

Cloudfify policy-driven Orchestration for Multi-Cloud Services with Kubernetes-managed containerized F5 Networks functions

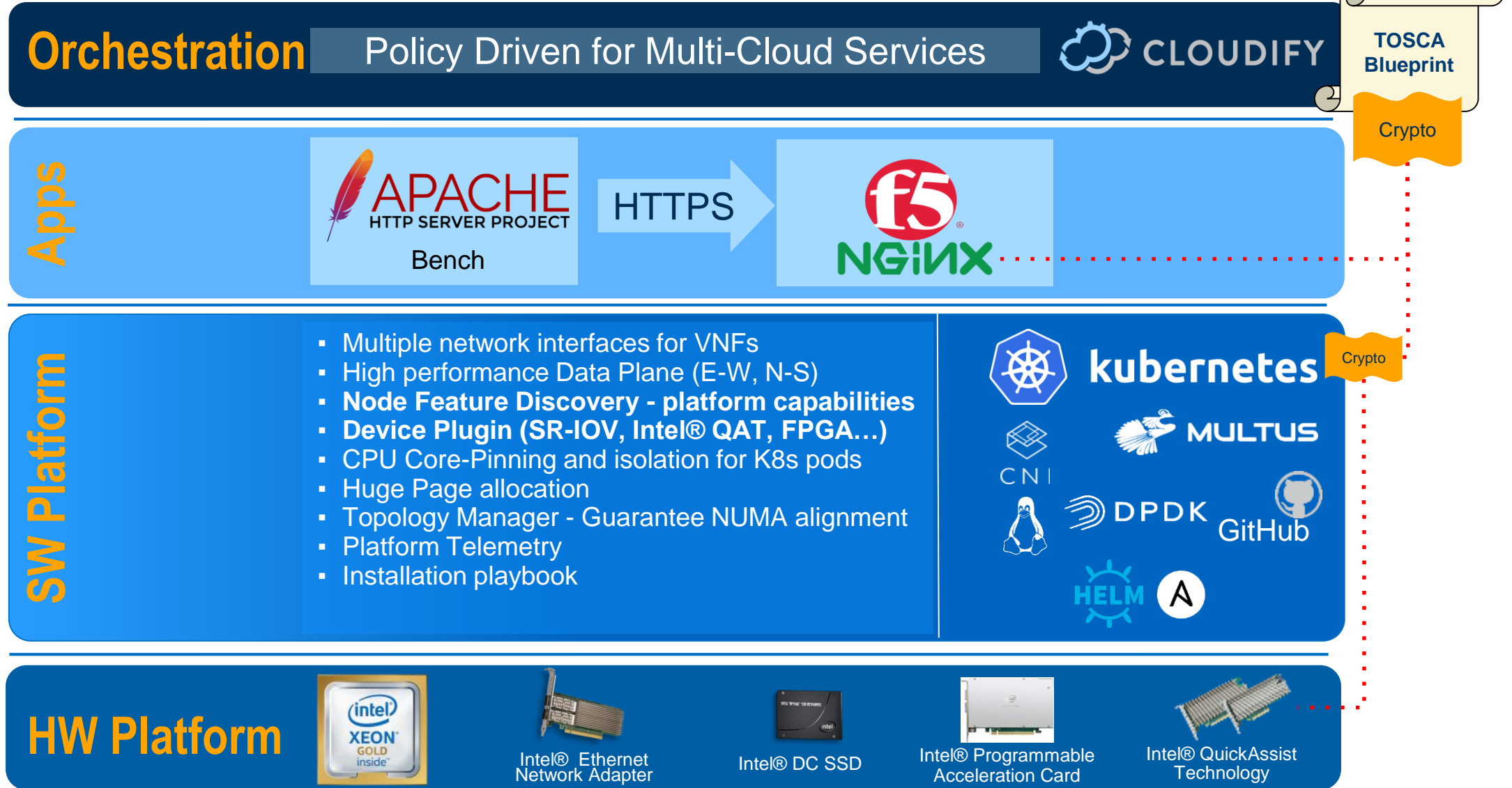
Shay Naeh, Senior Solution Architect, Cloudfify

Philip Klatte, Senior Product Manager, F5 Networks

Petar Torre, Principal Engineer, Intel

14 October 2019

Enabling Telco Cloud in layered stack with Kubernetes



Cloudify Site Map

Cloudify Spire 5.0 default_tenant admin

- Dashboard
- Cloudify Catalog
- Local Blueprints
- Deployments
- Site Management
- Tenant Management
- Admin Operations
- System Resources
- Statistics
- Logs

Get started with the Hello World Wizard

3 BLUEPRINTS **3 DEPLOYMENTS** **19 PLUGINS** **1 COMPUTE NODES** **0 RUNNING EXECUTIONS**

Deployment Wizard **Upload Blueprint** **Create Deployment** **Upload Plugin**

Blueprint: Deployment: Execution Status:

Sites Map

United States of America

Leaflet | Wikimedia

Executions Executions Statuses Graph

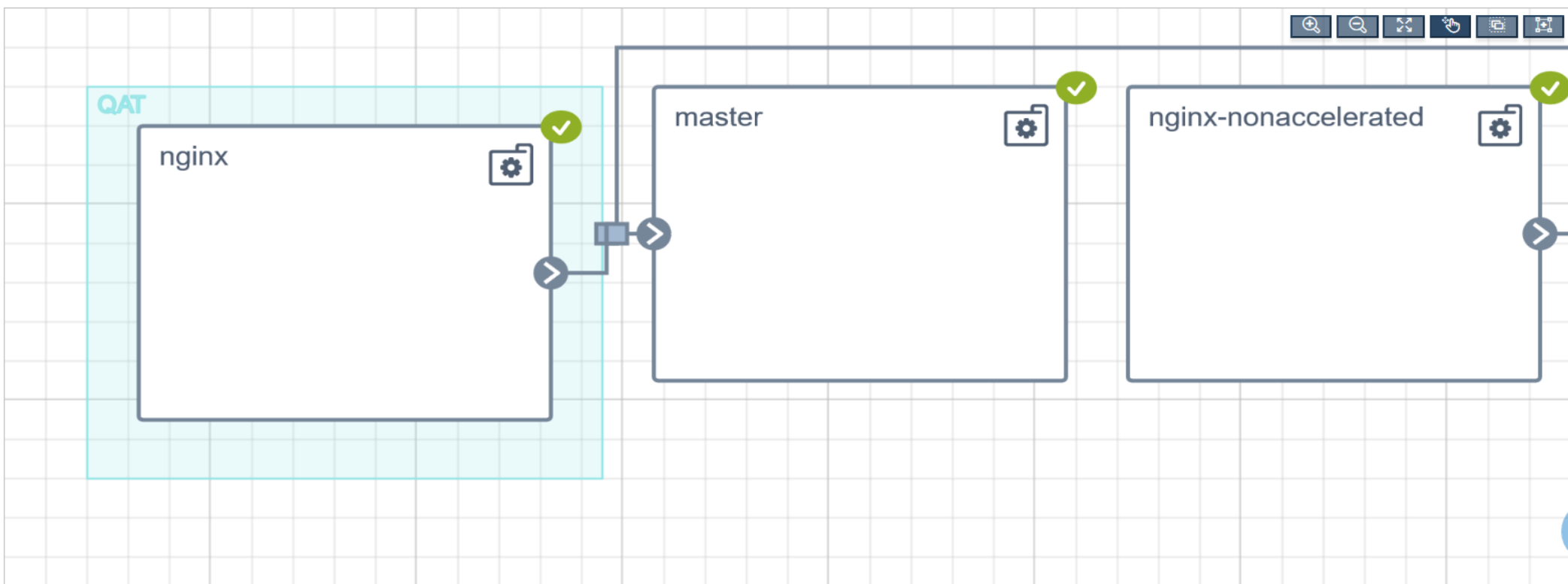
Deployments

- Dashboard
- Cloudify Catalog
- Local Blueprints
- Deployments
- Site Management
- Tenant Management
- Admin Operations
- System Resources
- Statistics
- Logs

Deployments > pods-ab

Execute workflow Update deployment Delete deployment

Deployment Topology



Placement with Intel® QuickAssist Technology (QAT)

Cloudify Spire 5.0 default_tenant admin

Dashboard
Cloudify Catalog
Local Blueprints
Deployments
Site Management
Tenant Management
Admin Operations
System Resources
Statistics
Logs

Deployments > pods-ab

Execute workflow Update deployment

Deployment Topology

QAT

nginx

master

nginx-nonaccelerated

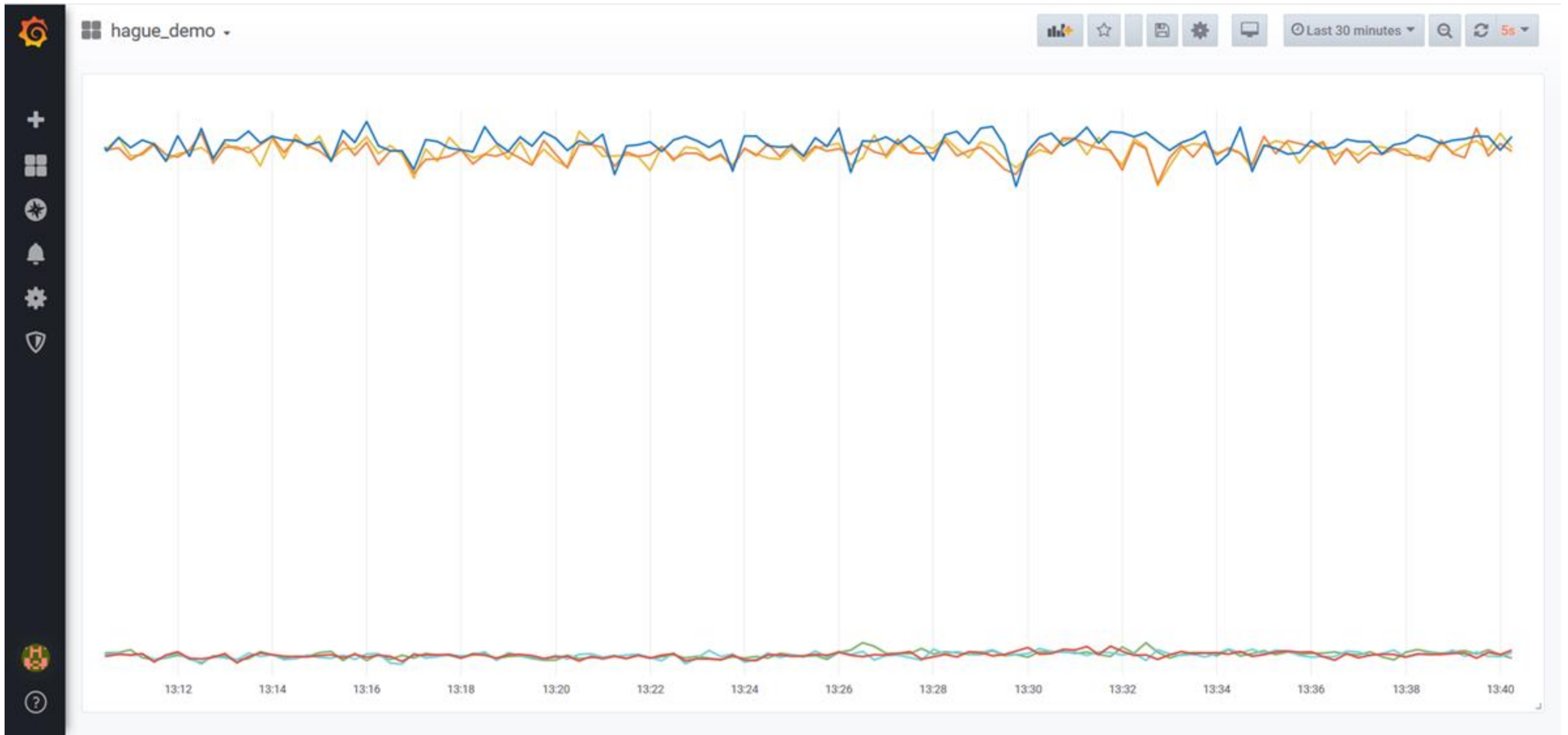
NGINX deployed on QAT K8S node

NGINX deployed on any K8S node

```
Listing labels for Node./node2:
node.alpha.kubernetes-incubator.io/nfd-network-sriov=true
node.alpha.kubernetes-incubator.io/nfd-cpuid-HTT=true
node.alpha.kubernetes-incubator.io/nfd-cpuid-SSSE3=true
node.alpha.kubernetes-incubator.io/nfd-cpuid-HLE=true
node.alpha.kubernetes-incubator.io/nfd-cpuid-FMA3=true
node.alpha.kubernetes-incubator.io/nfd-cpuid-AVX2=true
kubernetes.io/hostname=node2
node.alpha.kubernetes-incubator.io/nfd-cpuid-MMXEXT=true
node.alpha.kubernetes-incubator.io/nfd-cpuid-AVX512DQ=true
node.alpha.kubernetes-incubator.io/nfd-cpuid-MMX=true
node.alpha.kubernetes-incubator.io/nfd-rdt-RDTMBA=true
node.alpha.kubernetes-incubator.io/nfd-cpuid-RTM=true
qat=true
node.alpha.kubernetes-incubator.io/nfd-cpuid-F16C=true
node.alpha.kubernetes-incubator.io/nfd-cpuid-BMI2=true
node.alpha.kubernetes-incubator.io/nfd-rdt-RDTCMT=true
node.alpha.kubernetes-incubator.io/nfd-cpuid-ADX=true
node-role.kubernetes.io/node=
node.alpha.kubernetes-incubator.io/nfd-memory-numa=true
node.alpha.kubernetes-incubator.io/nfd-rdt-RDTL3CA=true
node.alpha.kubernetes-incubator.io/nfd-cpuid-SSE=true
node.alpha.kubernetes-incubator.io/nfd-rdt-RDTMON=true
beta.kubernetes.io/arch=amd64
node.alpha.kubernetes-incubator.io/nfd-cpuid-NX=true
node.alpha.kubernetes-incubator.io/nfd-cpuid-SSE4.2=true
node.alpha.kubernetes-incubator.io/nfd-pstate-turbo=true
node.alpha.kubernetes-incubator.io/nfd-rdt-RDTMBM=true
node.alpha.kubernetes-incubator.io/nfd-cpuid-RDTSCP=true
node.alpha.kubernetes-incubator.io/nfd-iommu-enabled=true
node.alpha.kubernetes-incubator.io/nfd-cpuid-LZCNT=true
node.alpha.kubernetes-incubator.io/nfd-cpuid-SSE2=true
node.alpha.kubernetes-incubator.io/nfd-cpuid-AVX512VL=true
beta.kubernetes.io/os=linux
node.alpha.kubernetes-incubator.io/nfd-cpuid-AESNI=true
node.alpha.kubernetes-incubator.io/nfd-cpuid-AVX512CD=true
node.alpha.kubernetes-incubator.io/nfd-storage-nonrotationaldisk=true
node.alpha.kubernetes-incubator.io/node-feature-discovery.version=v0.3.0
```

```
[root@master1 ~]# kubectl describe pod demo-g2xln
Name:          demo-g2xln
Namespace:    default
Priority:      0
PriorityClassName: <none>
Node:         node2/192.168.0.237
Start Time:   Thu, 10 Oct 2019 13:13:23 +0300
Labels:       <none>
Annotations:  k8s.v1.cni.cncf.io/networks-status:
              [{"name": "cni0",
               "interface": "eth0",
               "ips": ["10.244.2.23"],
               "mac": "0a:58:0a:f4:02:17",
               "default": true,
               "dns": {}}]
Status:       Running
IP:           10.244.2.23
Containers:
  demonginx:
    image: registry.k8s.io/nginxinc/nginx-debugger
    ports:
      - containerPort: 80
    conditions:
      Type           Status
      Initialized    True
      Ready           True
      ContainersReady True
      PodScheduled   True
Volumes:
  default-token-mljgc:
    Type:          Secret (a volume populated by a Secret)
    SecretName:    default-token-mljgc
    Optional:      false
QoS Class:        Burstable
Node-Selectors:   qat=true
Tolerations:      node.kubernetes.io/not-ready:NoExecute for 300s
                  node.kubernetes.io/unreachable:NoExecute for 300s
Events:           <none>
```


Results



Intent based placement

The screenshot displays the Cloudify Spire dashboard with a 'Deploy Service' modal window open. The dashboard background shows various metrics: 23 EDGE SITES, 230, 41, 4125, and 120 RUNNING EXECUTIONS. The modal window is titled 'Deploy Service' and contains the following sections:

- Region:** A search box labeled 'Select regions'.
- Edge Type:** Three icons with checkboxes: a green triangle (checked), a blue gear, and a purple 'X' (checked).
- Infrastructure:** Four icons with checkboxes: AWS (checked), vmware, and two others.
- Status:** Three icons with checkboxes: Green (checked), Overloaded, and Service down.
- Placement policy:** Three checked options: Low Latency, CPU Intensive, and Huge Memory. This section is circled in green.

On the right side of the modal, there is a circular progress indicator showing '100% of edges' and a summary table:

Edges stacks	Edges
20	312

At the bottom of the modal, there is a 'Review stacks' button and a 'Back' link.

Cloud Instances – F5 BIG IP on AWS

The screenshot displays the Cloudify Premium 5.0 console interface. The top navigation bar includes the Cloudify logo and the text "Cloudify Premium 5.0". The left sidebar contains a menu with items: Dashboard, Cloudify Catalog, Local Blueprints, Deployments, Site Management, Tenant Management, Admin Operations, System Resources, Statistics, and Logs. The main content area is titled "Deployments | BIG-IP" and features three action buttons: "Execute workflow", "Update deployment", and "Delete deployment". Below these buttons is a "Deployment Topology" diagram. A callout box with the text "Big-IP instance deployment on AWS" has an arrow pointing to the "host" node in the topology. The topology consists of several interconnected nodes: "host", "ip", "nic", "security_group" (containing "security_group_rules"), "cloud_init", "subnet", "keypair", "centos_core_ami", and "vpc". A watermark "Click to release scroller" is visible in the center of the topology diagram. At the bottom of the console, there is a "Deployment Nodes" section with a search bar and a table with the following columns: Name, Type, Contained in, Connected to, Host, Creator, # Instances, and Groups.

Name	Type	Contained in	Connected to	Host	Creator	# Instances	Groups
------	------	--------------	--------------	------	---------	-------------	--------

F5 Use Cases: Initial Demo Setup

Cloudify Orchestration Leverages Node Feature Discovery
to Optimize NGINX Service Placement

Intel® EPA and Cloudify® Orchestration – Setup

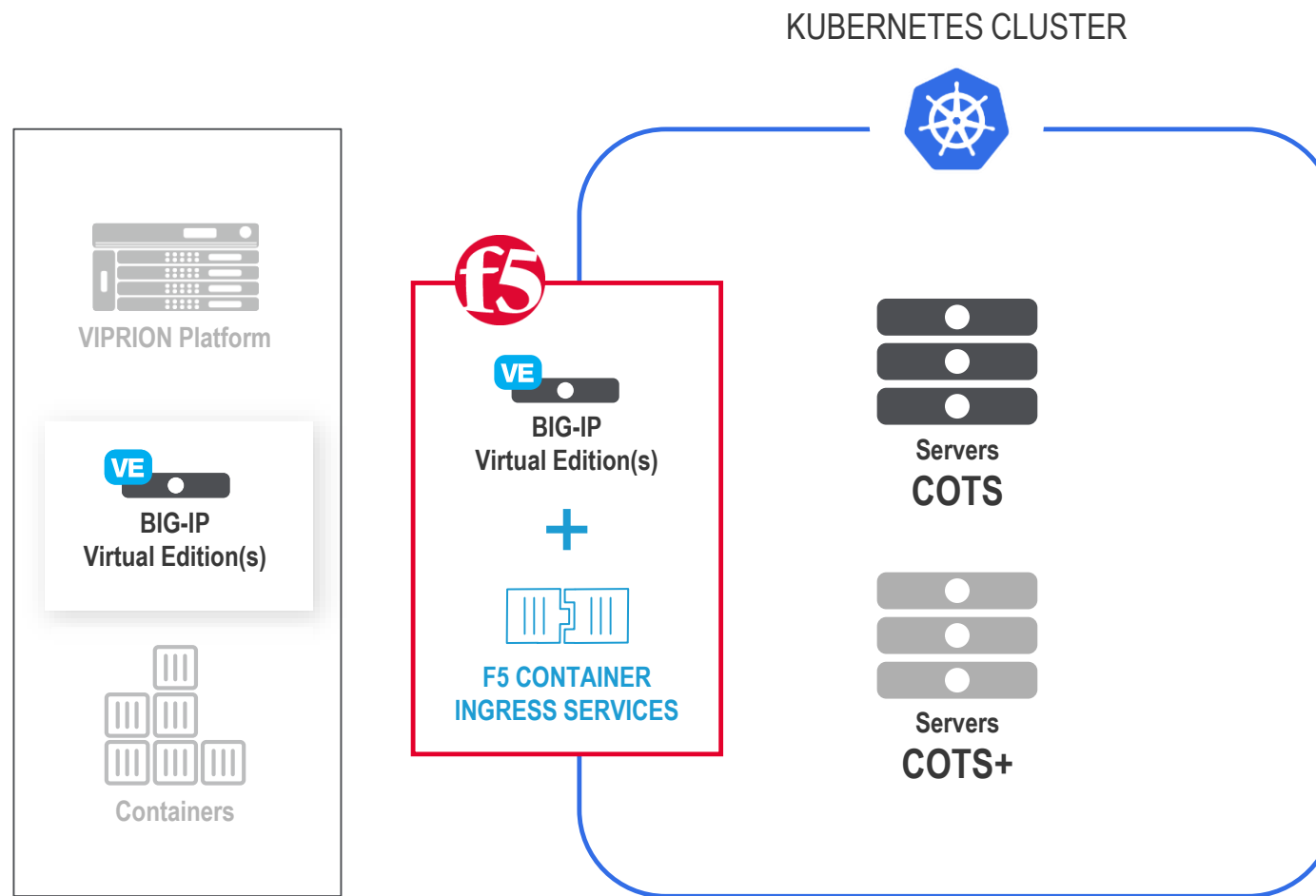
Ingress examples

Security (PAC, DDOS)

Encrypt/Decrypt

Traffic steering

F5 Container Ingress Service can be deployed using virtual or physical models. For the demo, we used BIG-IP Virtual Edition.



Intel® EPA and Cloudfify® Orchestration – Setup

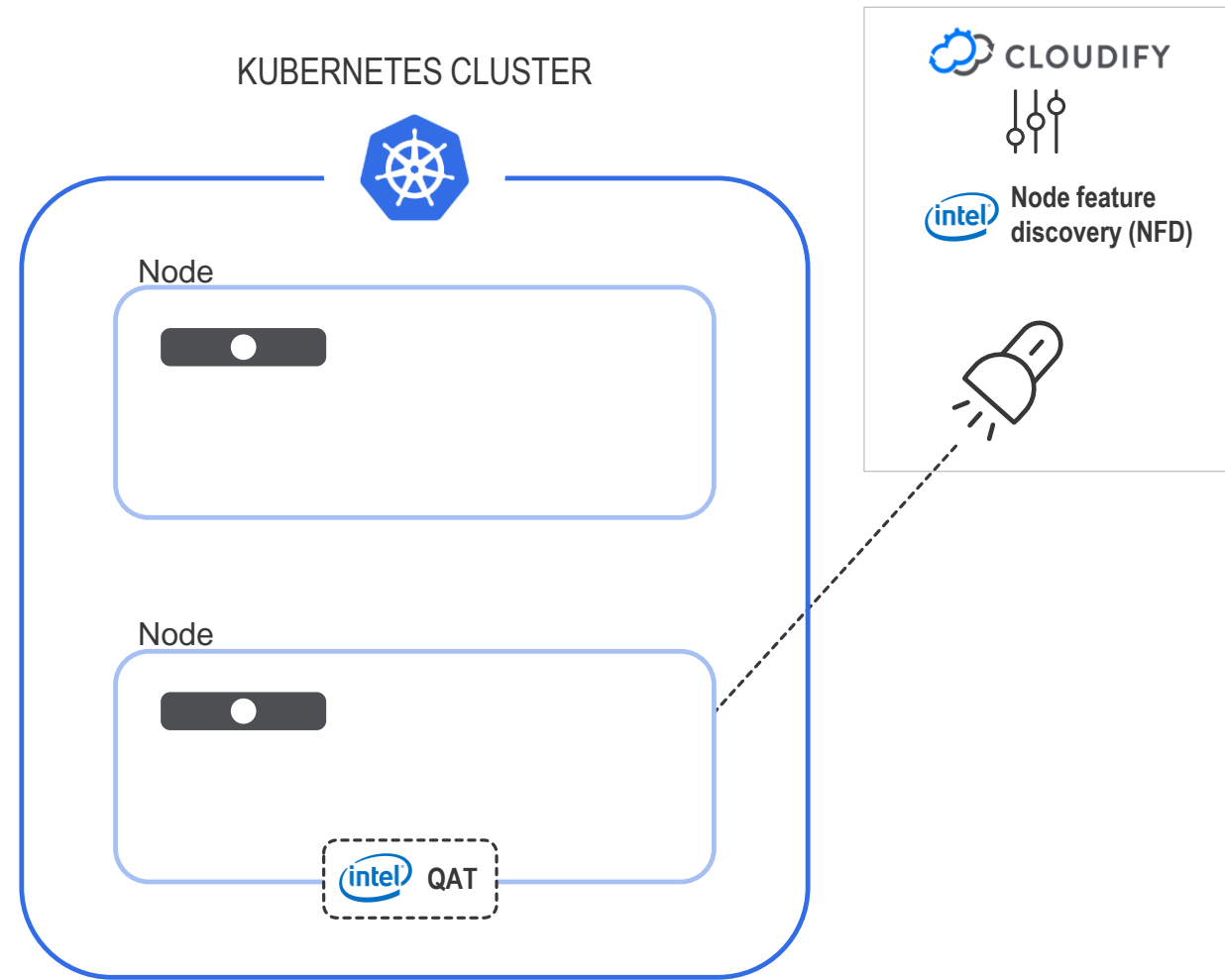
Ingress examples

Security (PAC, DDOS)

Encrypt/Decrypt

Traffic steering

**Cloudfify orchestrates
labeling nodes, using Intel
Node Feature Discovery
ex. Intel® QuickAssist
Technology, (Intel® QAT)**



Intel® EPA and Cloudify® Orchestration – Setup

Ingress examples

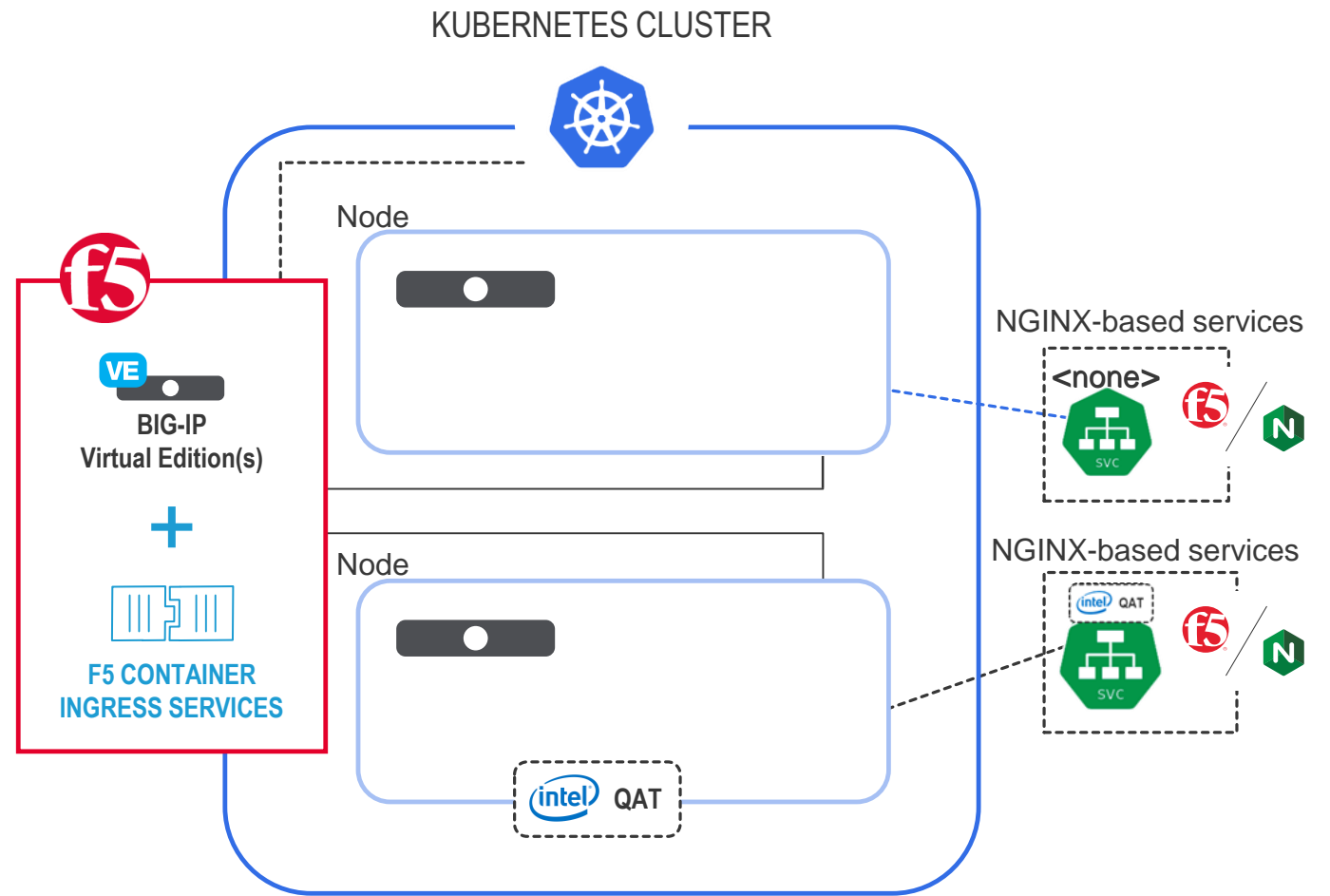
Security (PAC, DDOS)

Encrypt/Decrypt

Traffic steering

Pods are deployed
based on node affinity,
with some optimized for
Intel® QAT offload.

F5 CIS establishes
connections to the new
services, utilizing the
K8s API.



Intel® EPA and Cloudfify® Orchestration – Setup

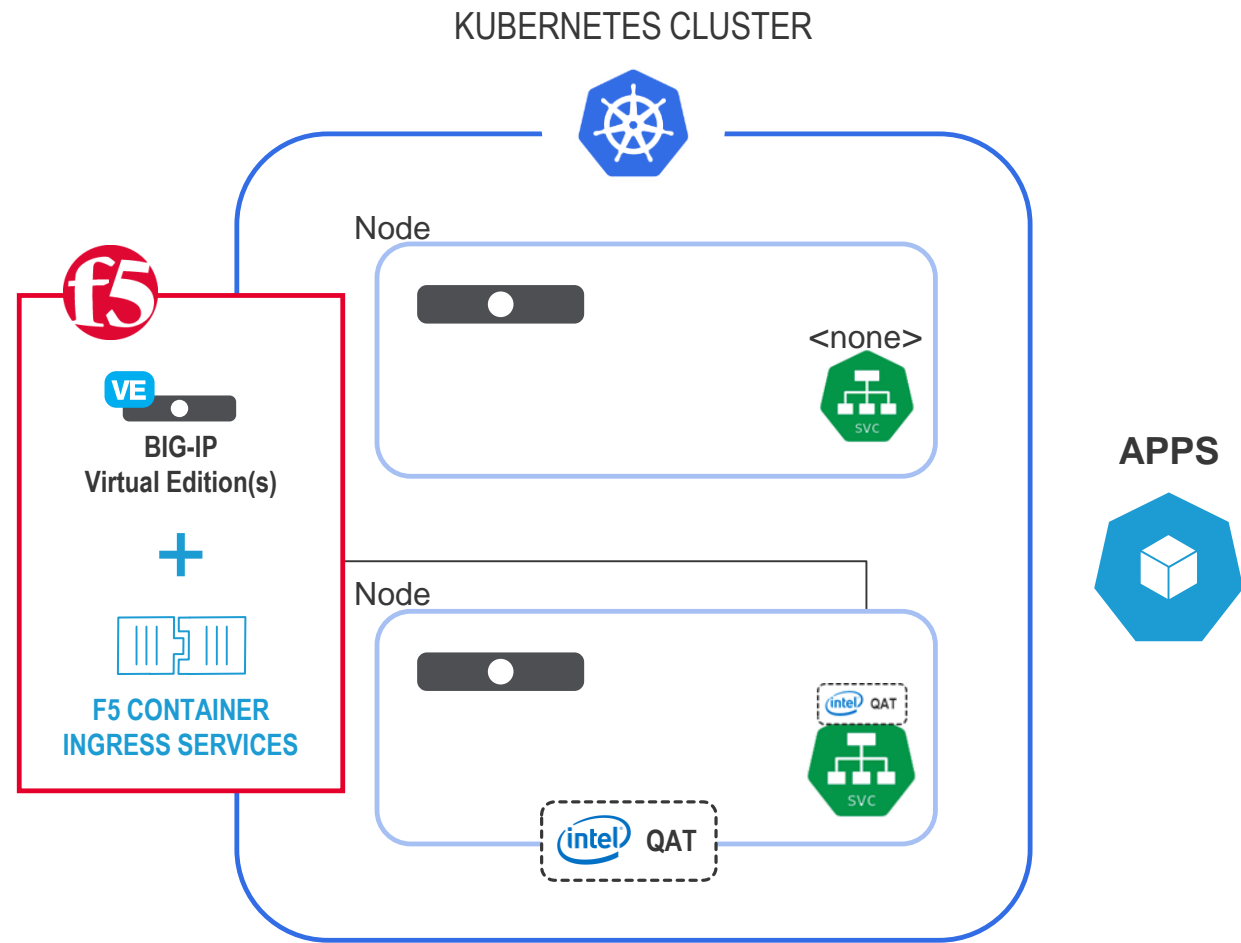
Ingress examples

Security (PAC, DDOS)

Encrypt/Decrypt

Traffic steering

Backend application pods
are deployed with no
particular affinity.



Intel® EPA and Cloudify® Orchestration – Setup

Ingress examples

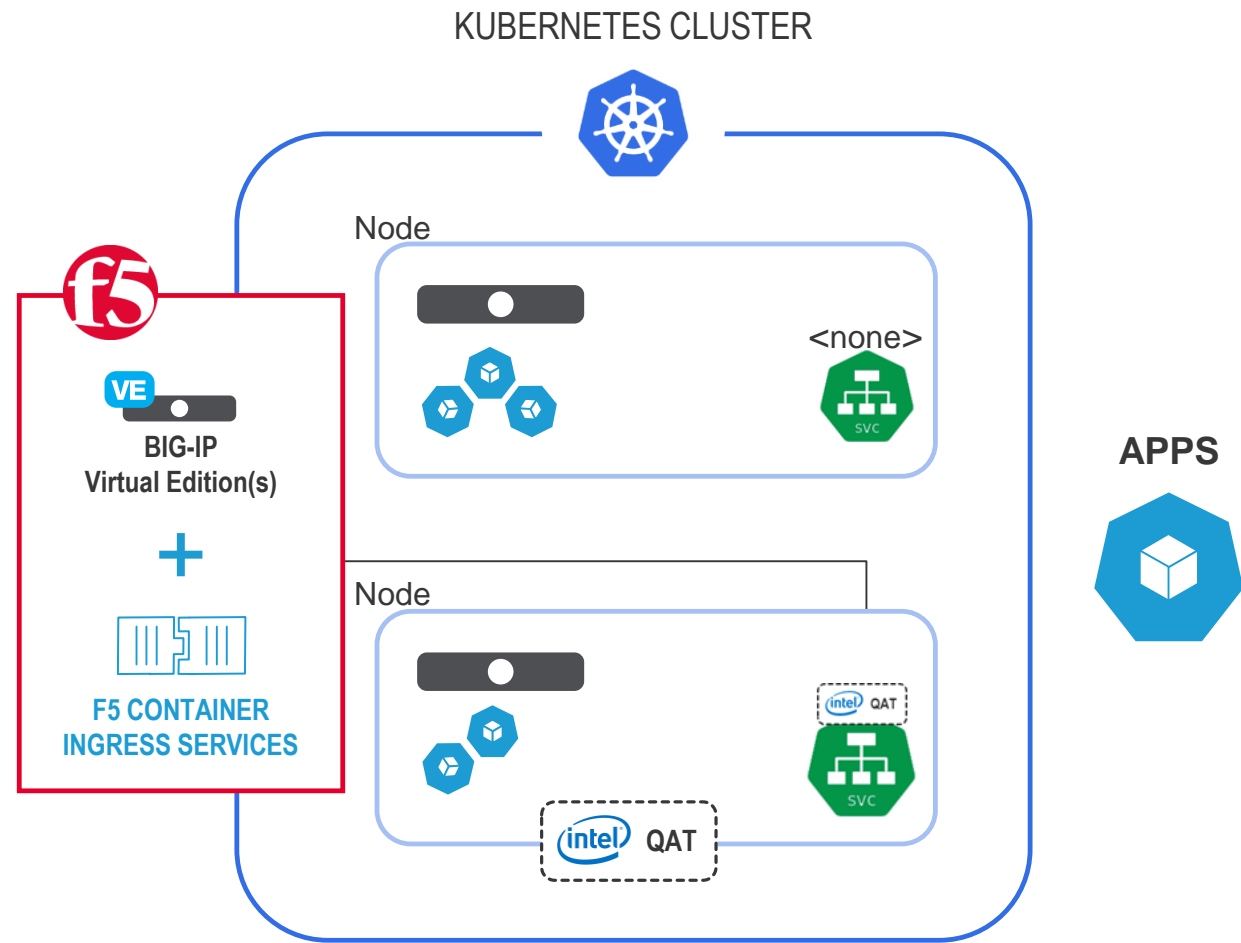
Security (PAC, DDOS)

Encrypt/Decrypt

Traffic steering

Services have self-sorted
to be on optimized nodes.

Now the cluster is
ready for traffic.



F5 Use Cases: Ingress security

F5 Container Ingress Services Leveraging Intel® Programmable Acceleration Card (PAC) to Optimize Security

Intel® EPA and Cloudify® Orchestration – Setup

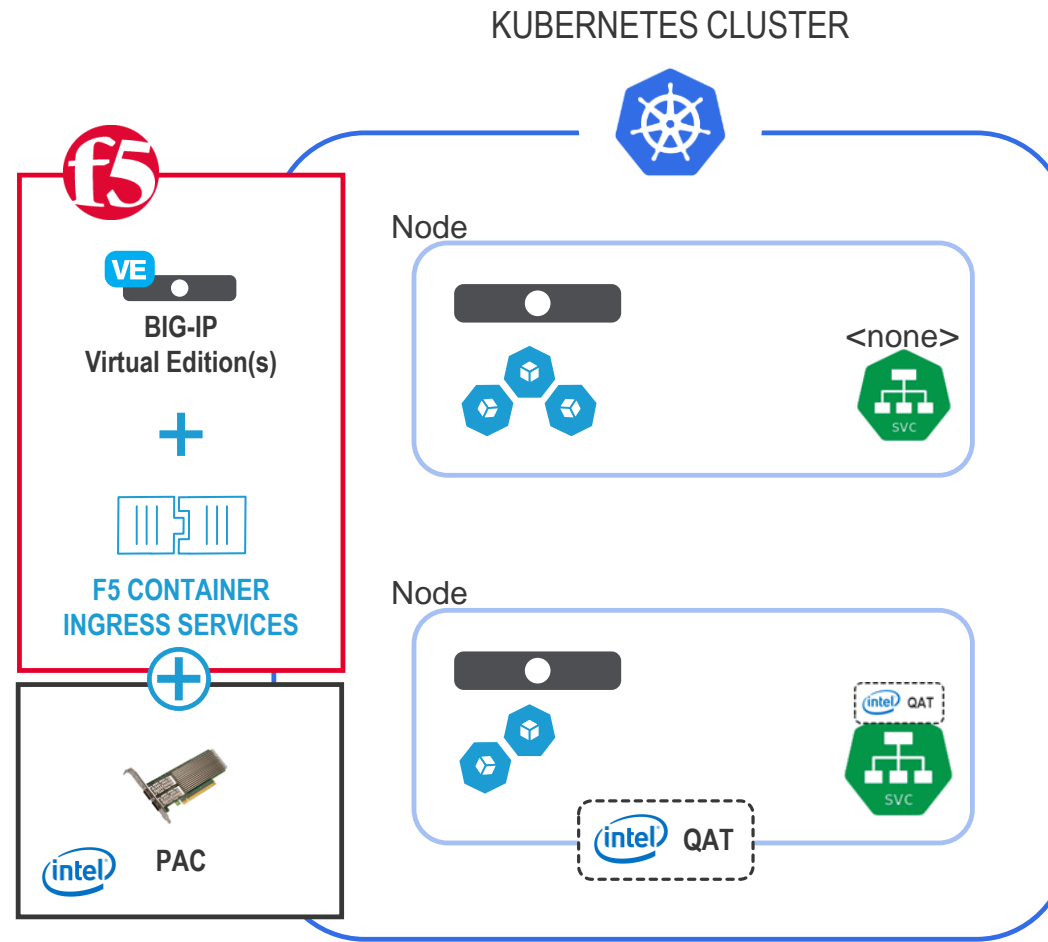
Ingress examples

Security (PAC, DDOS)

Encrypt/Decrypt

Traffic steering

In this scenario, we are using BIG-IP Virtual Edition on a node with a SmartNIC installed.



Intel® EPA and Cloudify® Orchestration – Setup

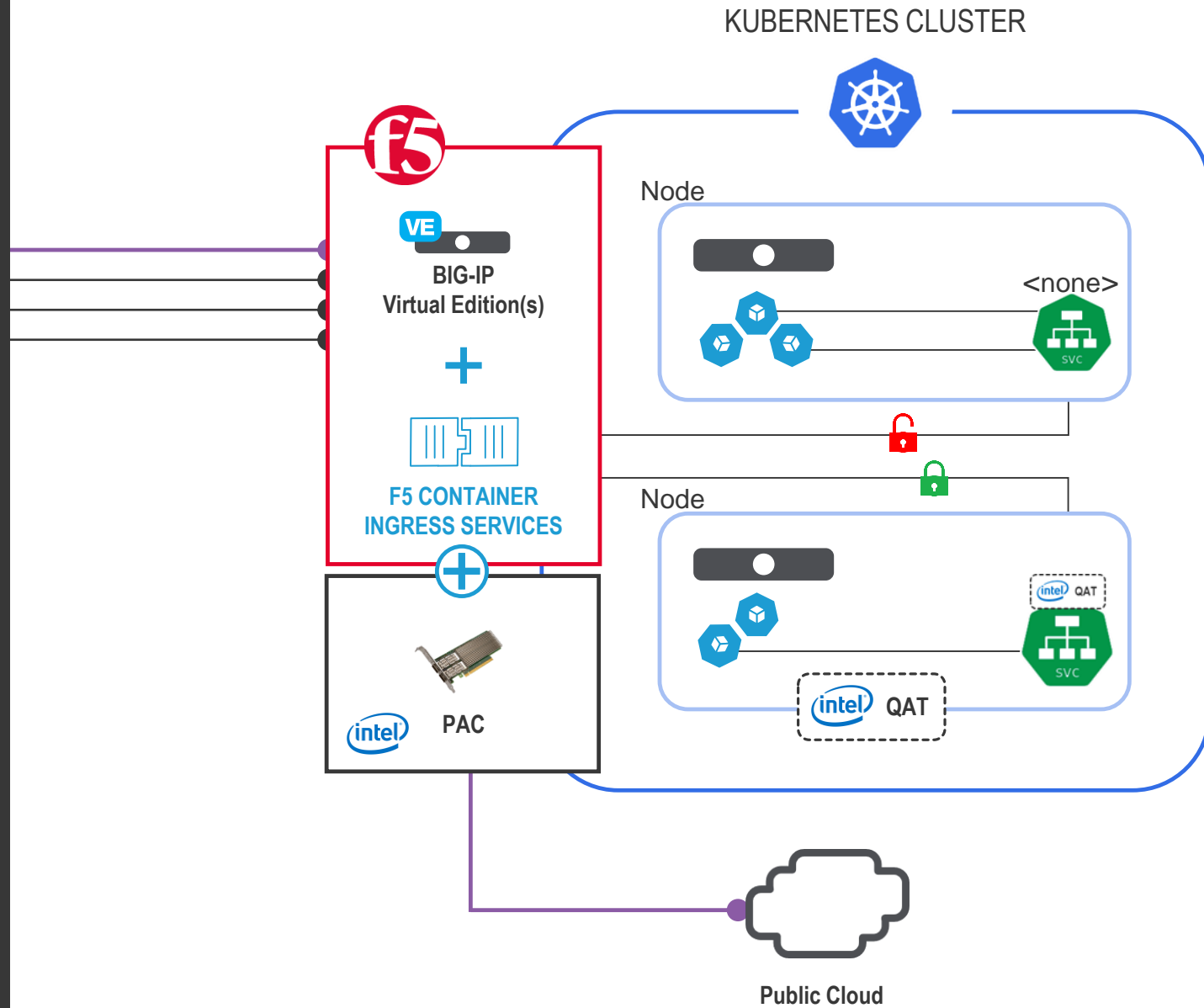
Ingress examples

Security (PAC, DDOS)

Encrypt/Decrypt

Traffic steering

Legitimate application
traffic arrives and is
directed to the correct
service in K8s or in
another cloud.



- Encrypted
- Unencrypted

Intel® EPA and Cloudify® Orchestration – Setup

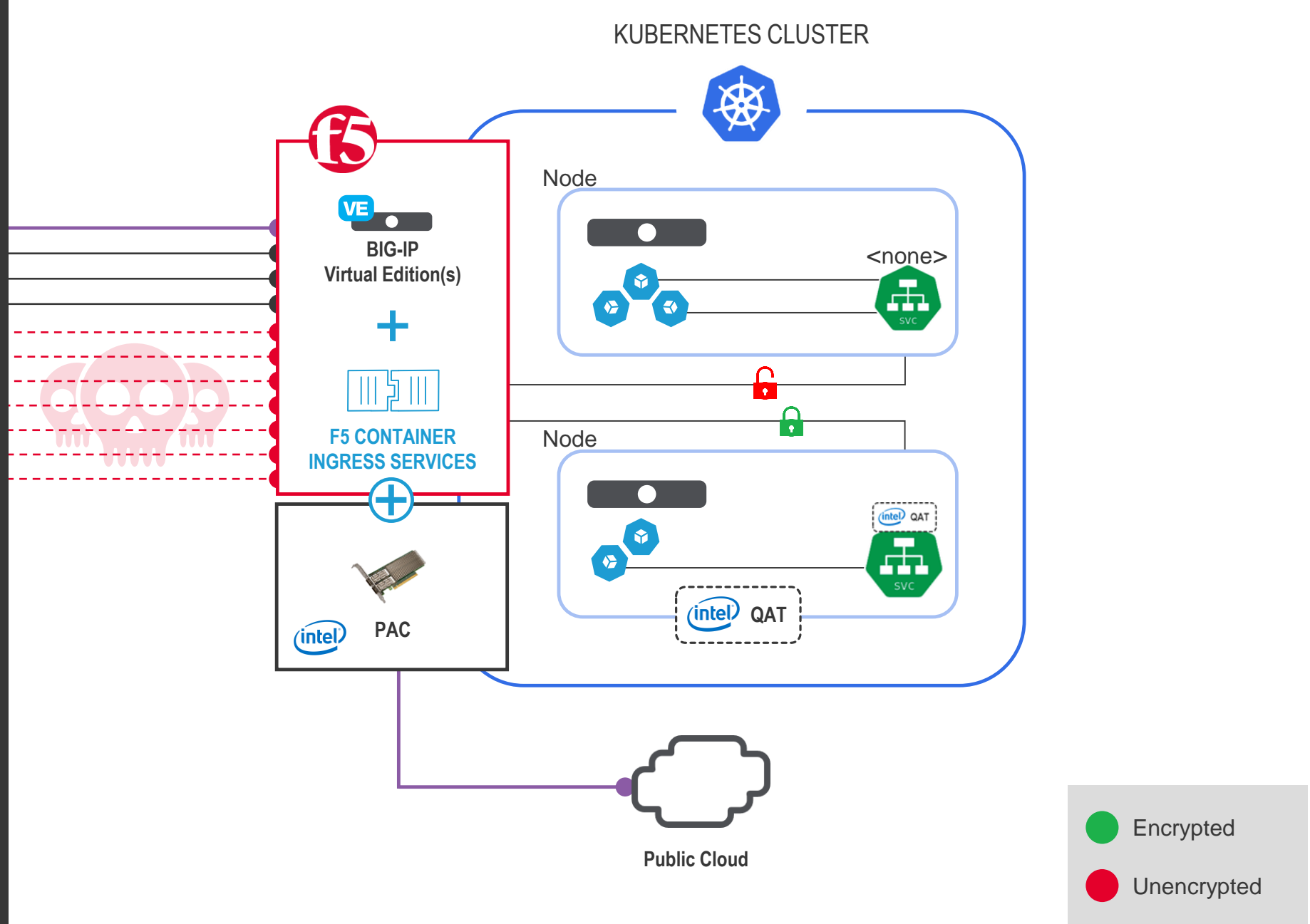
Ingress examples

Security (PAC, DDOS)

Encrypt/Decrypt

Traffic steering

A volumetric/DDoS
attack arrives and is
identified as an attack.



Intel® EPA and Cloudify® Orchestration – Setup

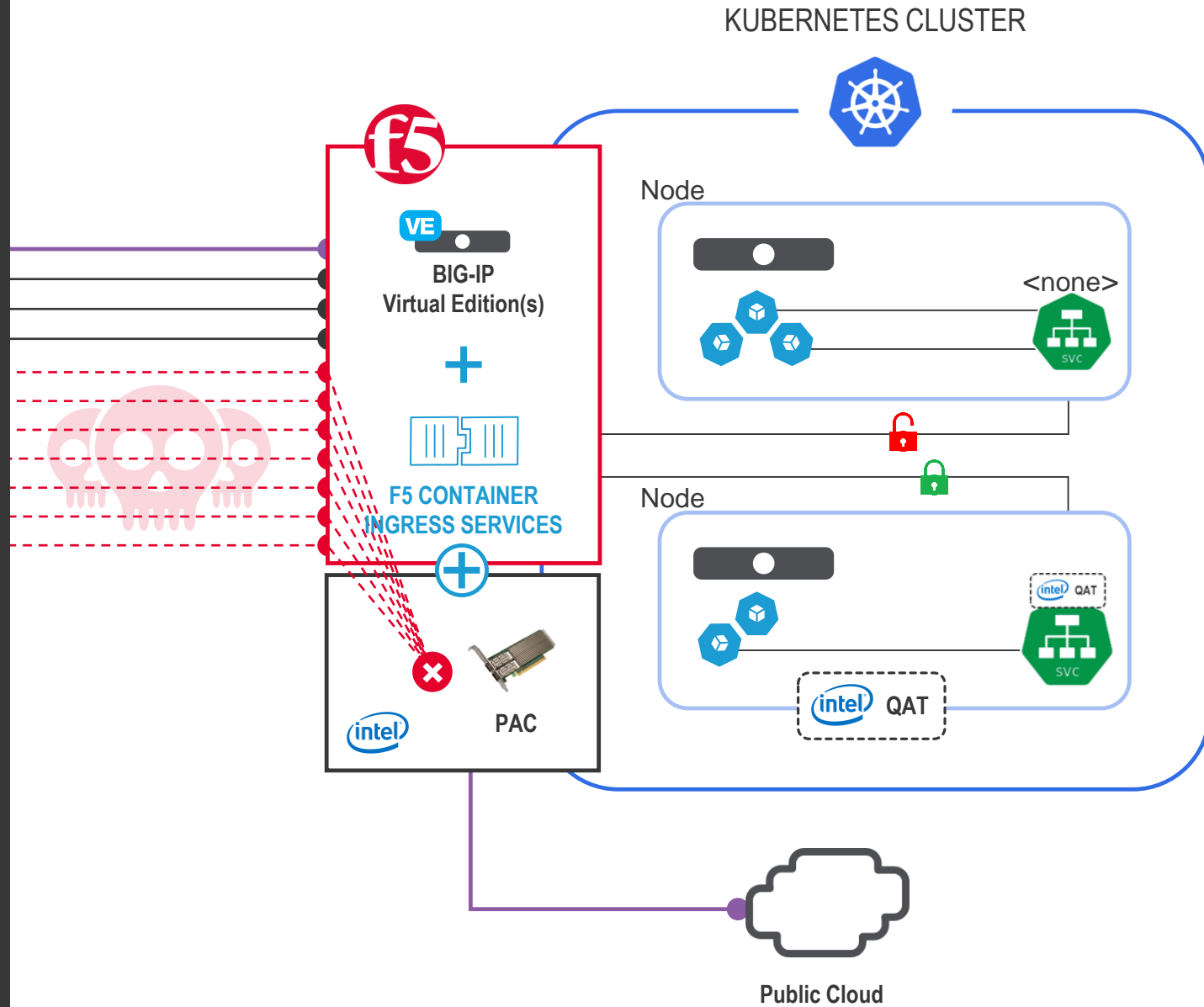
Ingress examples

Security (PAC, DDOS)

Encrypt/Decrypt

Traffic steering

The attack traffic is accelerated onto the Intel® Programmable Acceleration Card (PAC) where it is either dropped or redirected to a scrubber



-  Encrypted
-  Unencrypted

Intel® EPA and Cloudify® Orchestration – Setup

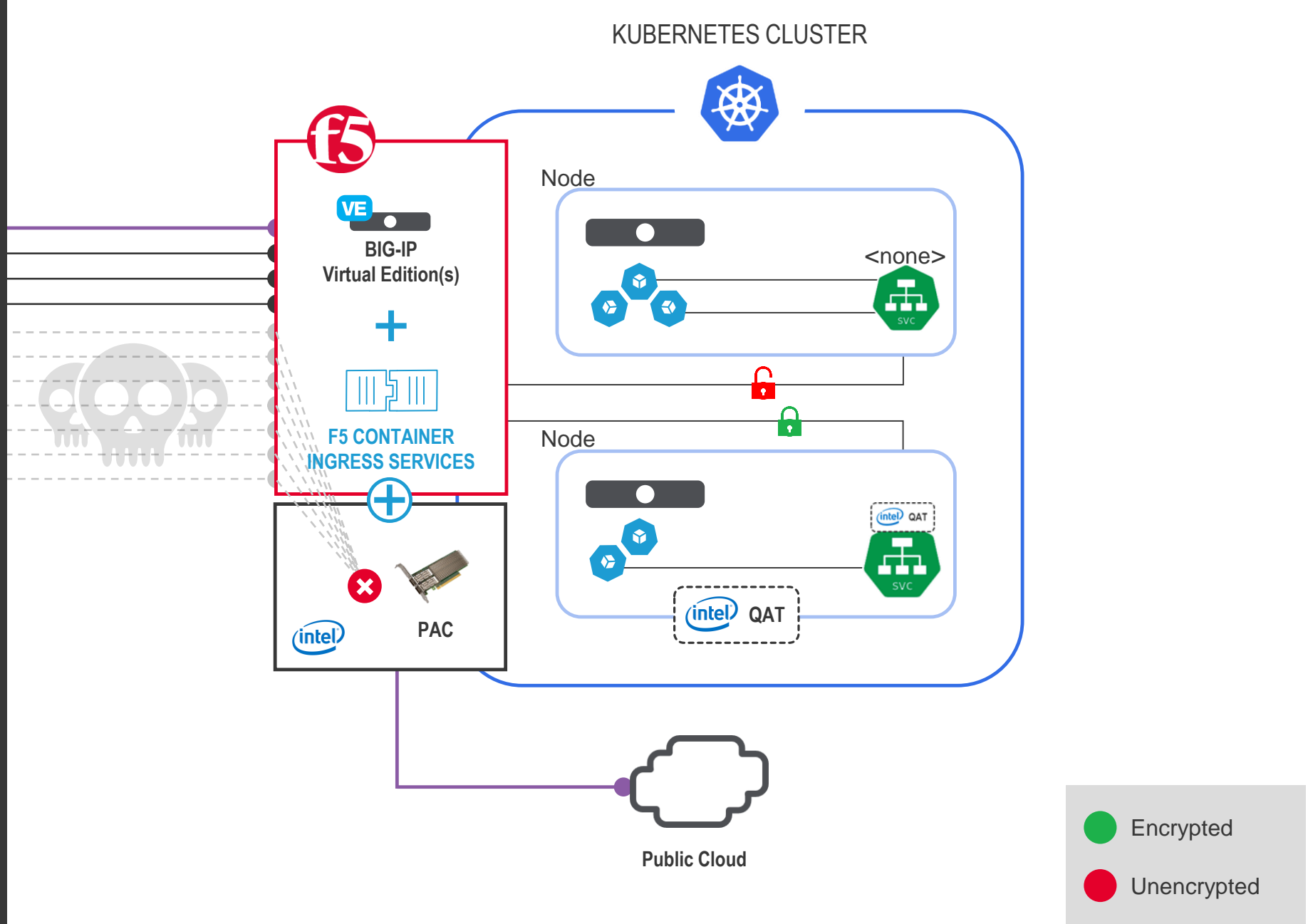
Ingress examples

Security (PAC, DDOS)

Encrypt/Decrypt

Traffic steering

Normal traffic
continues to flow.



F5 Use Cases: Ingress encryption/decryption offload

F5 Container Ingress Services Leveraging Intel® Quick Assist Technology (QAT)
to Optimize Encryption and Decryption

K8s Node Labels

COTS/COTS+ matchmaking

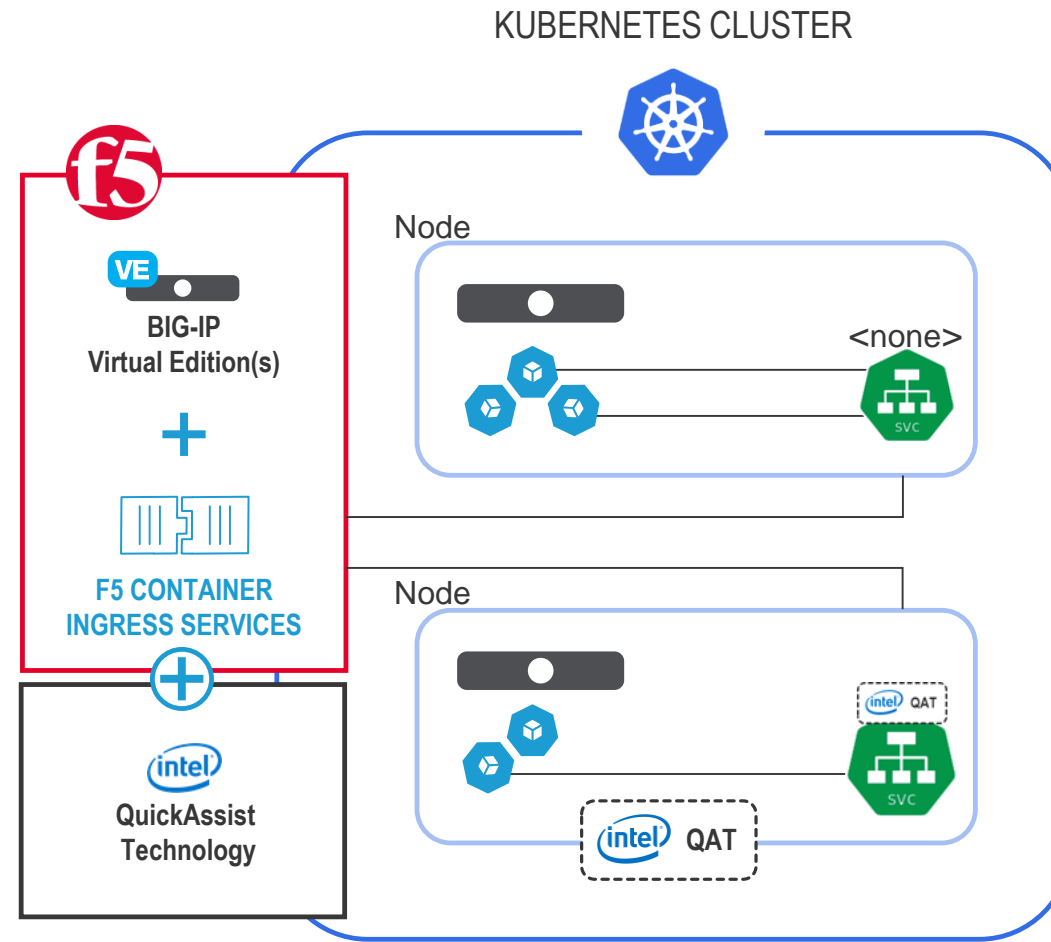
Ingress examples

Security (PAC, DDOS)

Encrypt/Decrypt

Traffic steering

Deployment: virtual F5 BIG-IP running on server with Intel® QuickAssist Technology (QAT)



- Encrypted
- Unencrypted

K8s Node Labels

COTS/COTS+ matchmaking

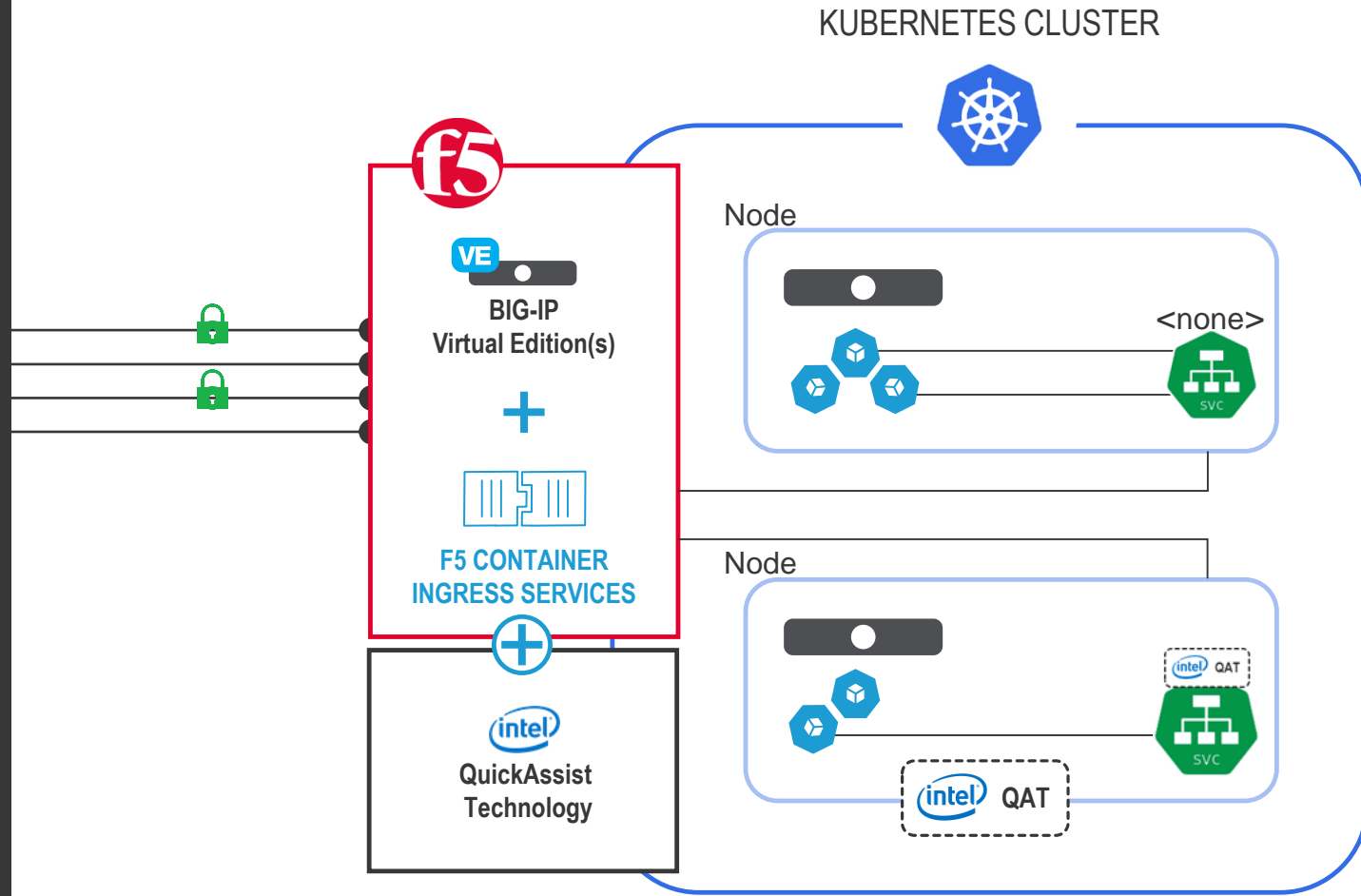
Ingress examples

Security (PAC, DDOS)

Encrypt/Decrypt

Traffic steering

Traffic decrypted using Intel® QAT offload



- Encrypted
- Unencrypted

K8s Node Labels

COTS/COTS+ matchmaking

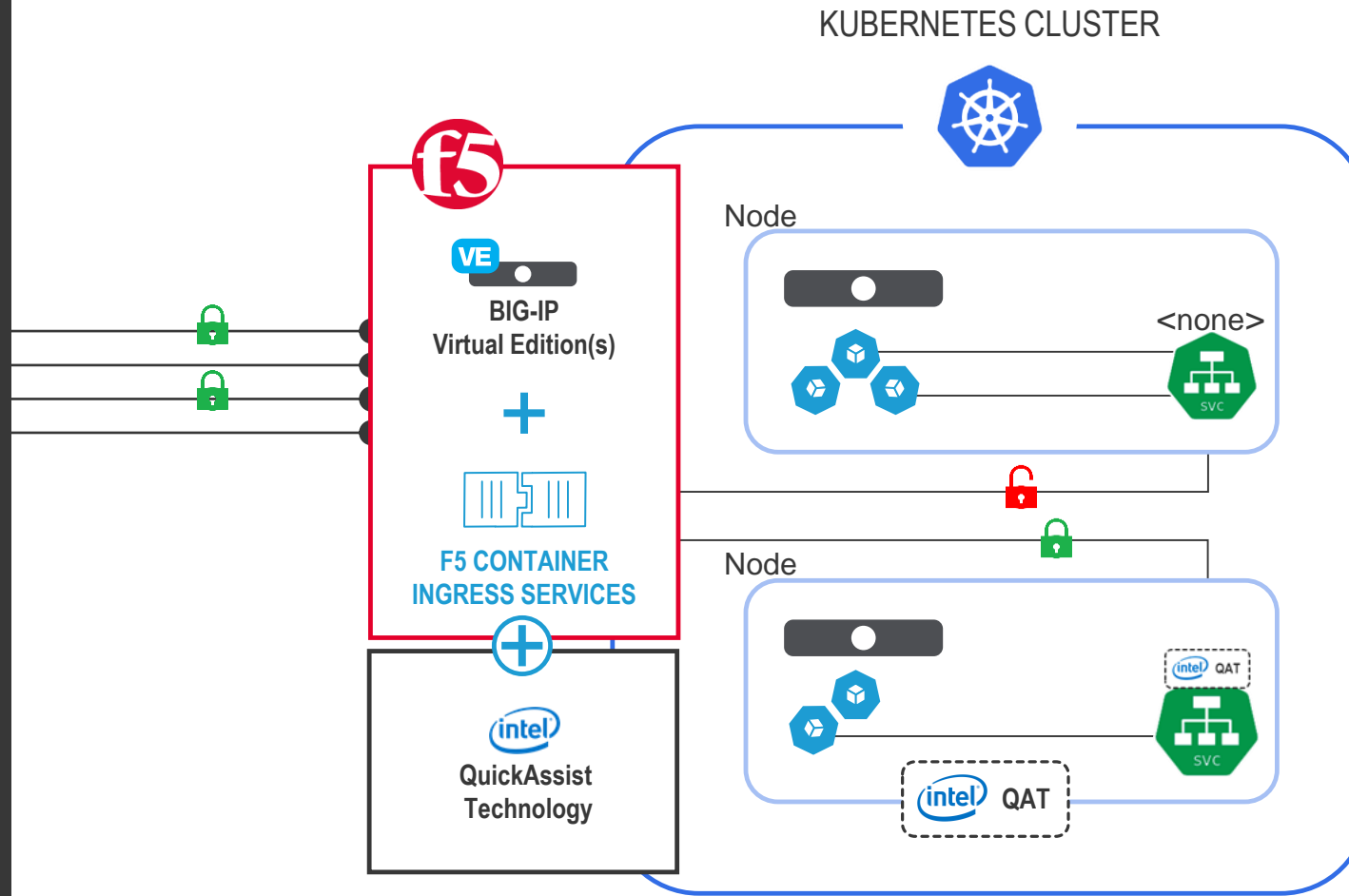
Ingress examples

Security (PAC, DDOS)

Encrypt/Decrypt

Traffic steering

Unencrypted and re-encrypted traffic sent to services within K8s



-  Encrypted
-  Unencrypted

K8s Node Labels

COTS/COTS+ matchmaking

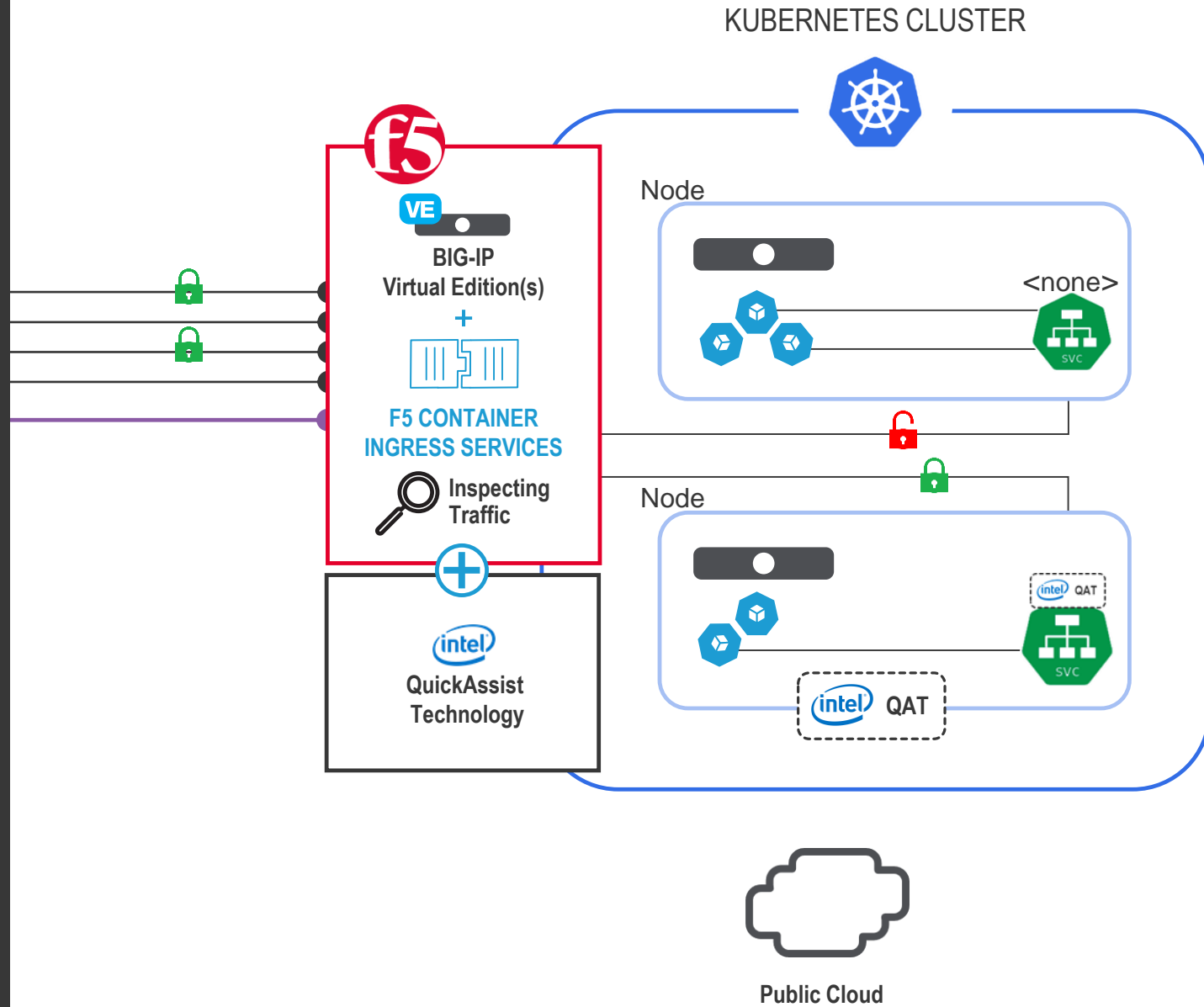
Ingress examples

Security (PAC, DDOS)

Encrypt/Decrypt

Traffic steering

Decrypted traffic identified for steering to another destination in Public Cloud



- Encrypted
- Unencrypted

K8s Node Labels

COTS/COTS+ matchmaking

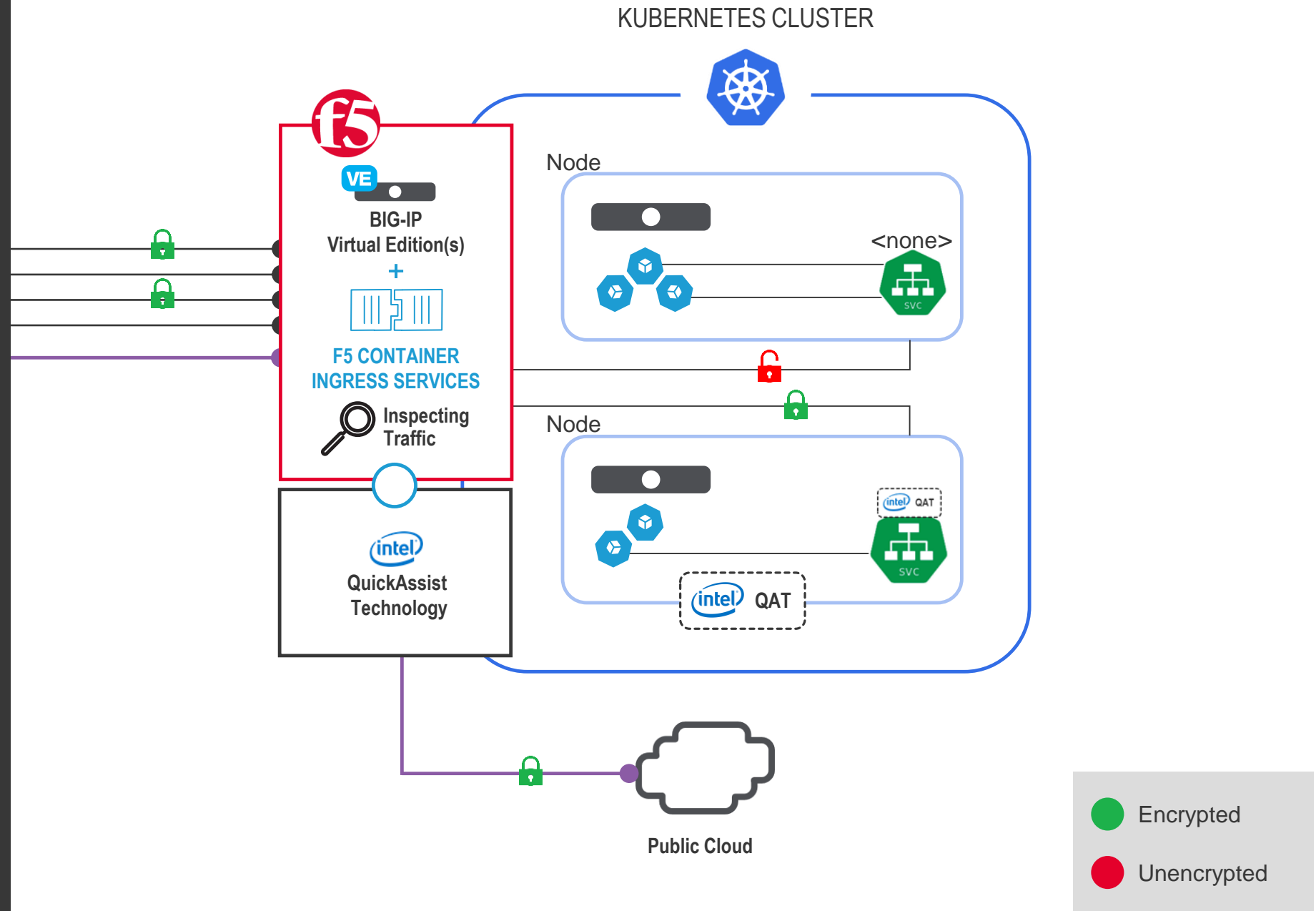
Ingress examples

Security (PAC, DDOS)

Encrypt/Decrypt

Traffic steering

Traffic re-encrypted
(using Intel® QAT
offload) and sent to
another destination in
Public Cloud



Visit us to see demonstrations at:

Intel booth B14



F5 Networks booth C6



Legal Disclaimers

- Copyright © Intel Corporation. All rights reserved.
- Other names and brands may be claimed as the property of others.
- Intel, the Intel logo and other Intel Marks are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries.
- No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.
- Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

