Using Cisco Network Service Orchestrator to Automate the Containerized Deployment of Cisco Prime Network Registrar

Contents

Bias-free documentation policy

Cisco follows a bias-free documentation policy. According to this policy, Cisco treats all persons with respect—regardless of race, color, ancestry, national origin, age, sex, citizenship, veteran status, marital status, sexual orientation, physical or mental ability, religious creed, or medical condition. Language or graphic elements that offend others violate our business philosophy and our company policy.

Abstract

As a software application, Cisco® Prime Network Registrar (PNR) can be deployed in various environments. Recently, Cisco PNR added Docker Containers as a deployment option. As deployments of PNR continue to grow, so does the need for automation. Cisco's Containerized Network Function Orchestrator (CNFO) empowers customers to automate across complex, high-scale, multivendor environments. This white paper describes how you can utilize CNFO to automate the deployment of multiple PNR nodes, including initial provisioning of the nodes to make them operational.

Audience

The intended audience of this document includes but is not limited to IT architects, sales engineers, field consultants, professional services, IT managers, partner engineering, and customers who want to take advantage of the benefits of orchestrating the PNR containerized services by using CNFO Core Function Pack (CFP).

Purpose

Cisco PNR automation provides a well-structured and strategic way to a fast, effective, and reliable deployment process. It allows you to streamline processes and deliver with significant operation efficiency. This document highlights the benefits of automating PNR service deployments by orchestrating it with CNFO CFP to provide a single unified interface to configure, manage, and monitor the PNR services. The document considers a typical PNR DNS use case to provision a regional pod and create an authoritative DNS pod and caching/recursive DNS pod.

Cisco Prime Network Registrar - overview

Cisco PNR is a scalable, high-performance, and extensible solution that provides services for Dynamic Host Configuration Protocol (DHCP), Domain Name System (DNS) acting as an Authoritative DNS (ADNS), and Caching DNS (CDNS). It is designed for Internet Service Providers (ISPs), Multiple Service Operators (MSOs), and Enterprises to provide policy-based, robust, and scalable DNS and DHCP services for medium to large IP networks.

A PNR can be run either as a Physical Network Function (PNF), or a Virtual Network Function (VNF), or as a Containerized Network Function (CNF). In a Containerized Network Function, a PNR pod can be either a regional pod for central management and licensing, or an authoritative DNS pod for persistent storage of DNS zones and Resource Records (RRs), or a caching/recursive DNS pod to traverse multiple DNS authoritative servers to resolve client queries.

Containerized Network Functions Orchestration – overview

The Containerized Network Functions Orchestrator ties CNFs into Cisco's Network Service Orchestrator (NSO). NSO and CNFO offer the ability to handle high levels of complexity and scale from one place. CNFO/NSO is a normalizing tool; it can push the same service out across multiple vendors and device types at scale. CNFO and NSO abstract away the need for a user to interact with or understand how each individual device functions.

Solution overview

In a container environment, it is desirable to automate the PNR cluster deployment to bring up the cluster quickly and easily and to be able to use an active container immediately.

Manually configuring the PNR pod deployment parameters, such as the ones listed below to enable a service (for example, a DNS), can be time consuming.

1. Bring up a PNR regional pod by K8s yaml file, add licenses to it, and set up user credentials for the pod.
2. Bring up the PNR ADNS pod on K8s and enable the DNS service on the ADNS pod.
3. Add user credentials to the ADNS server along with CPNR regional IP and port information for licensing.
4. Bring up the PNR CDNS pod on K8s and enable the DNS service on the CDNS pod.
5. Add user credentials to the CDNS server along with PNR regional IP and port information for licensing.
6. Set up the bootstrap (day 0) configuration for ADNS and CDNS pods.
7. Apply the day 1 configuration to create zones and provide values separately for the ADNS pod and CDNS pod deployments.

The deployment process provides opportunities to automate such manual tasks. Orchestrating PNR by using CNFO CFP proves as an integrated solution to automate and capitalize on the resources, optimize time utilization, and provide a unified interface to monitor and manage the PNR pod deployments. This automation helps in faster service deployment, increased flexibility, and improved application performance by being able to respond dynamically to any changes or events. PNR automation helps to easily manage customization complexities in deployments.

The automation process uses NSO as the primary orchestration engine to bridge the gap in conventional management of DNS. NSO orchestrates physical network functions, virtual network functions through NFVO, and now the containerized network functions through CNFO CFP.

Using NSO CNFO CFP, a customer can provision a typical PNR DNS use case to instantiate a PNR Regional pod, an Authoritative DNS, and a Caching/Recursive DNS. Once instantiated, the NSO NED for PNR helps to automate the configurations to bring these three containers into service.

The NSO platform provides a seamless user experience with device agnosticism and the ability to orchestrate at a scale. For a simplified and efficient user experience, all these various devices are managed through a single unified interface.

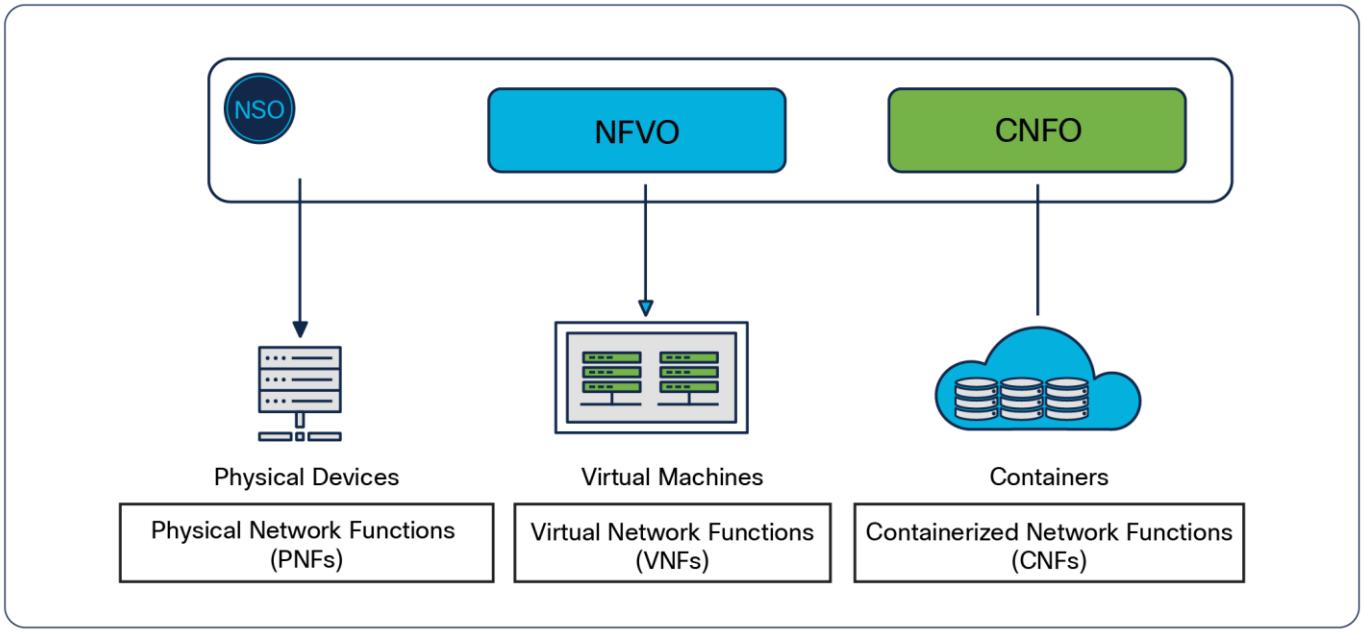

Figure 1.
With CNFO, NSO can orchestrate physical, virtual, and cloud-native network functions

Solution benefits

Automating PNR deployment by using NSO CNFO CFP provides the following benefits:

- **End-to-end orchestration:** The Cisco PNR CNFO CFP integration automates the initialization, configuration, and deployment of PNR DNS services via CNFO CFP. This integration provides an intelligent approach to a unified view of end-to-end user experience to deploy, manage, and troubleshoot the PNR pods.

  From CNFO CFP, you can deploy CNFs and perform CRUD (Create, Read, Update, Delete) operations on PNR devices. A sample custom package is used to deploy the DNS services. CNFO CFP helps to configure the PNR devices from multiple locations. The automation also supports updates to PNR via CNFO CFP to override Helm values and Day n configurations for the deployments.

  This automation also extends and accommodates multivendors to respond efficiently to various market requirements.

- **Orchestration at scale:** CNFO CFP can deploy multiple instances of DNS in a single transaction from NSO to achieve the desired scale. The configurations are templatized and a sync to is performed to restore and apply the DNS configurations.

- **Lifecycle Management:** The Cisco PNR CNFO CFP integration automates the stages of the PNR lifecycle management. It automates the initial configuration, maintenance, and retiring or removing of the services. It offers greater insight to the data quickly and real-time monitoring of the services and allows you to adapt easily to varying business requirements, while at the same time offers control and security over the deployment process.

The PNR automation abstracts away the complexity of manually configuring the PNR deployment parameters (such as day 0 and day 1 configurations) and performs the following tasks:

1. Deployment of a PNR regional pod initialized with licenses and administrator account.
2. Deployment of a PNR ADNS pod and CDNS pod initialized with administrator account, Day-0 configuration and Day-1 configuration, and setup of the regional IP.

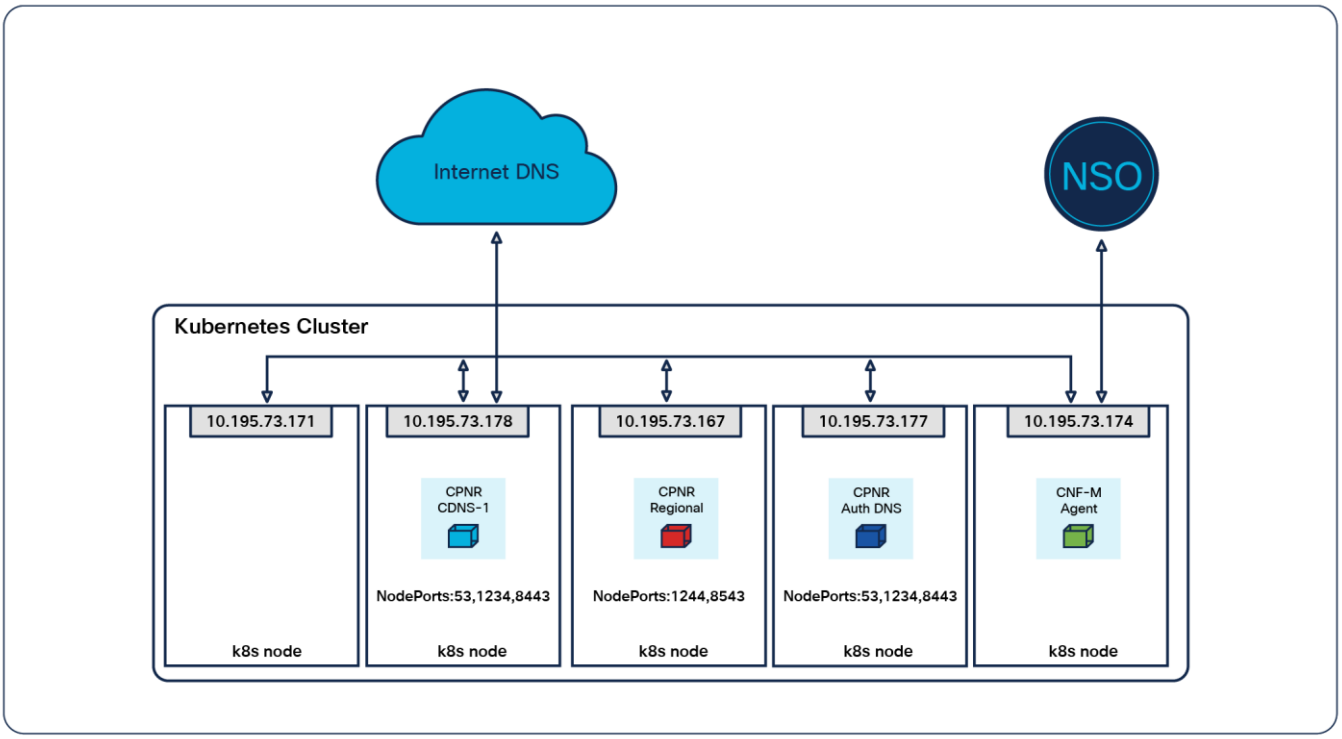

Figure 2.
Lab setup for CNFO PNR Kubernetes Cluster

After the deployment, push day-2 DNS configuration on PNR ADNS and CDNS containers.

When a PNR DNS service is deployed, you can view the status of the PNR service deployments from CNFO CFP. Specific operational commands can be used to obtain operational information for the cluster. For example, you can override the initial Helm values during a create operation, or update image tags or resources. Once the device is onboarded, you can perform any day 1 or day n configurations and monitor the status of the DNS services from the CNFO CFP operational data. Execute DNS queries to verify if the PNR cluster is functional.

Solution implementation

The following CNFO CFP bootstrap information is required to begin the automation process.

1. Deploy the Containerized Network Function Manager (CNFM) and add it as a device to the NSO device tree.
2. Register the PNR Helm repository with CNFM.
3. Configure the K8s clusters.
4. Generate CNF Descriptors (CNFDs) from PNR Helm charts to deploy the PNR cluster.
5. Create the key-value pair in the CNFD for the service account.

The automation process internally uses a PNR NED and Helm charts that package components required for pod deployments. The Helm charts comprise details about the cluster and the required deployment types. These Helm charts inject some of the initial configurations. The Helm chart comprises the following attributes to deploy the PNR cluster:

1. Support of single as well as multinode PNR deployments
2. Ability to enable or disable persistent storage [**mountPath: /var/nwreg2/<local/regional>**]
3. Support of **postStart** lifecycle hook to inject initial configuration script
4. Environment variables are loaded with defaults
5. **NodePort** service can access the pod from external network

CNFD leverages the configuration details from the Helm chart. A PNR payload references the CNFD and contains details of the cluster on which to deploy the PNR pod. The PNR service internally instantiates a CNF-instance on CNFO CFP. The CNF manager in the CNFO core function pack manages the orchestration. The CNF instance service uses the PNR NED to onboard the PNR device onto the NSO device tree.

Once the initial configuration for day 0 is complete and the PNR is onboarded onto the NSO device tree, PNR service creates an auth DNS zone. Resource Records (RR) are then added to the zone, and a mapping to this zone is created from the CDNS service. A corresponding CDNS exception object is created on the CDNS container linking the CDNS service to the ADNS service.

Automation hardware and software requirements

This section outlines the hardware and software requirements to automate PNR deployments.

The following components are used in the orchestration of PNR and CNFO CFP:

- **PNR regional container:** This is the docker container for the PNR 11.1 regional service.
- **PNR local container:** This is the docker container for PNR 11.1 ADNS and CDNS services.
- **PNR Helm chart:** This is the K8s Helm chart to package PNR containers, YAML files, initialization scripts, and day 0 configuration values.
- **cisco-cnfo:** This CNFO CFP package includes core logic to deploy a PNR Helm chart by using a CNFM pod running on a K8s cluster. The package communicates with the CNFM pod via a Netconf protocol. The operational data provides the status of each service deployment.
- **cisco-pnr_rest-gen-3.6:** This PNR NED package is used to communicate with the PNR pod device via REST protocol. Once the PNR pod device is ready and onboarded on NSO, this package is used to configure day 1/day n configurations on the PNR pod.
- **cnfm-nc-1.1:** NSO uses this Netconf NED package to communicate with the cnfm pod device.
- **PNR-service:** This PNR service package is an optional sample package that includes the minimum required logic to deploy the PNR cluster on K8s. This customizable package acts as a Northbound service and sits on top of CNFO CFP to automate the PNR cluster deployment. The PNR service package configures and pushes the configurations on PNR pods. It uses the cnf-instance from CNFO CFP to deploy the pod. CNF instance service takes care of onboarding the PNR devices onto the NSO device tree. CNFO CFP uses the PNR Helm chart to deploy the PNR pod on a Kubernetes cluster. The Helm chart can be updated with different values to deploy specific configurations for the cluster.

The following table shows the software versions required for the PNR CNFO CFP integration:

Table 1. Software versions required for the PNR CNFO CFP integration

Component	Version
Cisco Prime Network Registrar	v11.1
Cisco NSO Containerized Network Functions Orchestration Core Function Pack	v1.1.0
Cisco Network Service Orchestrator	v5.7.5
ConfD	v7.5

Automation Workflow

The following diagram shows how CNFO CFP is used to automate the PNR cluster deployment process.

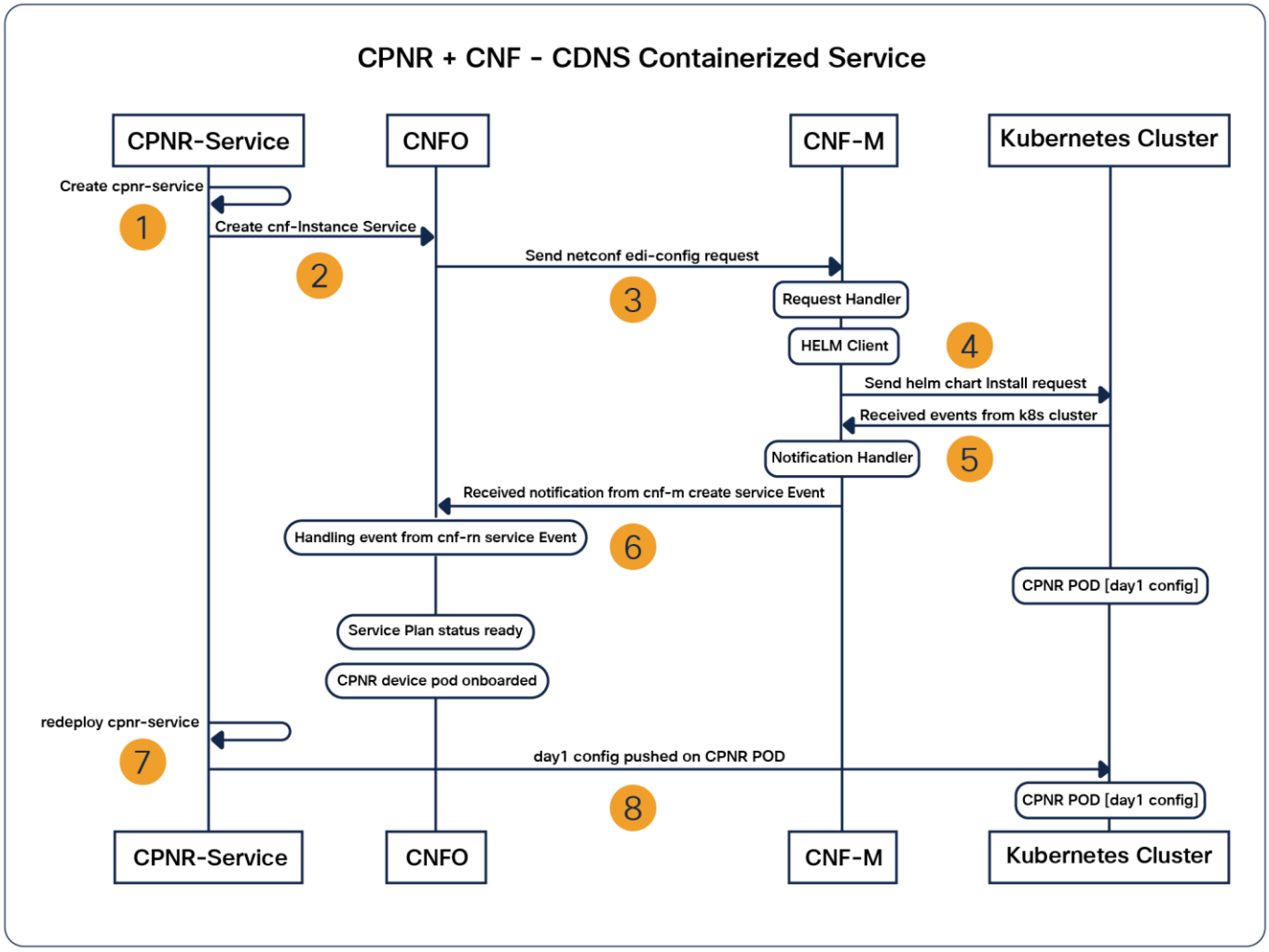

Figure 3.
CNFO Call Flow

A PNR service is a top-level service sitting on top of NSO. When you create a PNR service, the service internally creates a CNF-instance service, which orchestrates the PNR deployment on K8s cluster by using CNFM. CNFO CFP uses core logic to orchestrate the PNR CNFs on the K8s cluster.

The CNF-instance helps to onboard the PNR as a device onto CNFO to deploy the PNR pod on K8s.

Subsequently, you can use CNFO to manage and monitor the lifecycle of the PNR service through NSO.

The CNFO operational data displays the status of the PNR service.

Use case

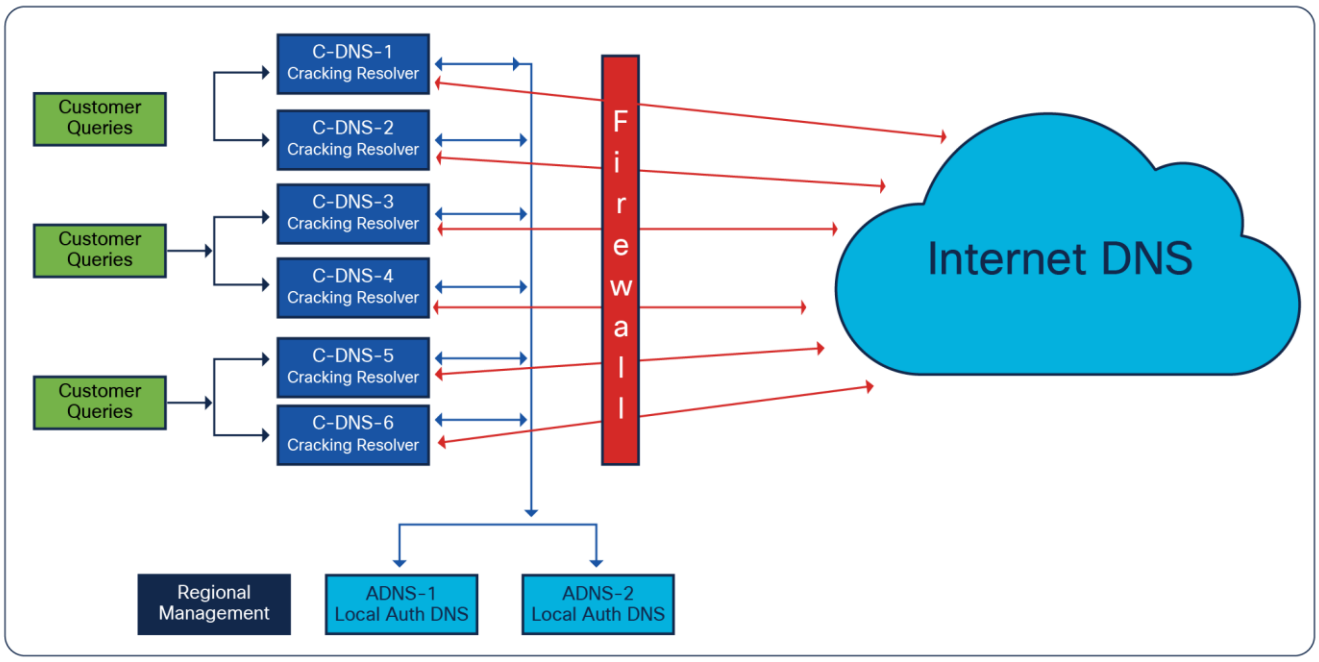

Figure 4.
PNR DNS Use Case

A typical use case for PNR DNS in Mobile Service provider networks is to provide UE internet access via PNR CDNS as well as access to internal infrastructure DNS data via ADNS. This same setup is often replicated repeatedly at each service provider's site, making it a significant reason for automation.

The use cases discussed in this topic are specific to DNS to deploy the PNR pod. The following are the three components used in this DNS use case.

A. Regional pod (controller)

B. Authoritative DNS

C. Caching/Recursive DNS

The PNR service allows CRUD operations on the PNR cluster.

- Create the PNR service to deploy the PNR CNF (by installing the Helm chart).
- Delete the PNR service to undeploy the PNR CNF (by uninstalling the Helm chart).
- View the plan, cnf-result, and notifications.

The CNFO CFP operational commands for K8s and Helm help to view and verify the deployment status of the cluster.

- kubectl: get|logs|exec
- helm: test|list|dry-run

  CNFO CFP automates the tasks listed in the use cases below.

A. Automating the PNR regional pod deployment

The PNR regional pod deployment automation:

1. Selects cluster worker nodes to deploy the regional pod.
2. Sets up a regional pod (or VM).

Note: The default CCM/SCP port for regional is 1244.

3. Sets the access credentials username and password for the regional pod to be deployed.
4. Loads the license file to the regional pod.

B. Automating the PNR authoritative DNS pod deployment

The PNR authoritative DNS pod deployment automation:

1. Selects the cluster worker node to deploy the ADNS pod.
2. Sets up a PNR local ADNS pod and starts it.
3. Sets the access credentials username and password for the ADNS pod to be deployed.
4. Sets the regional pod IP. Edit the **/var/nwreg2/local/conf/cnr.conf** file to update the following values for regional IP address, port, and the type of service used in the deployment:
   - cnr.regional-ip=<IPv4-Address>
   - cnr.regional-ccm-port=<port_number>
   - cnr.services=<type_of_service>
5. Restarts the PNR service to register with the regional cluster and license the instance to enable the DNS services.
6. Creates configuration for the auth DNS zone and the RRs.
7. Adds an entry pointing to the DNS server itself and to test the RRs.

C. PNR caching DNS pod deployment

The PNR caching DNS pod deployment automation:

1. Selects the cluster worker node to deploy the CDNS pod.
2. Sets up a PNR CDNS pod and starts it.
3. Sets the access credentials username and password for the CDNS pod to be deployed.
4. Sets the regional cluster IP. Edit the **/var/nwreg2/local/conf/cnr.conf** file to update the following values for regional IP address, port, and the type of service used in the deployment:
   - cnr.regional-ip=<IPv4-Address>
   - cnr.regional-ccm-port=<port_number>
   - cnr.services=<type_of_service>
5. Restarts the PNR service to register with the regional cluster and license the instance to enable the DNS services.

6. Creates a configuration for an exception that points the CDNS server to the DNS server and inputs the zone on ADNS.
7. Adds an entry pointing to the DNS server itself and to test the RRs.

Conclusion

Automated deployment provides a proven and mature way to configure and maintain the deployment process. It increases productivity with superior customer experience and reduces delays in the process. This automation for speed and the ability to monitor and manage efficiently proves to be a significant advantage in PNR pod deployments.

The CNFO CFP provides a validated approach to automate PNR deployment in the network infrastructure. Automation optimizes the way the PNR pods are deployed. CNFO CFP helps to get the pods loaded and onboarded quickly and easily. With this automation approach, customers now have the ability to orchestrate the PNR pod on the K8s cluster and configure the PNR components to be deployed, so the cluster is ready to serve the DNS requests from the clients in minimal time and with minimal effort. It provides the benefit of monitoring, orchestration, workload optimization, and lifecycle management of the PNR CNF pod through NSO CNFO CFP. A deployment plan displays the deployment status.

References

The following documentation can be found on the Cisco website.

1. Cisco Network Service Orchestrator documentation
2. Cisco Prime Network Registrar documentation
3. Cisco Containerized Network Functions Orchestration Core Function Pack documentation

Americas Headquarters
Cisco Systems, Inc.
San Jose, CA

Asia Pacific Headquarters
Cisco Systems (USA) Pte. Ltd.
Singapore

Europe Headquarters
Cisco Systems International BV Amsterdam,
The Netherlands

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco Website at **https://www.cisco.com/go/offices**.

Printed in USA C11-3363304-00 01/23