnet을 생성합니다. 그러면 서브넷이 OpenShift 클러스터에서 F5 BIG-IP 서버로 확장됩니다. 호스트 서브넷 YAML 정의를 다운로드합니다.

```
wget https://github.com/F5Networks/k8s-bigip-
ctlr/blob/master/docs/config_examples/openshift/f5-kctlr-openshift-
hostsubnet.yaml
```

9. 호스트 서브넷 파일을 편집하고 OpenShift SDN용 BIG-IP VTEP(VXLAN 터널) IP를 추가합니다.

```
apiVersion: v1
kind: HostSubnet
metadata:
  name: f5-server
  annotations:
    pod.network.openshift.io/fixed-vnid-host: "0"
    pod.network.openshift.io/assign-subnet: "true"
# provide a name for the node that will serve as BIG-IP's entry into the
cluster
host: f5-server
# The hostIP address will be the BIG-IP interface address routable to
the
# OpenShift Origin nodes.
# This address is the BIG-IP VTEP in the SDN's VXLAN.
hostIP: 10.63.172.239
```

사용자 환경에 맞게 호스트 팁 및 기타 세부 정보를 변경합니다.

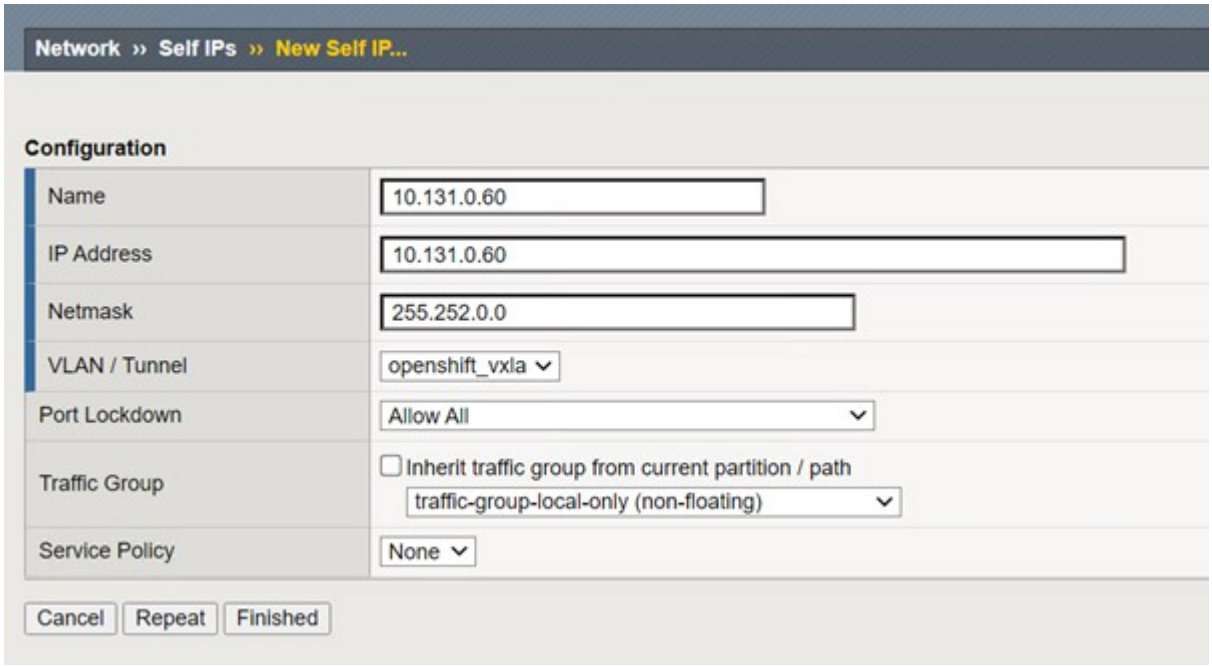
10. HostSubnet 리소스를 생성합니다.

```
[admin@rhel-7 ~]$ oc create -f f5-kctlr-openshift-hostsubnet.yaml

hostsubnet.network.openshift.io/f5-server created
```

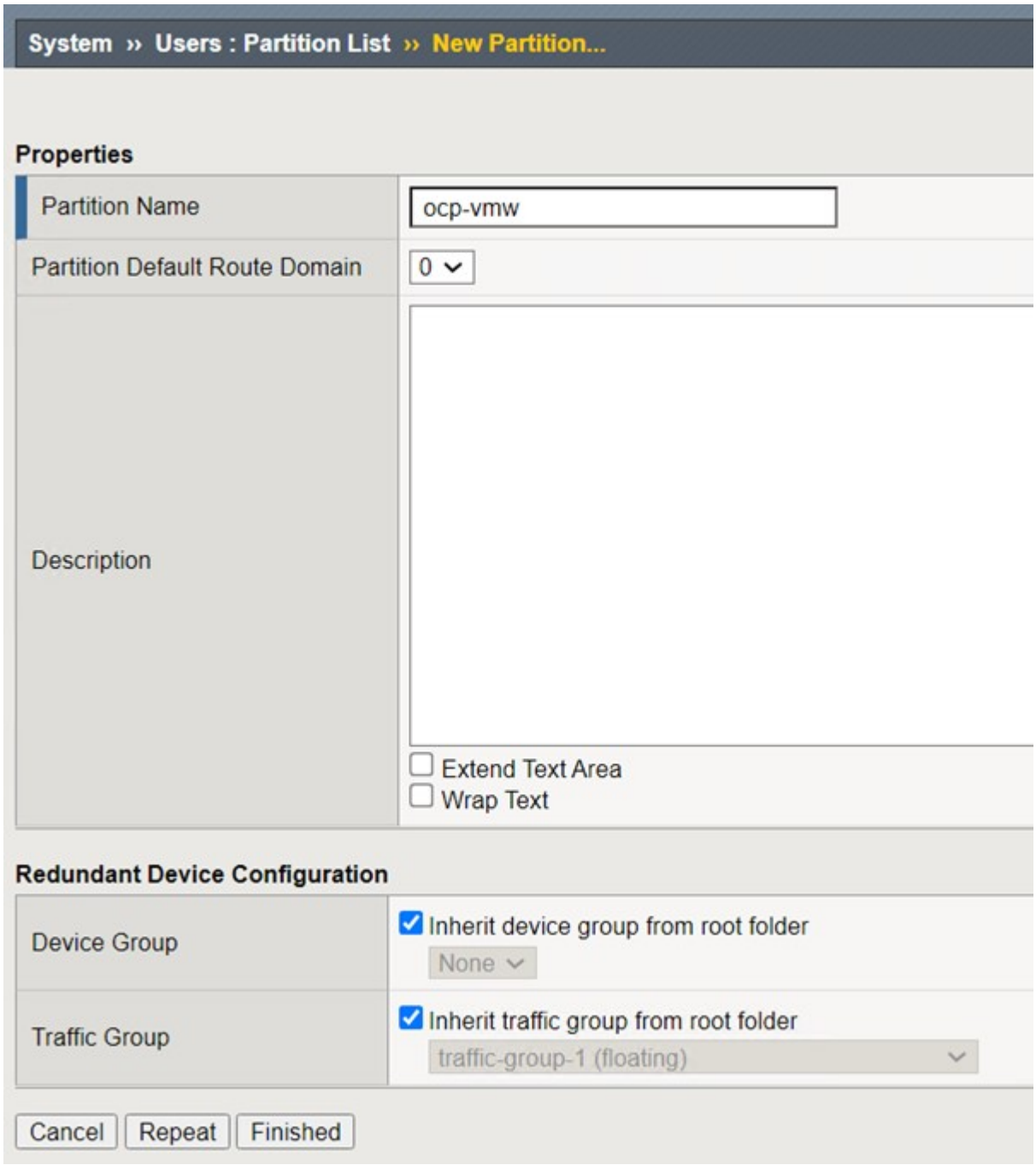
11. F5 BIG-IP 서버에 대해 생성된 호스트 서브넷의 클러스터 IP 서브넷 범위를 가져옵니다.

```
[admin@rhel-7 ~]$ oc get hostsubnet

NAME                           HOST                           HOST IP
SUBNET          EGRESS CIDRS   EGRESS IPS
f5-server                      f5-server                      10.63.172.239
10.131.0.0/23
ocp-vmw-nszws-master-0         ocp-vmw-nszws-master-0         10.63.172.44
10.128.0.0/23
ocp-vmw-nszws-master-1         ocp-vmw-nszws-master-1         10.63.172.47
10.130.0.0/23
ocp-vmw-nszws-master-2         ocp-vmw-nszws-master-2         10.63.172.48
10.129.0.0/23
ocp-vmw-nszws-worker-r8fh4     ocp-vmw-nszws-worker-r8fh4     10.63.172.7
10.130.2.0/23
ocp-vmw-nszws-worker-tvr46     ocp-vmw-nszws-worker-tvr46     10.63.172.11
10.129.2.0/23
ocp-vmw-nszws-worker-wdxhg     ocp-vmw-nszws-worker-wdxhg     10.63.172.24
10.128.2.0/23
ocp-vmw-nszws-worker-wg8r4     ocp-vmw-nszws-worker-wg8r4     10.63.172.15
10.131.2.0/23
ocp-vmw-nszws-worker-wtgfw     ocp-vmw-nszws-worker-wtgfw     10.63.172.17
10.128.4.0/23
```

12. F5 BIG-IP 서버에 해당하는 OpenShift의 호스트 서브넷 범위에서 IP를 사용하여 OpenShift VXLAN에서 셀프 IP를 생성합니다. F5 BIG-IP 시스템에 로그인하고 네트워크 > Self IP 로 이동한 다음 생성 을 클릭합니다. F5 BIG-IP 호스트 서브넷용으로 생성된 클러스터 IP 서브넷의 IP를 입력하고 VXLAN 터널을 선택한 다음 다른 세부 정보를 입력합니다. 그런 다음 마침 을 클릭합니다.



13. CIS에서 구성 및 사용할 F5 BIG-IP 시스템에 파티션을 생성합니다. 시스템 > 사용자 > 파티션 목록 으로 이동하고 생성 을 클릭한 다음 세부 정보를 입력합니다. 그런 다음 마침 을 클릭합니다.



F5는 CIS에서 관리하는 파티션에서 수동 구성을 수행하지 않을 것을 권장합니다.

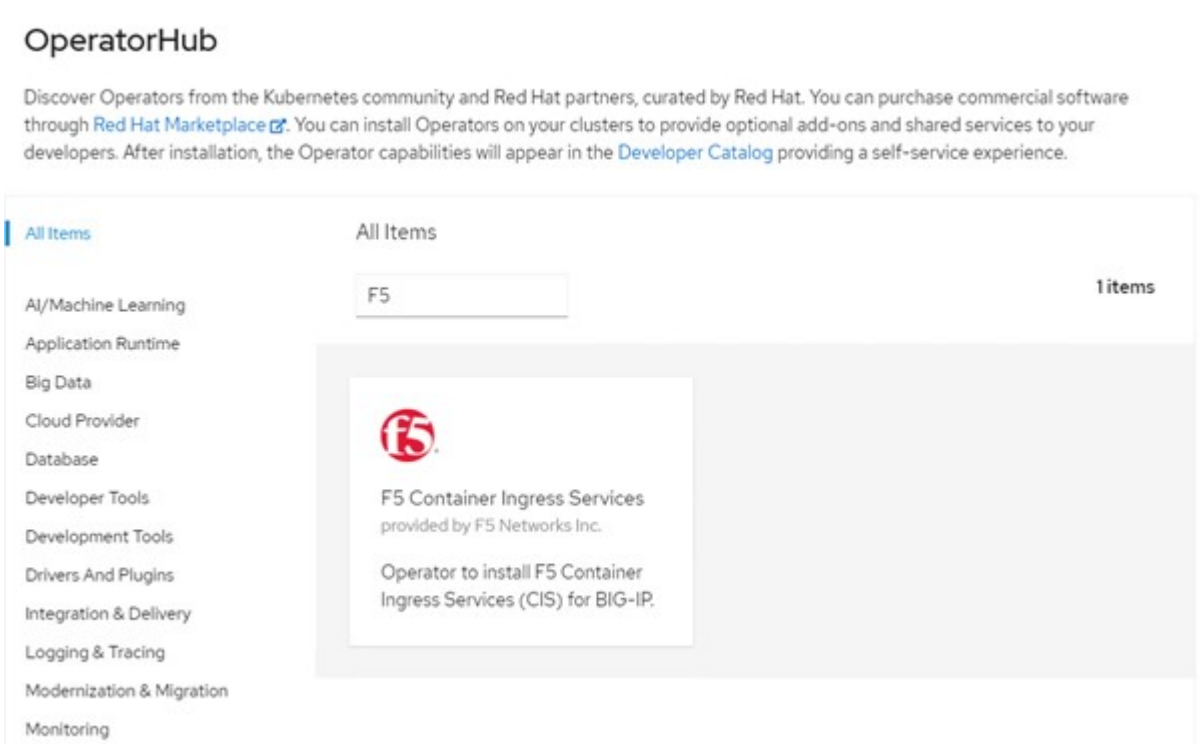
14. OperatorHub의 연산자를 사용하여 F5 BIG-IP CIS를 설치합니다. 클러스터 관리자 권한으로 Red Hat OpenShift 클러스터에 로그인하고 F5 BIG-IP 시스템 로그인 자격 증명을 사용하여 암호를 생성합니다. 이는 운영자의 필수 조건입니다.

```
[admin@rhel-7 ~]$ oc create secret generic bigip-login -n kube-system
--from-literal=username=admin --from-literal=password=admin

secret/bigip-login created
```

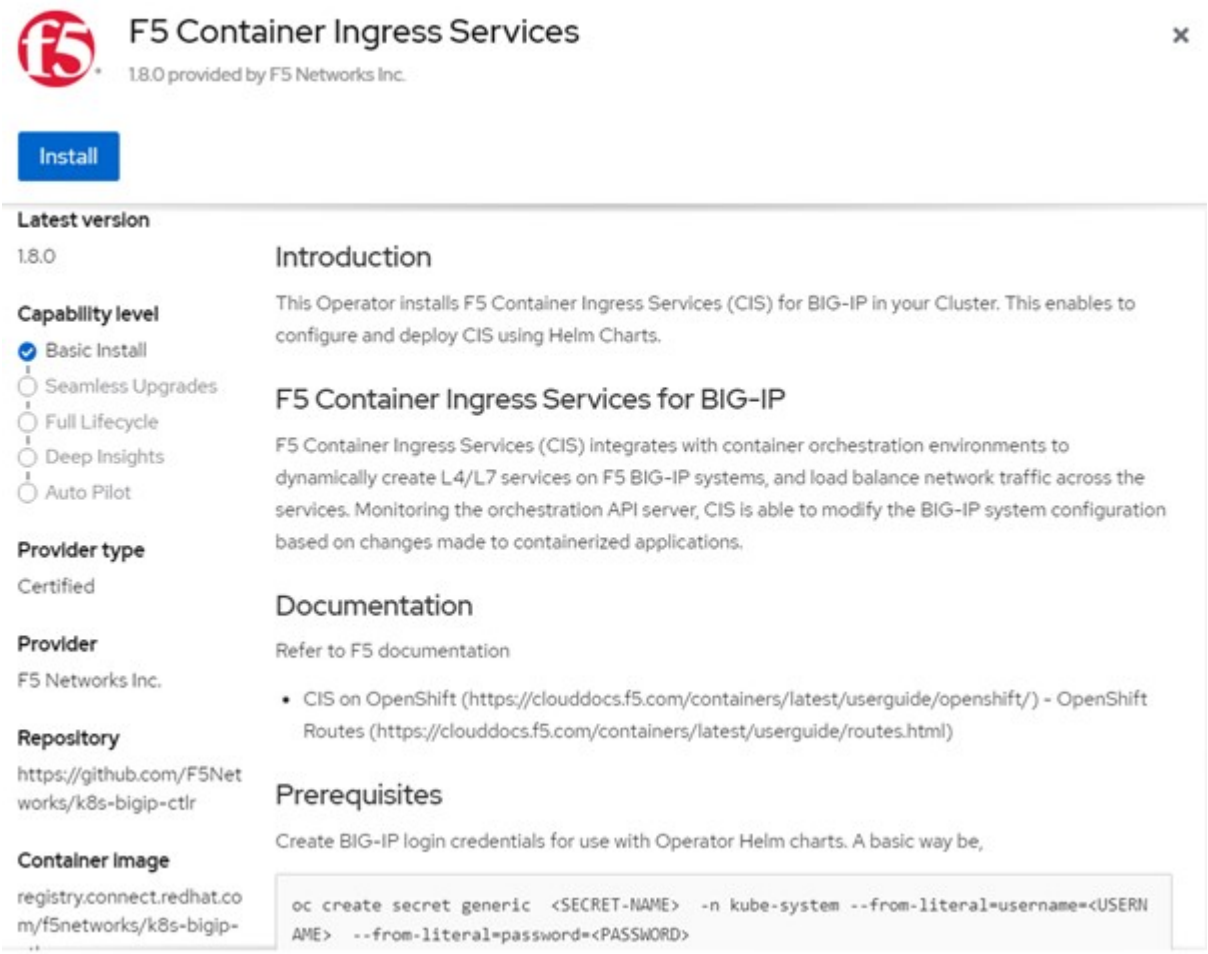
15. F5 CIS CRD를 설치합니다.

```
[admin@rhel-7 ~]$ oc apply -f
https://raw.githubusercontent.com/F5Networks/k8s-bigip-
ctlr/master/docs/config_examples/crd/Install/customresourcedefinitions.y
ml

customresourcedefinition.apiextensions.k8s.io/virtualservers.cis.f5.com
created
customresourcedefinition.apiextensions.k8s.io/tlsprofiles.cis.f5.com
created
customresourcedefinition.apiextensions.k8s.io/transportservers.cis.f5.co
m created
customresourcedefinition.apiextensions.k8s.io/externaldnss.cis.f5.com
created
customresourcedefinition.apiextensions.k8s.io/ingresslinks.cis.f5.com
created
```

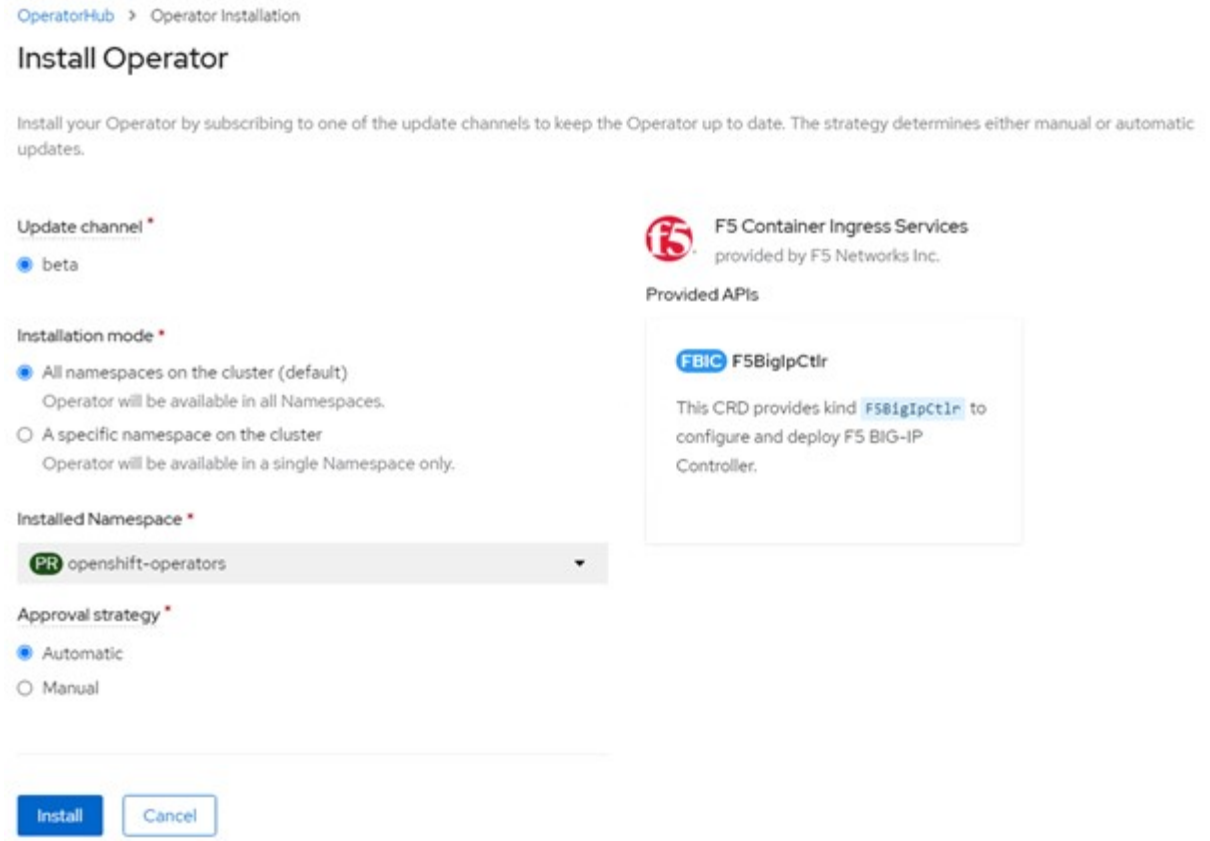
16. Operators > OperatorHub 로 이동하고 키워드 F5 를 검색한 다음 F5 Container Ingress Service 타일을 클릭합니다.



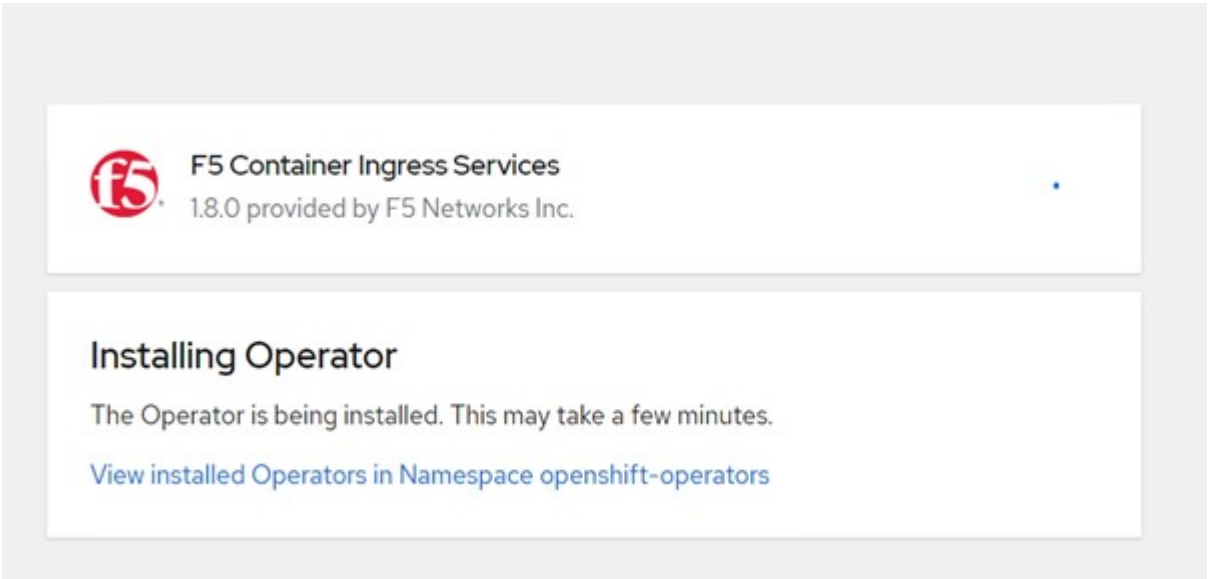
17. 운영자 정보를 읽고 설치를 클릭하십시오.



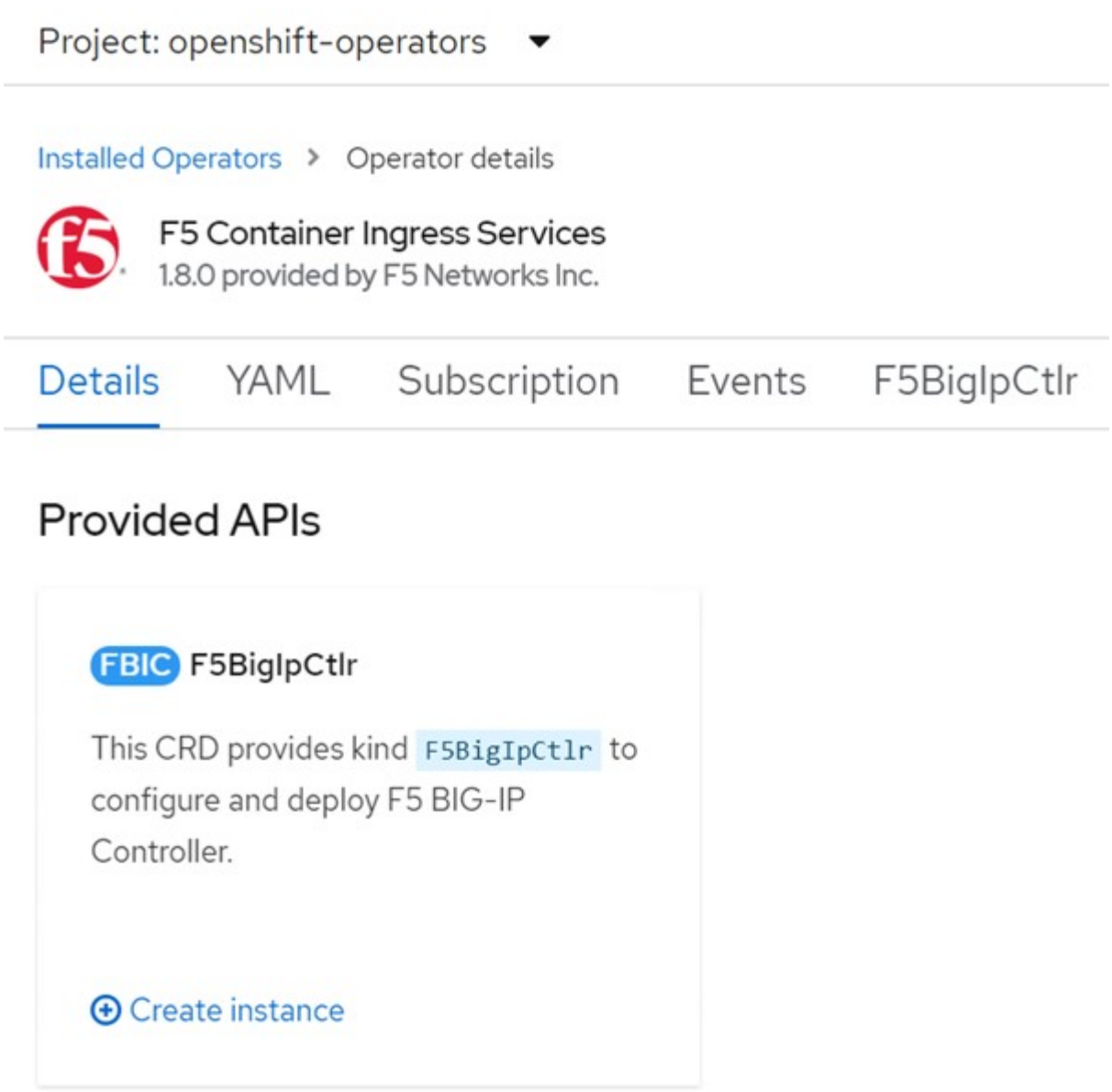
18. Install operator(설치 작업자) 화면에서 모든 기본 매개변수를 그대로 두고 Install(설치) 을 클릭합니다.



19. 운전자를 설치하는 데 시간이 걸립니다.



20. 운용자 설치 후 Installation Successful 메시지가 출력된다.
21. Operators > Installed Operators 로 이동하고 F5 Container Ingress Service 를 클릭한 다음 F5BigIpCtlr 타일에서 Create instance 를 클릭합니다.

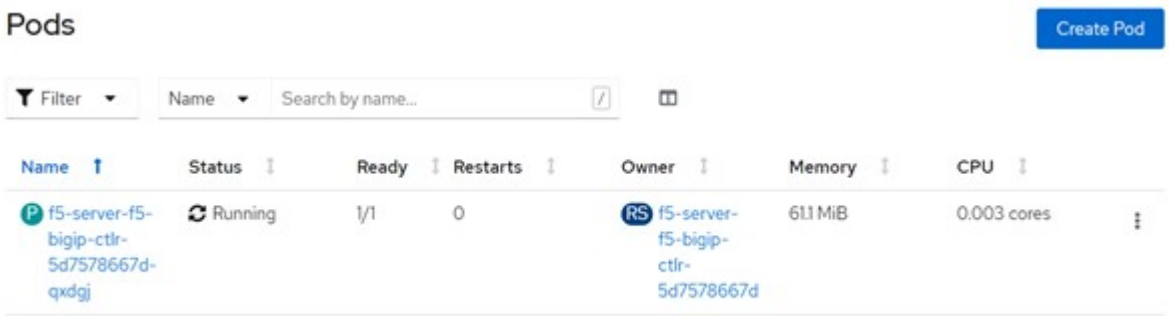


22. YAML View(YAML 보기) 를 클릭하고 필요한 매개변수를 업데이트한 후 다음 내용을 붙여 넣습니다.

콘텐츠를 복사하기 전에 설정 값을 반영하도록 아래의 매개 변수 'bigip_partition', 'openshift_sdn_name', 'bigip_url' 및 'bigip_login_secret'을 업데이트합니다.

```
apiVersion: cis.f5.com/v1
kind: F5BigIpCtlr
metadata:
  name: f5-server
  namespace: openshift-operators
spec:
  args:
    log_as3_response: true
    agent: as3
    log_level: DEBUG
    bigip_partition: ocp-vmw
    openshift_sdn_name: /Common/openshift_vxlan
    bigip_url: 10.61.181.19
    insecure: true
    pool-member-type: cluster
    custom_resource_mode: true
    as3_validation: true
    ipam: true
    manage_configmaps: true
  bigip_login_secret: bigip-login
  image:
    pullPolicy: Always
    repo: f5networks/cntr-ingress-svcs
    user: registry.connect.redhat.com
  namespace: kube-system
  rbac:
    create: true
  resources: {}
  serviceAccount:
    create: true
  version: latest
```

23. 이 콘텐츠를 붙여 넣은 후 만들기 를 클릭합니다. 그러면 kubbe-system 네임스페이스에 CIS 포드가 설치됩니다.



기본적으로 Red Hat OpenShift는 L7 로드 밸런싱을 위해 경로를 통해 서비스를 노출하는 방법을 제공합니다. 내장된 OpenShift 라우터는 이러한 경로의 트래픽을 광고 및 처리하는 역할을 합니다. 그러나 F5 CIS를 구성하여 외부 F5 BIG-IP 시스템을 통한 라우트를 지원할 수도 있습니다. 이 시스템은 보조 라우터로 실행하거나 자체 호스팅된 OpenShift 라우터에 대한 대체 라우터로 실행할 수 있습니다. CI는 OpenShift 라우트의 라우터 역할을 하는 BIG-IP 시스템에 가상 서버를 생성하고 BIG-IP는 광고 및 트래픽 라우팅을 처리합니다. 이 기능을 활성화하는 매개변수에 대한 자세한 내용은 여기 에서 설명서를 참조하십시오. 이러한 매개 변수는 APPS/v1 API의 OpenShift 배포 리소스에 대해 정의됩니다. 따라서 F5BigIpCtlr 리소스 cis.f5.com/v1 API와 함께 사용할 경우 매개변수 이름에 대한 하이픈(-)을 밑줄(_)으로 바꿉니다.

24. CIS 자원 생성에 전달되는 인자는 IPAM:TRUE, CUSTOM_RESOURCE_MODE:TRUE입니다. 이러한 매개변수는 IPAM 컨트롤러와 CIS 통합을 활성화하는 데 필요합니다. F5 IPAM 리소스를 생성하여 CIS가 IPAM 통합을 활성화했는지 확인합니다.

```
[admin@rhel-7 ~]$ oc get f5ipam -n kube-system

NAMESPACE    NAME                                AGE
kube-system    ipam.10.61.181.19.ocp-vmw       43s
```

25. F5 IPAM 컨트롤러에 필요한 서비스 계정, 역할 및 rolebinding을 만듭니다. YAML 파일을 생성하고 다음 내용을 붙여 넣습니다.

```
[admin@rhel-7 ~]$ vi f5-ipam-rbac.yaml

kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: ipam-ctlr-clusterrole
rules:
  - apiGroups: ["fic.f5.com"]
    resources: ["ipams","ipams/status"]
    verbs: ["get", "list", "watch", "update", "patch"]
---
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: ipam-ctlr-clusterrole-binding
  namespace: kube-system
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: ipam-ctlr-clusterrole
subjects:
  - apiGroup: ""
    kind: ServiceAccount
    name: ipam-ctlr
    namespace: kube-system
---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: ipam-ctlr
  namespace: kube-system
```

26. 리소스를 생성합니다.

```
[admin@rhel-7 ~]$ oc create -f f5-ipam-rbac.yaml

clusterrole.rbac.authorization.k8s.io/ipam-ctlr-clusterrole created
clusterrolebinding.rbac.authorization.k8s.io/ipam-ctlr-clusterrole-
binding created
serviceaccount/ipam-ctlr created
```

27. YAML 파일을 생성하고 아래에 제공된 F5 IPAM 배포 정의를 붙여 넣습니다.

아래 SPEC.template.spec.containers[0].args의 IP 범위 매개 변수를 업데이트하여 설정에 해당하는 ipamLabels 및 IP 주소 범위를 반영합니다.

IPAM 컨트롤러가 정의된 범위에서 IP 주소를 검색하고 할당하기 위해서는 ipamlabels ["range1" 및 "range2"(아래 예의 경우)에 부하 분산 장치 유형의 서비스에 대한 주석을 달아야 합니다.

```
[admin@rhel-7 ~]$ vi f5-ipam-deployment.yaml

apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    name: f5-ipam-controller
  name: f5-ipam-controller
  namespace: kube-system
spec:
  replicas: 1
  selector:
    matchLabels:
      app: f5-ipam-controller
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: f5-ipam-controller
    spec:
      containers:
      - args:
        - --orchestration=openshift
        - --ip-range='{"range1":"10.63.172.242-10.63.172.249",
"range2":"10.63.170.111-10.63.170.129"}'
        - --log-level=DEBUG
        command:
        - /app/bin/f5-ipam-controller
        image: registry.connect.redhat.com/f5networks/f5-ipam-
controller:latest
        imagePullPolicy: IfNotPresent
        name: f5-ipam-controller
      dnsPolicy: ClusterFirst
      restartPolicy: Always
      schedulerName: default-scheduler
      securityContext: {}
      serviceAccount: ipam-ctlr
      serviceAccountName: ipam-ctlr
```

28. F5 IPAM 컨트롤러 배포를 생성합니다.

```
[admin@rhel-7 ~]$ oc create -f f5-ipam-deployment.yaml

deployment/f5-ipam-controller created
```

29. F5 IPAM 컨트롤러 포드가 실행 중인지 확인합니다.

```
[admin@rhel-7 ~]$ oc get pods -n kube-system

NAME                                      READY   STATUS    RESTARTS
AGE
f5-ipam-controller-5986cff5bd-2bvn6       1/1     Running   0
30s
f5-server-f5-bigip-ctlr-5d7578667d-qxdgj  1/1     Running   0
14m
```

30. F5 IPAM 스키마를 만듭니다.

```
[admin@rhel-7 ~]$ oc create -f
https://raw.githubusercontent.com/F5Networks/f5-ipam-
controller/main/docs/_static/schemas/ipam_schema.yaml

customresourcedefinition.apiextensions.k8s.io/ipams.fic.f5.com
```

검증

1. loadbalancer 형식의 서비스를 생성합니다

```
[admin@rhel-7 ~]$ vi example_svc.yaml

apiVersion: v1
kind: Service
metadata:
  annotations:
    cis.f5.com/ipamLabel: range1
  labels:
    app: f5-demo-test
  name: f5-demo-test
  namespace: default
spec:
  ports:
  - name: f5-demo-test
    port: 80
    protocol: TCP
    targetPort: 80
  selector:
    app: f5-demo-test
  sessionAffinity: None
  type: LoadBalancer
```

```
[admin@rhel-7 ~]$ oc create -f example_svc.yaml

service/f5-demo-test created
```

2. IPAM Controller가 외부 IP를 할당하는지 확인한다.

```
[admin@rhel-7 ~]$ oc get svc

NAME           TYPE           CLUSTER-IP       EXTERNAL-IP
PORT(S)        AGE
f5-demo-test   LoadBalancer   172.30.210.108   10.63.172.242
80:32605/TCP   27s
```

3. 배포를 생성하고 생성된 로드 밸런서 서비스를 사용합니다.

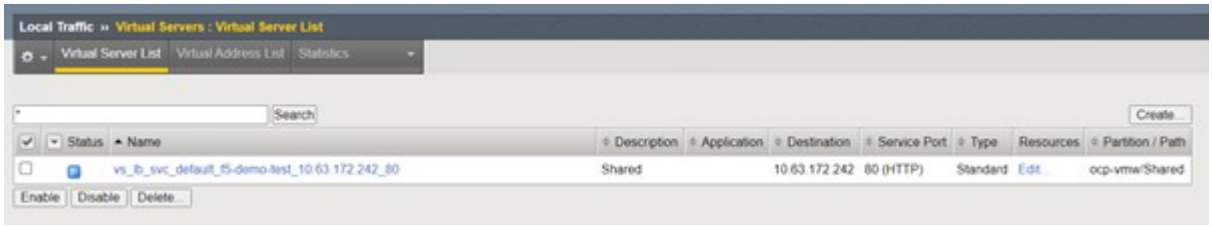
```
[admin@rhel-7 ~]$ vi example_deployment.yaml

apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: f5-demo-test
  name: f5-demo-test
spec:
  replicas: 2
  selector:
    matchLabels:
      app: f5-demo-test
  template:
    metadata:
      labels:
        app: f5-demo-test
    spec:
      containers:
      - env:
        - name: service_name
          value: f5-demo-test
        image: nginx
        imagePullPolicy: Always
        name: f5-demo-test
        ports:
        - containerPort: 80
          protocol: TCP
```

```
[admin@rhel-7 ~]$ oc create -f example_deployment.yaml

deployment/f5-demo-test created
```

4. Pod가 실행 중인지 확인합니다.

```
[admin@rhel-7 ~]$ oc get pods

NAME                           READY   STATUS    RESTARTS   AGE
f5-demo-test-57c46f6f98-47wwp   1/1     Running   0          27s
f5-demo-test-57c46f6f98-cl2m8   1/1     Running   0          27s
```

5. OpenShift에서 loadbalancer 유형의 서비스를 위해 BIG-IP 시스템에 해당 가상 서버가 생성되었는지 확인한다. Local Traffic > Virtual Servers > Virtual Server List로 이동합니다.



개인 이미지 레지스트리 만들기

같은 공용 레지스트리를 사용하여 대부분의 Red Hat OpenShift 배포에 사용됩니다 "키.오" 또는 "DockerHub를 참조하십시오" 고객의 대부분의 요구사항을 충족합니다. 그러나 고객이 자신의 개인 또는 사용자 지정 이미지를 호스팅하려는 경우가 있습니다.

이 절차에서는 Astra Trident 및 NetApp ONTAP에서 제공하는 영구 볼륨의 지원을 받는 개인 이미지 레지스트리 만들기에 대해 설명합니다.

Astra Control Center에는 Astra 컨테이너에 필요한 이미지를 호스팅하기 위한 레지스트리가 필요합니다. 다음 섹션에서는 Red Hat OpenShift 클러스터에 비공개 레지스트리를 설정하고 Astra Control Center 설치를 지원하는 데 필요한 이미지를 푸시하는 단계를 설명합니다.

개인 이미지 레지스트리를 만드는 중입니다

1. 현재 기본 스토리지 클래스에서 기본 주석을 제거하고 OpenShift 클러스터의 기본값으로 Trident 지원 스토리지 클래스에 주석을 추가합니다.

```
[netapp-user@rhel7 ~]$ oc patch storageclass thin -p '{"metadata":
{"annotations": {"storageclass.kubernetes.io/is-default-class":
"false"}}}'
storageclass.storage.k8s.io/thin patched

[netapp-user@rhel7 ~]$ oc patch storageclass ocp-trident -p
'{"metadata": {"annotations": {"storageclass.kubernetes.io/is-default-
class": "true"}}}'
storageclass.storage.k8s.io/ocp-trident patched
```

2. 'sepec' 부분에 다음과 같은 저장 매개변수를 입력하여 Imageregfollection 연산자를 편집합니다.

```
[netapp-user@rhel7 ~]$ oc edit
configs.imageregistry.operator.openshift.io

storage:
  pvc:
    claim:
```

3. 사용자 지정 호스트 이름을 사용하여 OpenShift 경로를 생성하기 위해 'sepec' 섹션에 다음 매개 변수를 입력합니다. 저장하고 종료합니다.

```
routes:
  - hostname: astra-registry.apps.ocp-vmw.cie.netapp.com
    name: netapp-astra-route
```

위 라우트 구성은 루트에 대한 사용자 지정 호스트 이름을 원하는 경우에 사용됩니다. OpenShift가 기본 호스트 이름을 사용하여 경로를 만들도록 하려면 'sepec' 섹션 ddefaultRoute: true'에 다음 매개 변수를 추가할 수 있습니다.

사용자 지정 **TLS** 인증서

루트에 사용자 지정 호스트 이름을 사용하는 경우 기본적으로 OpenShift Ingress 연산자의 기본 TLS 구성을 사용합니다. 그러나 루트에 사용자 지정 TLS 구성을 추가할 수 있습니다. 이렇게 하려면 다음 단계를 완료하십시오.

a. 루트의 TLS 인증서 및 키로 암호를 만듭니다.

```
[netapp-user@rhel7 ~]$ oc create secret tls astra-route-tls -n
openshift-image-registry –cert/home/admin/netapp-astra/tls.crt
--key=/home/admin/netapp-astra/tls.key
```

b. Imageregfollerator를 편집하고 다음 파라미터를 'sepec' 섹션에 추가합니다.

```
[netapp-user@rhel7 ~]$ oc edit
configs.imageregistry.operator.openshift.io

routes:
  - hostname: astra-registry.apps.ocp-vmw.cie.netapp.com
    name: netapp-astra-route
    secretName: astra-route-tls
```

4. 상상의 궤변운영자를 다시 편집하고 운영자의 관리상태를 마노화 상태로 변경합니다. 저장하고 종료합니다.

```
oc edit configs.imageregistry/cluster

managementState: Managed
```

5. 모든 전제 조건이 충족되면 개인 이미지 레지스트리에 대해 PVC, POD 및 서비스가 생성됩니다. 몇 분 후에 레지스트리가 가동되어야 합니다.

```
[netapp-user@rhel7 ~]$oc get all -n openshift-image-registry

NAME                                                  READY   STATUS
RESTARTS   AGE
pod/cluster-image-registry-operator-74f6d954b6-rb7zr   1/1     Running
3          90d
pod/image-pruner-1627257600-f5cpj                     0/1     Completed
0          2d9h
pod/image-pruner-1627344000-swqx9                     0/1     Completed
0          33h
pod/image-pruner-1627430400-rv5nt                     0/1     Completed
0          9h
pod/image-registry-6758b547f-6pnj8                    1/1     Running
0          76m
pod/node-ca-bwb5r                                     1/1     Running
0          90d
pod/node-ca-f8w54                                     1/1     Running
0          90d
pod/node-ca-gjx7h                                     1/1     Running
0          90d
pod/node-ca-lcx4k                                     1/1     Running
0          33d
pod/node-ca-v7zmx                                     1/1     Running
0          7d21h
pod/node-ca-xpppp                                     1/1     Running
0          89d

NAME                              TYPE        CLUSTER-IP       EXTERNAL-
IP   PORT(S)     AGE
service/image-registry            ClusterIP   172.30.196.167   <none>
5000/TCP    15h
service/image-registry-operator   ClusterIP   None             <none>
60000/TCP   90d

NAME                      DESIRED   CURRENT   READY   UP-TO-DATE
AVAILABLE   NODE SELECTOR            AGE
daemonset.apps/node-ca   6         6         6       6            6
kubernetes.io/os=linux   90d

NAME                                          READY   UP-TO-DATE
AVAILABLE   AGE
deployment.apps/cluster-image-registry-operator   1/1     1            1
90d
deployment.apps/image-registry                    1/1     1            1
15h
```

```
NAME                                                                DESIRED
CURRENT   READY   AGE
replicaset.apps/cluster-image-registry-operator-74f6d954b6   1         1
1       90d
replicaset.apps/image-registry-6758b547f                    1         1
1       76m
replicaset.apps/image-registry-78bfbd7f59                   0         0
0       15h
replicaset.apps/image-registry-7fcc8d6cc8                   0         0
0       80m
replicaset.apps/image-registry-864f88f5b                    0         0
0       15h
replicaset.apps/image-registry-cb47fffb                     0         0
0       10h

NAME                                 COMPLETIONS   DURATION   AGE
job.batch/image-pruner-1627257600   1/1           10s        2d9h
job.batch/image-pruner-1627344000   1/1           6s         33h
job.batch/image-pruner-1627430400   1/1           5s         9h

NAME                          SCHEDULE    SUSPEND   ACTIVE   LAST
SCHEDULE   AGE
cronjob.batch/image-pruner   0 0 * * *   False     0        9h
90d

NAME                                       HOST/PORT
PATH   SERVICES         PORT    TERMINATION   WILDCARD
route.route.openshift.io/public-routes   astra-registry.apps.ocp-
vmw.cie.netapp.com          image-registry   <all>   reencrypt     None
```

6. 수신 운영자 OpenShift 레지스트리 경로에 기본 TLS 인증서를 사용하는 경우 다음 명령을 사용하여 TLS 인증서를 가져올 수 있습니다.

```
[netapp-user@rhel7 ~]$ oc extract secret/router-ca --keys=tls.crt -n
openshift-ingress-operator
```

7. OpenShift 노드가 레지스트리에 액세스하여 이미지를 가져올 수 있도록 하려면 OpenShift 노드의 Docker 클라이언트에 인증서를 추가합니다. TLS 인증서를 사용하여 OpenShift-config 네임스페이스에 configmap을 만들고 이를 클러스터 이미지 구성에 패치하여 인증서를 신뢰할 수 있도록 합니다.

```
[netapp-user@rhel7 ~]$ oc create configmap astra-ca -n openshift-config
--from-file=astra-registry.apps.ocp-vmw.cie.netapp.com=tls.crt

[netapp-user@rhel7 ~]$ oc patch image.config.openshift.io/cluster
--patch '{"spec":{"additionalTrustedCA":{"name":"astra-ca"}}}'
--type=merge
```

8. OpenShift 내부 레지스트리는 인증에 의해 제어됩니다. 모든 OpenShift 사용자는 OpenShift 레지스트리에 액세스할 수 있지만 로그인한 사용자가 수행할 수 있는 작업은 사용자 권한에 따라 다릅니다.
   a. 사용자 또는 사용자 그룹이 레지스트리에서 이미지를 가져올 수 있도록 하려면 사용자에게 레지스트리 뷰어 역할이 할당되어 있어야 합니다.

   ```
   [netapp-user@rhel7 ~]$ oc policy add-role-to-user registry-viewer
   ocp-user

   [netapp-user@rhel7 ~]$ oc policy add-role-to-group registry-viewer
   ocp-user-group
   ```

   b. 사용자 또는 사용자 그룹이 이미지를 쓰거나 푸시할 수 있도록 하려면 사용자에게 레지스트리 편집기 역할이 할당되어 있어야 합니다.

   ```
   [netapp-user@rhel7 ~]$ oc policy add-role-to-user registry-editor
   ocp-user

   [netapp-user@rhel7 ~]$ oc policy add-role-to-group registry-editor
   ocp-user-group
   ```

9. OpenShift 노드가 레지스트리에 액세스하고 이미지를 푸시 또는 풀려면 풀 비밀을 구성해야 합니다.

```
[netapp-user@rhel7 ~]$ oc create secret docker-registry astra-registry-
credentials --docker-server=astra-registry.apps.ocp-vmw.cie.netapp.com
--docker-username=ocp-user --docker-password=password
```

10. 그런 다음 이 풀 암호는 serviceaccount에 패치하거나 해당 pod 정의에서 참조할 수 있습니다.
    a. 서비스 계정에 패치를 적용하려면 다음 명령을 실행합니다.

    ```
    [netapp-user@rhel7 ~]$ oc secrets link <service_account_name> astra-
    registry-credentials --for=pull
    ```

    b. POD 정의의 Pull Secret을 참조하려면, 'sepec' 부분에 다음 파라미터를 추가한다.

```
imagePullSecrets:
  - name: astra-registry-credentials
```

11. OpenShift 노드 이외의 워크스테이션에서 이미지를 푸시하거나 풀려면 다음 단계를 완료하십시오.

a. Docker 클라이언트에 TLS 인증서를 추가합니다.

```
[netapp-user@rhel7 ~]$ sudo mkdir /etc/docker/certs.d/astra-
registry.apps.ocp-vmw.cie.netapp.com

[netapp-user@rhel7 ~]$ sudo cp /path/to/tls.crt
/etc/docker/certs.d/astra-registry.apps.ocp-vmw.cie.netapp.com
```

b. OC 로그인 명령을 사용하여 OpenShift에 로그인합니다.

```
[netapp-user@rhel7 ~]$ oc login --token=sha256~D49SpB_lesSrJYwrM0LIO
-VRcjWHu0a27vKa0 --server=https://api.ocp-vmw.cie.netapp.com:6443
```

c. podman/docker 명령을 사용하여 OpenShift 사용자 자격 증명을 사용하여 레지스트리에 로그인합니다.

포더맨

```
[netapp-user@rhel7 ~]$ podman login astra-registry.apps.ocp-
vmw.cie.netapp.com -u kubeadmin -p $(oc whoami -t) --tls
-verify=false
```

+참고: kubadmin 사용자를 사용하여 개인 레지스트리에 로그인하는 경우 암호 대신 토큰을 사용합니다.

Docker 를 참조하십시오

```
[netapp-user@rhel7 ~]$ docker login astra-registry.apps.ocp-
vmw.cie.netapp.com -u kubeadmin -p $(oc whoami -t)
```

+참고: kubadmin 사용자를 사용하여 개인 레지스트리에 로그인하는 경우 암호 대신 토큰을 사용합니다.

d. 이미지를 밀거나 당깁니다.

포더맨

```
[netapp-user@rhel7 ~]$ podman push astra-registry.apps.ocp-
vmw.cie.netapp.com/netapp-astra/vault-controller:latest
[netapp-user@rhel7 ~]$ podman pull astra-registry.apps.ocp-
vmw.cie.netapp.com/netapp-astra/vault-controller:latest
```

Docker 를 참조하십시오

```
[netapp-user@rhel7 ~]$ docker push astra-registry.apps.ocp-
vmw.cie.netapp.com/netapp-astra/vault-controller:latest
[netapp-user@rhel7 ~]$ docker pull astra-registry.apps.ocp-
vmw.cie.netapp.com/netapp-astra/vault-controller:latest
```