NetScaler CPX 13.1

Machine translated content

Disclaimer

La versión oficial de este contenido está en inglés. Para mayor comodidad, parte del contenido de la documentación de Cloud Software Group solo tiene traducción automática. Cloud Software Group no puede controlar el contenido con traducción automática, que puede contener errores, imprecisiones o un lenguaje inadecuado. No se ofrece ninguna garantía, ni implícita ni explícita, en cuanto a la exactitud, la fiabilidad, la idoneidad o la precisión de las traducciones realizadas del original en inglés a cualquier otro idioma, o que su producto o servicio de Cloud Software Group se ajusten a cualquier contenido con traducción automática, y cualquier garantía provista bajo el contrato de licencia del usuario final o las condiciones de servicio, o cualquier otro contrato con Cloud Software Group, de que el producto o el servicio se ajusten a la documentación no se aplicará en cuanto dicha documentación se ha traducido automáticamente. Cloud Software Group no se hace responsable de los daños o los problemas que puedan surgir del uso del contenido traducido automáticamente.

Contents

Acerca de NetScaler CPX

July 22, 2024

NetScaler CPX es un controlador de entrega de aplicaciones basado en contenedores que se puede aprovisionar en un host Docker. NetScaler CPX permite a los clientes aprovechar las capacidades del motor de Docker y utilizar las funciones de equilibrio de carga y administración de tráfico de NetScaler para aplicaciones basadas en contenedores. Puede implementar una o más instancias de NetScaler CPX como instancias independientes en un host de Docker.

Una instancia de NetScaler CPX proporciona un rendimiento de hasta 1 Gbps.

Como formato en contenedores de NetScaler, NetScaler CPX se integra perfectamente en el entorno de Kubernetes y forma una parte integral de la solución nativa en la nube de NetScaler. La solución nativa en la nube de NetScaler le ayuda a crear y entregar aplicaciones de software con velocidad, agilidad y eficiencia en un entorno de Kubernetes. Con la solución nativa en la nube de NetScaler, puede garantizar la fiabilidad y la seguridad de nivel empresarial para su entorno de Kubernetes. Para obtener más información, consulte la Solución nativa en la nube de NetScaler.

En este documento se asume que está familiarizado con Docker y su funcionamiento. Para obtener información sobre Docker, consulte la documentación de Docker en https://docs.docker.com.

Funciones disponibles

NetScaler CPX admite las siguientes funciones:

- Disponibilidad de aplicaciones
  - Equilibrio de carga L4 y conmutación de contenido L7
  - Descarga de SSL
  - Traducción de protocolos IPv6
  - Equilibrio de carga de Microsoft MySQL
  - Controles de velocidad de AppExpert
  - Dirección de tráfico consciente del suscriptor
  - Protección contra sobretensiones y colas prioritarias
  - Protocolos de redirección dinámica
- aceleración de aplicaciones
  - Optimizaciones de TCP de clientes y servidores
  - Redirección de memoria caché
  - AppCompress
  - AppCache

- Seguridad de las aplicaciones
  - Reescritura y respuesta de L7
  - Defensas DoS L4
  - Defensas DoS L7
  - Firewall de aplicaciones web (WAF). NetScaler CPX admite todas las funciones WAF que se admiten en otros factores de forma de NetScaler. Para obtener información sobre las funciones de WAF compatibles, consulte NetScaler Web App Firewall.
  - Autenticación, autorización y auditoría (AAA) para el tráfico de aplicaciones
- Optimización del protocolo TCP
  - TCP de rutas múltiples
  - Control de la congestión de aumento binario (BIC) y TCP
- Capacidad de administración sencilla
  - Registro web
  - AppFlow
  - NetScaler Application Delivery Management
  - Análisis de acciones
- Optimización de aplicaciones
  - Almacenamiento en caché integrado
- Redirección BGP e inyección de estado de ruta (RHI)

> **Nota:**
>
> Las funciones de interfaz como Rx, Tx, GRO, GSO y LRO están inhabilitadas para las interfaces (host Linux) asignadas al dispositivo NetScaler CPX. Estas funciones permanecen en estado inhabilitado incluso después de que se detiene el dispositivo NetScaler CPX. Además, la MTU se cambia a 1500 bytes para dichas interfaces.

Plataformas compatibles

NetScaler CPX se admite en las siguientes plataformas:

- Kubernetes
- Red Hat OpenShift
- Nubes públicas
  - Servicio Amazon Elastic Kubernetes (EKS)
  - Servicio Azure Kubernetes (AKS)
  - Motor de Google Kubernetes (GKE)

- Ranchero
- Servicio de contenedores pivotales (PKS)
- Docker versión 1.12 y superior

Arquitectura y flujo de tráfico

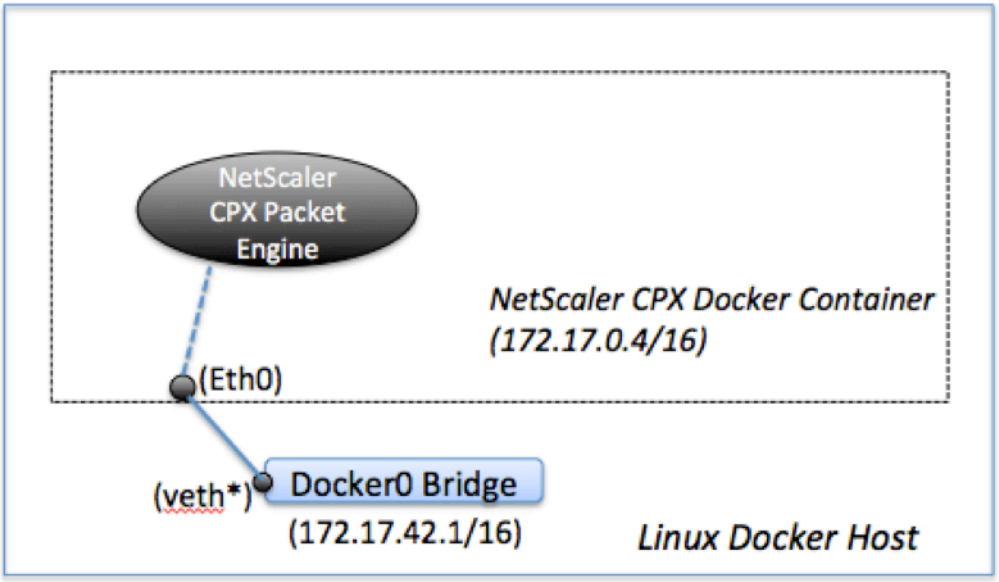
November 23, 2023

En esta sección se describe la arquitectura del modo puente de NetScaler CPX y el flujo de tráfico. NetScaler CPX también se puede implementar en modo host.

Cuando aprovisiona una instancia de NetScaler CPX en un host de Docker, el motor de Docker crea una interfaz virtual, eth0, en la instancia CPX. Esta interfaz eth0 está conectada directamente a una interfaz virtual (veth*) en el puente docker0. El motor de Docker también asigna una dirección IP a la instancia de NetScaler CPX en la red 172.17.0.0/16.

La puerta de enlace predeterminada para la instancia CPX es la dirección IP del puente docker0, lo que significa que cualquier comunicación con la instancia de NetScaler CPX se realiza a través de la red Docker. Todo el tráfico entrante recibido desde el puente docker0 lo recibe la interfaz eth0 en la instancia NetScaler CPX y el motor de paquetes NetScaler CPX lo procesa.

Esta ilustración ilustra la arquitectura de una instancia de NetScaler CPX en un host Docker.



Cómo funciona una dirección IP única en NetScaler CPX

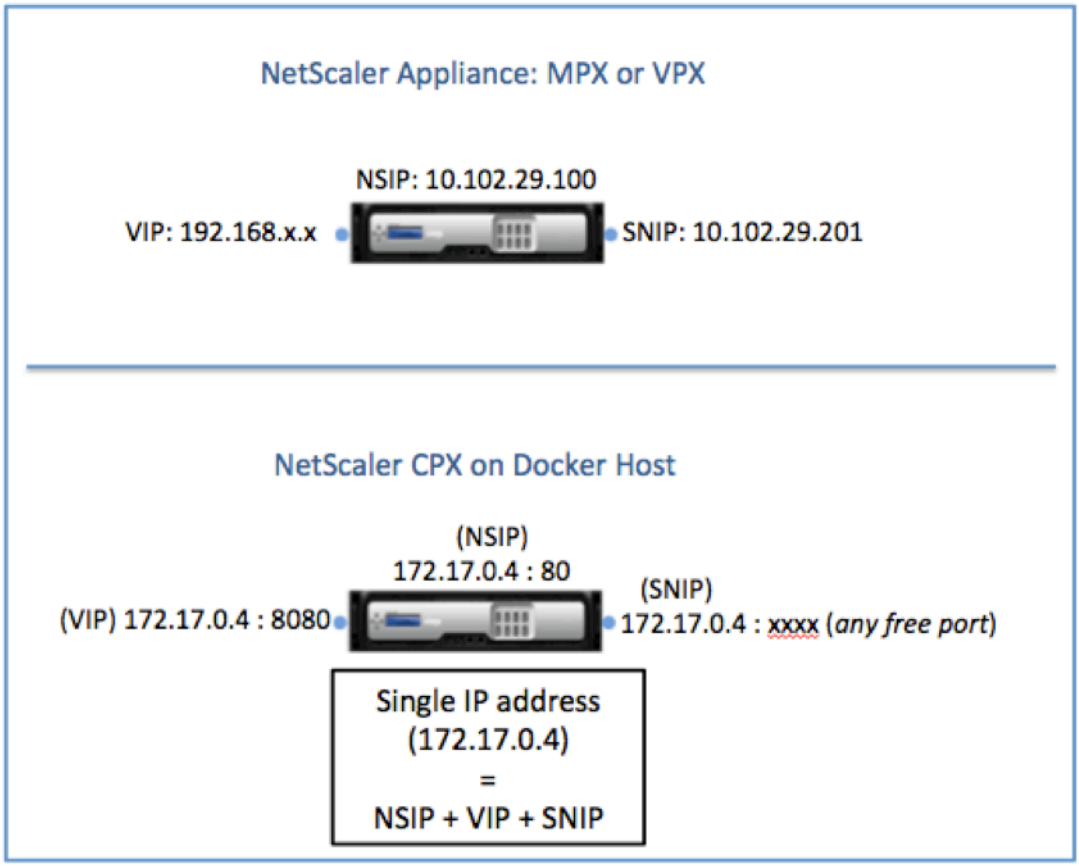
Un dispositivo NetScaler MPX o VPX normal requiere al menos tres direcciones IP para funcionar:

- Dirección IP de administración denominada dirección IP de NetScaler (NSIP)
- Dirección IP de subred (SNIP) para comunicarse con el conjunto de servidores
- Direcciones IP (VIP) del servidor virtual para aceptar solicitudes de clientes

Una instancia de NetScaler CPX funciona con una sola dirección IP que se utiliza tanto para la administración como para el tráfico de datos.

Durante el aprovisionamiento, el motor de Docker solo asigna una dirección IP privada (dirección IP única) a una instancia de NetScaler CPX. Las tres funciones IP de una instancia de NetScaler se multiplexan en una dirección IP. Esta única dirección IP utiliza diferentes números de puerto para funcionar como NSIP, SNIP y VIP (s).

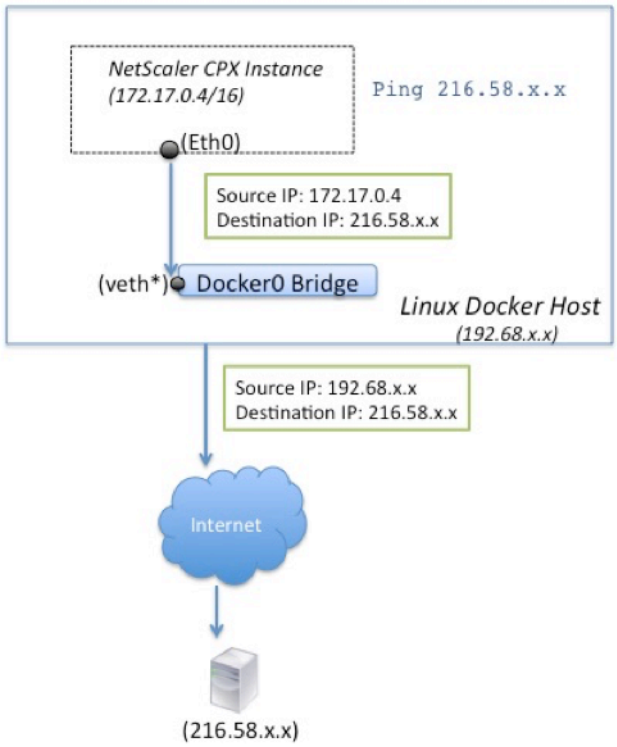
La siguiente imagen ilustra cómo se usa una sola dirección IP para realizar las funciones de NSIP, SNIP y VIP (s).



Flujo de tráfico para solicitudes que se originan en la instancia de NetScaler CPX

Docker configura implícitamente tablas IP y una regla NAT para dirigir el tráfico que se origina en la instancia de NetScaler CPX a la dirección IP de docker0.

Esta ilustración ilustra cómo una solicitud ping que se origina en una instancia de NetScaler CPX llega al destino.



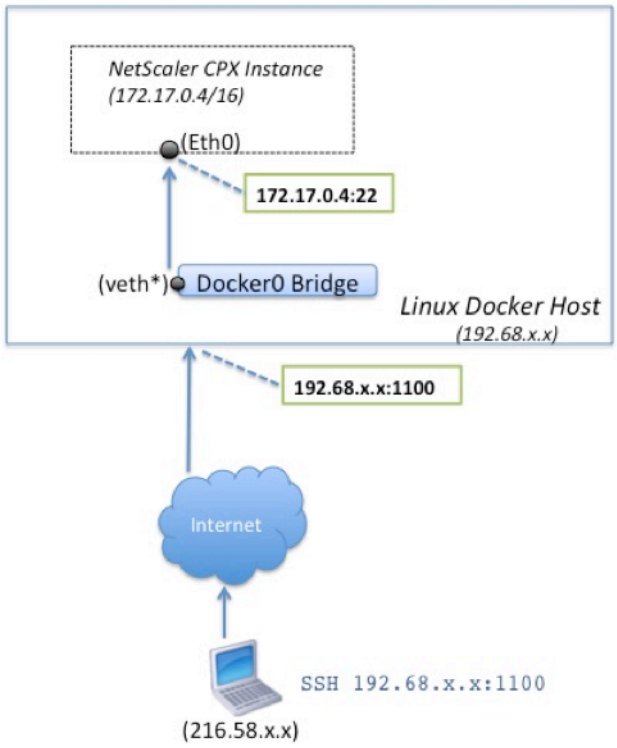
En este ejemplo, el motor de paquetes envía la solicitud ping en la interfaz eth0 con la dirección IP de origen como la dirección IP de NetScaler CPX (172.17.0.4). A continuación, el host de Docker realiza la traducción de direcciones de red (NAT) para agregar la dirección IP del host (192.68.x.x) como dirección IP de origen y envía la solicitud al destino (216.58.x.x). La respuesta de la dirección IP de destino sigue la misma ruta a la inversa. El host Docker realiza NAT en la respuesta y reenvía la respuesta a la instancia de NetScaler CPX en la interfaz eth0.

Flujo de tráfico para solicitudes que se originan en la red externa

Para habilitar la comunicación externa, al aprovisionar NetScaler CPX, debe establecer parámetros de modo que Docker exponga ciertos puertos, como 80, 22 y cualquier otro puerto que quiera. Si no ha configurado ningún puerto para que esté expuesto durante el aprovisionamiento, debe configurar reglas NAT en el host de Docker para que estos puertos estén disponibles.

La solicitud del cliente que se origina en Internet es recibida por el host de Docker, que luego realiza la traducción de direcciones de puerto (PAT) para asignar la dirección IP pública y el puerto a la dirección IP y el puerto únicos de la instancia de NetScaler CPX, y reenvía el tráfico a la instancia.

Esta ilustración muestra cómo el host Docker realiza la traducción de direcciones de puerto para dirigir el tráfico a la dirección IP única y al puerto NetScaler CPX.



En este ejemplo, la dirección IP del host de Docker es 192.68.x.x y la dirección IP única de la instancia de NetScaler CPX es 172.17.0.4. El puerto SSH 22 de la instancia de NetScaler CPX se asigna al puerto 1100 en el host Docker. La solicitud SSH del cliente se recibe en la dirección IP 192.68.x.x en el puerto 1100. El host de Docker realiza la traducción de direcciones de puerto para asignar esta dirección y puerto a la dirección IP única 172.17.0.4 en el puerto 22 y reenvía la solicitud del cliente.

Licencias de NetScaler CPX

July 22, 2024

NetScaler CPX es un controlador de entrega de aplicaciones basado en contenedores que se puede aprovisionar en un host Docker para equilibrar la carga de las aplicaciones basadas en microservicios. Necesita CPX con licencia para obtener un mejor rendimiento de la entrega de aplicaciones. NetScaler

CPX admite las licencias de grupos. NetScaler ADM puede actuar como su servidor de licencias para licenciar sus instancias de NetScaler CPX.

NetScaler ADM también está disponible en las instalaciones y en la nube. Puede usar NetScaler ADM para administrar licencias de capacidad agrupadas para todos los factores de forma de NetScaler.

Para obtener información sobre NetScaler ADM local, consulte NetScaler ADM local. Para obtener información sobre el servicio NetScaler ADM, consulte el servicio NetScaler ADM.

Tipos de licencias de NetScaler CPX

NetScaler CPX admite licencias de ancho de banda y grupos de CPU virtual (núcleo) para implementaciones locales y basadas en la nube.

Grupo de anchode banda: las licencias de NetScaler CPX se pueden asignar en función del consumo de ancho de banda de las instancias. Puede utilizar las licencias agrupadas para maximizar la utilización del ancho de banda al garantizar la asignación de ancho de banda necesaria a una instancia y no más de lo que se requiere. Actualmente, NetScaler CPX solo admite licencias de grupos de ancho de banda premium.

Grupo de vCPU: en las licencias virtuales basadas en el uso de CPU, la licencia especifica el número de CPU a las que tiene derecho una instancia de NetScaler CPX en particular. Por lo tanto, NetScaler CPX puede extraer licencias solo para la cantidad de CPU virtuales del servidor de licencias. NetScaler CPX desprotege las licencias en función del número de CPU que se ejecutan en el sistema. Para obtener más información sobre el grupo de vCPU, consulte Licencias de CPU virtuales de NetScaler.

Capacidad agrupada admitida para instancias NetScaler CPX

Producto	Ancho de banda máximo	Ancho de banda mínimo	Instancias mínimas	Instancias máximas	Unidad mínima de ancho de banda
NetScaler CPX	40000 **Nota**: Depende de la frecuencia de la CPU, la generación, etc.	20 Mbps	1	16	10 Mbps

Nota: Citrix está trabajando actualmente en un modelo de licencias basado en el consumo de

NetScaler CPX o basado en el pago a medida que crece para ofertas públicas basadas en la nube. Una vez listo, estará disponible en el mercado de la nube pública para su consumo.

¿Cómo funcionan las licencias de NetScaler CPX?

Capacidad agrupada de NetScaler CPX: un grupo de licencias común desde el que la instancia de NetScaler CPX puede extraer una licencia de instancia y solo el ancho de banda que necesite. Cuando la instancia ya no requiere estos recursos, los vuelve a registrar en el grupo común, lo que hace que los recursos estén disponibles para otras instancias que necesitan estas licencias.

Licencias de registro y salida de NetScaler CPX: NetScaler ADM asigna licencias de instancias NetScaler CPX a pedido. Una instancia de NetScaler CPX puede retirar la licencia de NetScaler ADM cuando se aprovisiona una instancia de NetScaler CPX y volver a registrar su licencia en NetScaler ADM cuando se destruye una instancia.

Comportamiento de NetScaler CPX: una sola instancia de NetScaler CPX que desprotege un rendimiento de hasta 1 Gbps, solo se desprotege del grupo de instancias y no del grupo de licencias de ancho de banda. NetScaler CPX funciona de esta manera hasta 1 Gbps de utilización de ancho de banda. Por ejemplo, si una instancia CPX consume un ancho de banda de 200 Mbps, utiliza el grupo de instancias de licencia, en lugar del conjunto de ancho de banda. Sin embargo, si una instancia de NetScaler CPX consume 1200 Mbps de rendimiento, los primeros 1000 Mbps se utilizan del grupo de instancias y los 200 Mbps restantes se consumen del grupo de ancho de banda.

NetScaler CPX Express

NetScaler CPX Express es una edición de software gratuita para implementaciones locales y en la nube. Cuando descarga la instancia de NetScaler CPX del repositorio de Quay, esta es la capacidad predeterminada disponible para los POC que no requieren un archivo de licencia y viene con estas funciones:

- Ancho de banda 20 Mbps
- Máximo 250 sesiones SSL
- Rendimiento SSL de 20 Mbps

Debe obtener una licencia de su instancia de NetScaler CPX para realizar la actualización y lograr un mejor rendimiento y implementaciones de producción.

Modelos de licencia de NetScaler CPX

NetScaler ofrece una gama de modelos de licencias de productos para NetScaler CPX a fin de cumplir con los requisitos de su organización. Puede seleccionar opciones como vCPU o ancho de banda y local o en la nube.

Según sus requisitos, puede elegir cualquiera de los siguientes modelos:

- Licencias basadas en ancho de banda para NetScaler CPX desde el servicio ADM
- Licencias basadas en vCPU para NetScaler CPX desde el servicio ADM
- Licencias basadas en ancho de banda para NetScaler CPX desde ADM local
- Licencias basadas en vCPU para NetScaler CPX desde ADM local

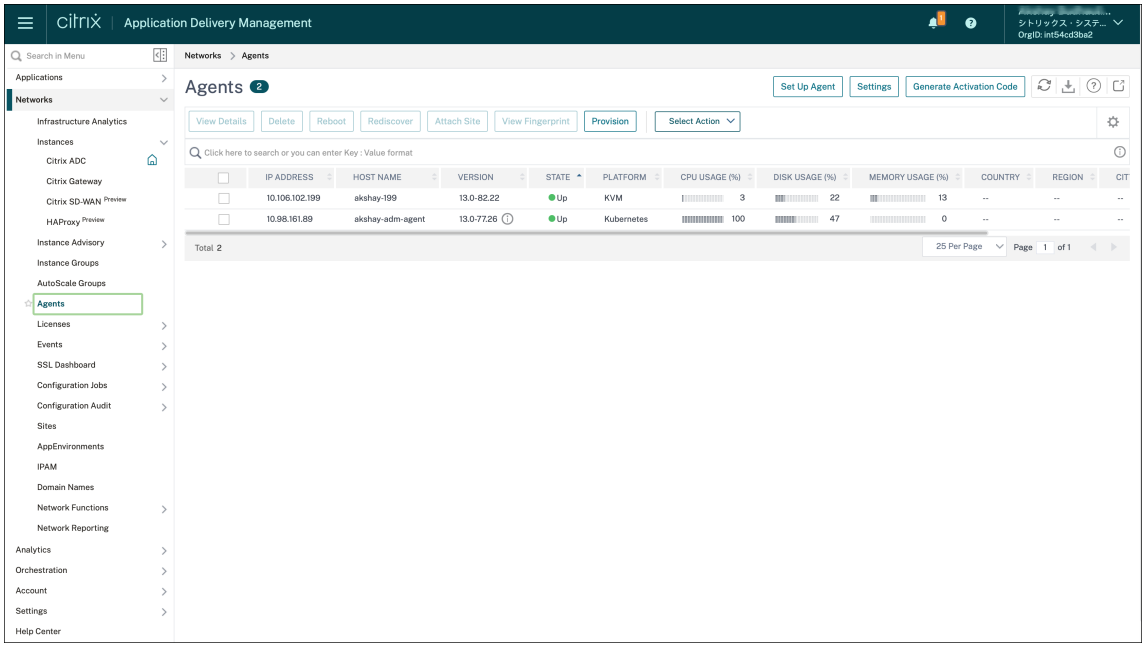
Aprovisionar licencias basadas en ancho de banda y basadas en CPU virtuales desde el servicio NetScaler ADM para NetScaler CPX

Realice los siguientes pasos para aprovisionar licencias basadas en ancho de banda y licencias basadas en CPU virtuales para NetScaler CPX desde el servicio NetScaler ADM.

1. Configure NetScaler ADM.

   Asegúrese de que la configuración del servicio NetScaler ADM funcione con el agente NetScaler ADM. Debe tener un servicio NetScaler ADM y una cuenta de agente NetScaler ADM para que las licencias de NetScaler CPX funcionen. Para obtener información sobre cómo configurar el servicio NetScaler ADM y el agente NetScaler ADM, consulte el servicio NetScaler ADM.

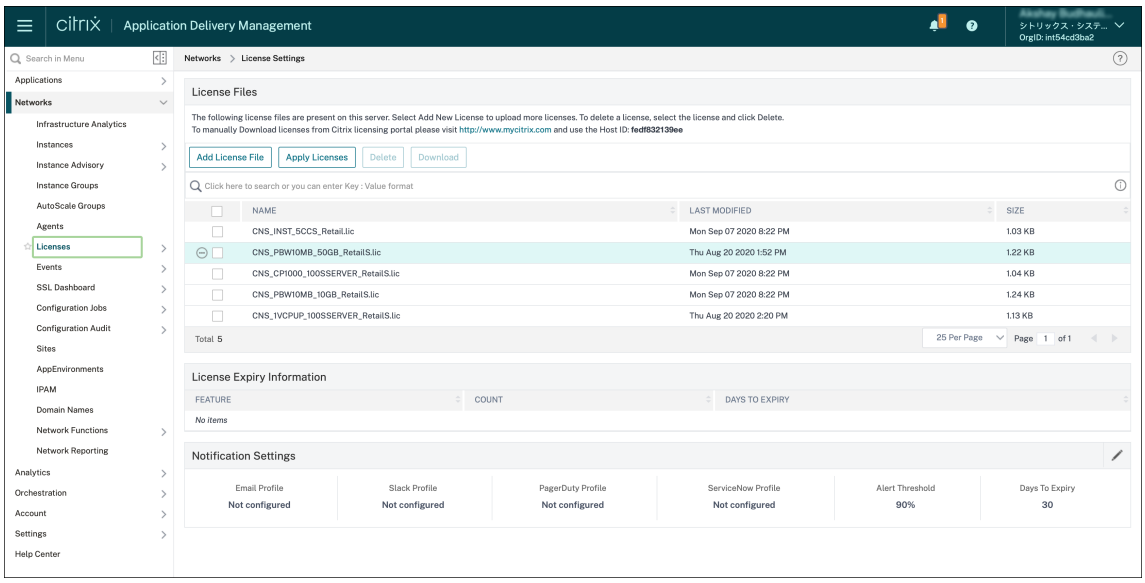
   Nota: En este procedimiento, se utiliza una configuración de agente NetScaler ADM de hipervisor (local). En la siguiente imagen, `10.106.102.199` se muestra el agente local que se utiliza para obtener licencias de NetScaler CPX.



2. Agregue el grupo de licencias de instancias de NetScaler al servicio NetScaler ADM.

   Se supone que tiene un conjunto de licencias de ancho de banda disponibles para el servicio ADM. Para obtener información sobre cómo cargar un archivo de licencia en NetScaler ADM, con-

sulte Configurar la capacidad agrupada. En la siguiente imagen, `CNS_INST_200CC_Retail.lic` se usa como el ancho de banda y el grupo de licencias de instancias.



3. Implemente la instancia de NetScaler CPX en el clúster de Kubernetes. Asegúrese de que las siguientes variables de entorno se agreguen al archivo YAML de NetScaler CPX para licenciar la instancia de NetScaler CPX.

   Para las licencias basadas en ancho de banda del servicio NetScaler ADM, especifique las siguientes variables de entorno en el archivo YAML:

   - nombre: “LS_IP”
     valor: “10.105.158.166”//IP del agente ADM como se mencionó en el paso 1
   - nombre: “LS_PORT”
     valor: “27000”// Puerto que escucha el servidor de licencias ADM
   - nombre: “BANDWIDTH”
     valor: “3000”//Capacidad en Mbps que quiere asignar a CPX
   - nombre: “EDITION”
     valor: “Standard”o “Enterprise”//para elegir una edición de licencia en particular que incluya Standard, Platinum y Enterprise. De forma predeterminada, se selecciona Platinum.

   Para las licencias basadas en CPU virtuales del servicio NetScaler ADM, especifique las siguientes variables de entorno en el archivo YAML:

   - nombre: “LS_IP”
     valor: “10.102.216.173”//IP del agente ADM como se mencionó en el paso 1
   - nombre: “LS_PORT”
     valor: “27000”// Puerto que escucha el servidor de licencias ADM
   - nombre: “CPX_CORES”
     valor: “4”// Cantidad de núcleos que quiere asignar

- nombre: “PLATFORM”
  valor: “CP1000”// Cantidad de núcleos. El recuento de desprotección es igual al número de núcleos.

4. Descargue el `cpx-bandwidth-license-adm-service.yaml` archivo mediante el siguiente comando:

```
kubectl create namespace bandwidth
wget https://raw.githubusercontent.com/citrix/cloud-native-getting-started/master/cpx-licensing/manifest/cpx-bandwidth-license-adm-service.yaml
```

5. Implemente el YAML modificado en el clúster de Kubernetes con el siguiente comando:

```
kubectl create -f cpx-bandwidth-license-adm-service.yaml -n bandwidth
```

6. Inicie sesión en NetScaler CPX para verificar la información de instancias mediante el siguiente comando:

```
kubectl exec -it 'cpx-pod-ip-name' bash -n bandwidth
```

7. Para ver la información de licencias de la instancia de NetScaler CPX determinada, ejecute los siguientes comandos:

```
cli_script.sh “show licenseserver”
cli_script.sh “show capacity”
```

   Puede realizar un seguimiento del ancho de banda asignado y la capacidad de vCPU en el portal de servicios ADM.

Aprovisionar licencias basadas en ancho de banda y licencias basadas en vCPU para NetScaler CPX desde NetScaler ADM local

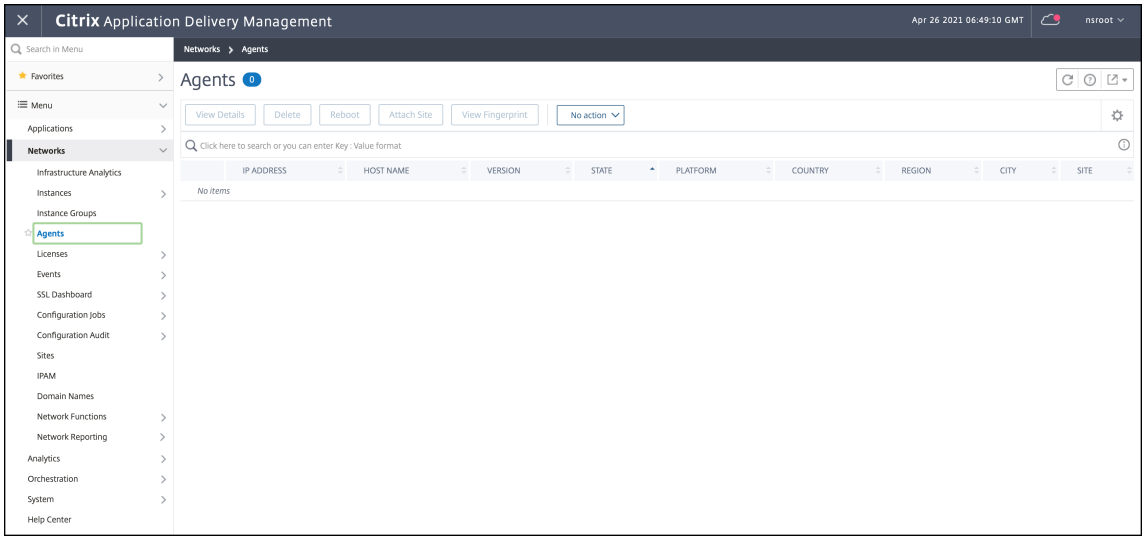
Realice los siguientes pasos para aprovisionar en función del ancho de banda y de la CPU virtual en NetScaler CPX desde NetScaler ADM local.

1. Configure NetScaler ADM.

   Asegúrese de que la configuración local de ADM esté lista. Asegúrese de que NetScaler ADM local con o sin la implementación del agente ADM para las licencias de NetScaler CPX esté funcionando.

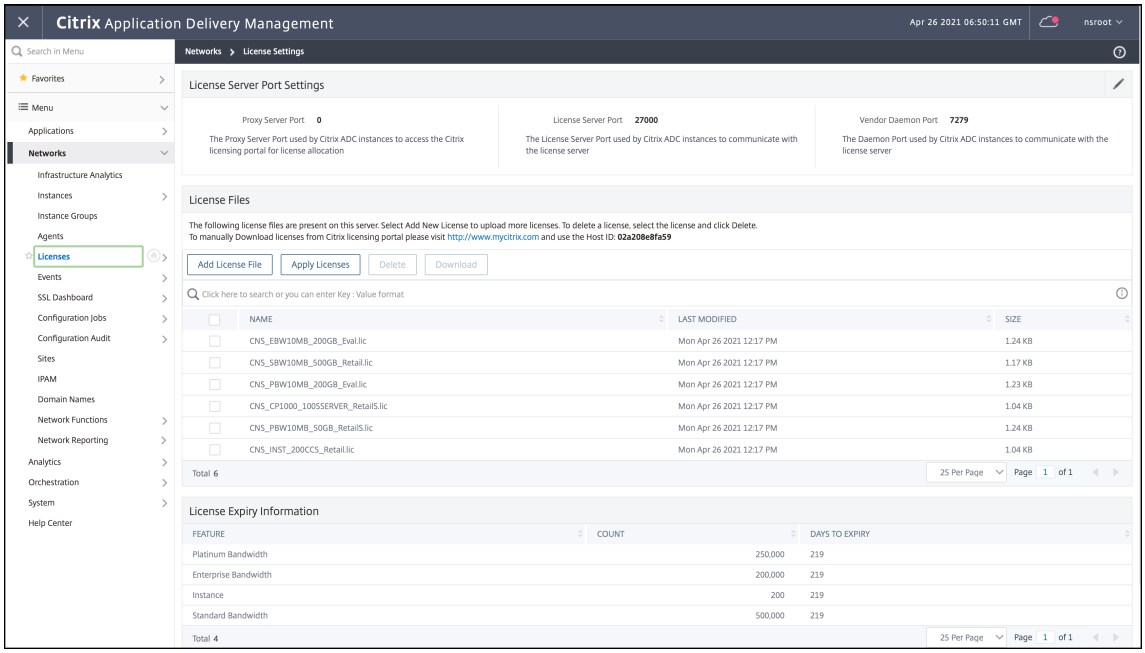
   Para obtener información sobre cómo configurar NetScaler ADM local y el agente NetScaler ADM, consulte el servicio NetScaler ADM.

   Nota: En este ejemplo, se utiliza un agente ADM integrado con ADM local. En la siguiente imagen, puede ver que no hay ningún agente implementado.

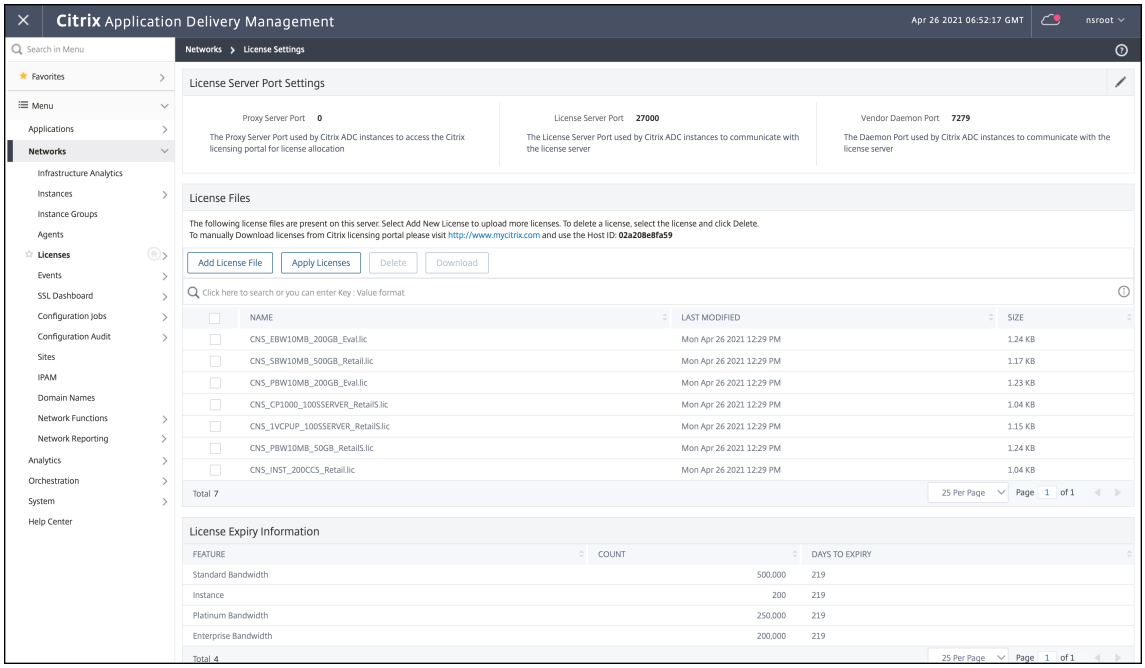


2. Agregue el grupo de licencias de instancias de NetScaler a ADM local.

   Se supone que tiene un conjunto de licencias de ancho de banda disponible para ADM local. Para obtener información sobre cómo cargar un archivo de licencia en NetScaler ADM, consulte Licencias. En la siguiente imagen, `CNS_INST_200CC_Retail.lic` se usa como el ancho de banda y el grupo de licencias de instancias.



   En la siguiente imagen, CP1000 se usa como el grupo de licencias de vCPU.



3. Implemente la instancia de NetScaler CPX en el clúster de Kubernetes. Asegúrese de que las siguientes variables de entorno se agreguen al archivo YAML de NetScaler CPX para licenciar la instancia de NetScaler CPX.

   Para las licencias basadas en ancho de banda de NetScaler ADM local, especifique las siguientes variables de entorno en el archivo YAML:

   - nombre: “LS_IP”
     valor: “10.105.158.144”// IP de instancia local de ADM, si ha implementado el agente ADM, esta es la dirección IP de su agente como se describe en el paso 1
   - nombre: “LS_PORT”
     valor: “27000”// Puerto que escucha el servidor de licencias ADM
   - nombre: “BANDWIDTH”
     valor: “3000”//Capacidad en Mbps que quiere asignar a CPX

   Para las licencias basadas en CPU virtuales de NetScaler ADM local, especifique las siguientes variables de entorno en el archivo YAML:

   - nombre: “LS_IP”
     valor: “10.105.158.144”// IP de instancia local de ADM, si tiene implementación de agente ADM, esta será la IP de su agente como se describe en el paso 1
   - nombre: “LS_PORT”
     valor: “27000”// Puerto que escucha el servidor de licencias ADM
   - nombre: “CPX_CORES”
     valor: “4”// La cantidad de núcleos que quiere asignar
   - nombre: “PLATFORM”

valor: “CP1000”// Cantidad de núcleos. El recuento de desprotección es igual al número de núcleos.

4. Descargue el `cpx-bandwidth-license-adm-onprem.yaml` archivo mediante el siguiente comando:

```
kubectl create namespace bandwidth
wget https://raw.githubusercontent.com/citrix/cloud-native-getting-started/master/cpx-licensing/manifest/cpx-bandwidth-license-adm-onprem.yaml
```

5. Implemente el YAML modificado en el clúster de Kubernetes con el siguiente comando:

```
kubectl create -f cpx-bandwidth-license-adm-onprem.yaml -n bandwidth
```

6. Inicie sesión en NetScaler CPX para verificar la información de instancias mediante el siguiente comando:

```
kubectl exec -it <cpx-pod-ip-name> bash -n bandwidth
```

7. Para ver la información de licencias de la instancia de NetScaler CPX, ejecute los siguientes comandos:

```
cli_script.sh “show licenseserver”
cli_script.sh “show capacity”
```

Puede realizar un seguimiento del ancho de banda asignado y la capacidad de vCPU en el portal local de ADM.

Comandos para limpiar las implementaciones

Puede usar los siguientes comandos para limpiar las distintas implementaciones de YAML:

```
kubectl delete -f cpx-bandwidth-license-adm-service.yaml -n bandwidth
kubectl delete -f cpx-core-license-adm-service.yaml -n core
kubectl delete -f cpx-bandwidth-license-adm-onprem.yaml -n bandwidth
kubectl delete -f cpx-core-license-adm-onprem.yaml -n core
kubectl delete namespace bandwidth
kubectl delete namespace core
```

Implementación de una instancia de NetScaler CPX en Docker

November 23, 2023

Las instancias de NetScaler CPX están disponibles como un archivo de imagen de Docker en el registro de contenedores de Quay. Para implementar una instancia, descargue la imagen de NetScaler CPX del registro de contenedores Quay y, a continuación, implemente la instancia mediante el comando `docker run` o la herramienta de redacción de Docker.

Requisitos previos

Asegúrese de que:

- El sistema host de Docker tiene al menos:
  - 1 CPU
  - 2 GB de RAM

    > **Nota:**Para obtener un mejor rendimiento de NetScaler CPX, puede definir la cantidad de motores de procesamiento que quiere que inicie la instancia de NetScaler CPX. Para cada motor de procesamiento adicional que agregue, asegúrese de que el host de Docker contenga la cantidad equivalente de CPU virtuales y la cantidad de memoria en GB. Por ejemplo, si quiere agregar 4 motores de procesamiento, el host de Docker debe contener 4 vCPU y 4 GB de memoria.

- El sistema host Docker ejecuta Linux Ubuntu versión 14.04 o posterior.
- La versión 1.12 de Docker está instalada en el sistema host. Para obtener información sobre la instalación de Docker en Linux, consulte ladocumentación de Docker.
- El host de Docker tiene conectividad a Internet.

  Nota: NetScaler CPX tiene problemas cuando se ejecuta en ubuntu versión 16.04.5, kernel versión 4.4.0-131-generic. Por lo tanto, no se recomienda ejecutar NetScaler CPX en ubuntu versión 16.04.5 kernel versión 4.4.0-131-generic.

  Nota: Las siguientes versiones de kubelet y kube-proxy tienen algunas vulnerabilidades de seguridad y no se recomienda utilizar Citrix NetScaler CPX con estas versiones:

  - kubelet/kube-proxy v1.18.0-1.18.3
  - kubelet/kube-proxy v1.17.0-1.17.6
  - kubelet/kube-proxy <=1.16.10

  Para obtener información sobre cómo mitigar esta vulnerabilidad, consulte Mitigar esta vulnerabilidad.

Descarga de la imagen de NetScaler CPX desde Quay

Puede descargar la imagen de NetScaler CPX desde el registro de contenedores de Quay mediante el comando `docker pull` e implementarla en su entorno. Use el siguiente comando para descargar la imagen de NetScaler CPX del registro de contenedores de Quay:

```
docker pull quay.io/citrix/citrix-k8s-cpx-ingress:13.0-xx.xx
```

Por ejemplo, si quiere descargar la versión 13.0-64.35, utilice el siguiente comando:

```
docker pull quay.io/citrix/citrix-k8s-cpx-ingress:13.0-64.35
```

Use el siguiente comando para comprobar si la imagen de NetScaler CPX está instalada en imágenes de docker:

```
root@ubuntu:~# docker images | grep 'citrix-k8s-cpx-ingress'
quay.io/citrix/citrix-k8s-cpx-ingress                    13.0-64.35
              952a04e73101        2 months ago        469 MB
```

Puede implementar la última imagen de NetScaler CPX desde el registro de contenedores Quay.

Implementación de la instancia de NetScaler CPX mediante el comando docker run

En el host, puede instalar una instancia de NetScaler CPX en el contenedor de Docker mediante la imagen de Docker de NetScaler CPX que cargó en el host. Mediante el comando `docker run`, instale la instancia de NetScaler CPX con la configuración predeterminada de NetScaler CPX.

> **Importante:**
>
> Si ha descargado NetScaler CPX Express de CPX Express, asegúrese de leer y entender el Contrato de licencia de usuario final (EULA) disponible en:CPX Expressy de aceptar el EULA al implementar la instancia de NetScaler CPX.

Instale la instancia de NetScaler CPX en el contenedor de Docker mediante el siguiente comando ***docker run***:

```
docker run -dt -P --privileged=true --net=host – e NS_NETMODE= ” HOST ”
   -e CPX_CORES=<number of cores> --name <container_name> --ulimit core
   =-1 -e CPX_NW_DEV='<INTERFACES>' -e CPX_CONFIG= ’ {
 “ YIELD ” : ” NO ”  }
 ’  -e LS_IP=<LS_IP_ADDRESS> -e LS_PORT=<LS_PORT> e PLATFORM=CP1000 -v
    <host_dir>:/cpx <REPOSITORY>:<TAG>
<!--NeedCopy-->
```

```
docker run -dt --privileged=true --net=host -e NS_NETMODE="HOST" -e
   CPX_NW_DEV='eth1 eth2' -e CPX_CORES=5  – e CPX_CONFIG='{
 "YIELD":"No" }
```



```
3   ' -e LS_IP=10.102.38.134 -e PLATFORM=CP1000 -v /var/cpx:/cpx --name
       cpx_host cpx:13.0-x.x
4 <!--NeedCopy-->
```

En este ejemplo se crea un contenedor denominado en`mycpx`función de la imagen de Docker de NetScaler CPX.

El`-P`parámetro es obligatorio. Indica a Docker que asigne los puertos expuestos en el contenedor por la imagen de NetScaler CPX Docker. Esto significa asignar los puertos 9080, 22, 9443 y 161/UDP a los puertos del host Docker que se seleccionan aleatoriamente del intervalo definido por el usuario. Este mapeo se hace para evitar conflictos. Si más tarde crea varios contenedores NetScaler CPX en el mismo host de Docker. Las asignaciones de puertos son dinámicas y se configuran cada vez que se inicia o se reinicia el contenedor. Los puertos se usan de la siguiente manera:

- 9080 se usa para HTTP
- 9443 se usa para HTTPs
- 22 utilizados para SSH
- 161/UDP se utiliza para SNMP.

Si quiere asignaciones de puertos estáticas, use el parámetro -p para configurarlas manualmente.

La opción `--privileged=true` se usa para ejecutar el contenedor en modo privilegiado. Si ejecuta NetScaler CPX en modo Host de implementación, debe proporcionar todos los privilegios del sistema a NetScaler CPX. Si quiere ejecutar NetScaler CPX en modo puente con uno o varios núcleos, en lugar de esta opción, puede usar la opción `--cap-add=NET_ADMIN`. La opción `--cap-add=NET_ADMIN` le permite ejecutar el contenedor de NetScaler CPX con todos los privilegios de red.

`**--net=host` Es una opción de comando de ejecución de docker estándar que especifica que el contenedor se ejecuta en la pila de red del host y tiene acceso a todos los dispositivos de red.

> **Nota**
>
> Ignore esta opción si ejecuta NetScaler CPX en una red puente o ninguna.

`-e NS_NETMODE="HOST"` es una variable de entorno específica de NetScaler CPX que le permite especificar que NetScaler CPX se inicie en modo host. Una vez que NetScaler CPX se inicia en modo host, configura 4 reglas iptables predeterminadas en una máquina host para el acceso de administración a NetScaler CPX. Utiliza los siguientes puertos:

- 9995 para HTTP
- 9996 para HTTPS
- 9997 para SSH
- 9998 para SNMP

Si quiere especificar puertos diferentes, puede usar las siguientes variables de entorno:

- -e NS_HTTP_PORT=
- -e NS_HTTPS_PORT=
- -e NS_SSH_PORT=
- -e NS_SNMP_PORT=

> **Nota**
>
> Ignore esta variable de entorno si ejecuta NetScaler CPX en una red puente o ninguna.

Se`-e CPX_CORES`trata de una variable de entorno opcional específica de NetScaler CPX. Puede usarlo para mejorar el rendimiento de la instancia de NetScaler CPX definiendo el número de motores de procesamiento que quiere que se inicie el contenedor de NetScaler CPX.

Nota: NetScaler CPX admite de 1 a 16 núcleos.

> **Nota**
>
> Por cada motor de procesamiento adicional que agregue, asegúrese de que el host de Docker contenga la cantidad equivalente de CPU virtuales y la cantidad de memoria en GB. Por ejemplo, si quiere agregar 4 motores de procesamiento, el host de Docker debe contener 4 vCPU y 4 GB de memoria.

`-e EULA = yes`Se trata de una variable de entorno específica de NetScaler CPX obligatoria, que se requiere para comprobar que ha leído y comprendido el Contrato de licencia de usuario final (EULA) disponible en:CPX Express.

El`-e PLATFORM=CP1000`parámetro especifica el tipo de licencia de NetScaler CPX.

Si ejecuta Docker en una red host, puede asignar interfaces de red dedicadas al contenedor NetScaler CPX mediante la variable de entorno `-e CPX_NW_DEV`. Debe definir las interfaces de red separadas por un espacio en blanco. Las interfaces de red que defina se guardan en el contenedor NetScaler CPX hasta que desinstala el contenedor NetScaler CPX. Cuando se aprovisiona el contenedor NetScaler CPX, todas las interfaces de red asignadas se agregan al espacio de nombres de redes NetScaler.

> **Nota**
>
> Si ejecuta NetScaler CPX en una red puente, puede cambiar la red de contenedores, por ejemplo, configurar otra conexión de red al contenedor o eliminar una red existente. A continuación, asegúrese de reiniciar el contenedor NetScaler CPX para usar la red actualizada.

```
1 docker run -dt --privileged=true --net=host -e NS_NETMODE="HOST" -e
     EULA=yes -e CPX_NW_DEV='eth1 eth2' -e CPX_CORES=5 -e PLATFORM=CP1000
      --name cpx_host cpx:13.0-x.x
2 <!--NeedCopy-->
```

`-e CPX_CONFIG` es una variable de entorno específica de NetScaler CPX que le permite controlar el rendimiento de rendimiento del contenedor NetScaler CPX. Cuando NetScaler CPX no recibe ningún

tráfico entrante para procesar, produce la CPU durante este tiempo de inactividad, lo que resulta en un rendimiento bajo. Puede usar la variable de entorno `CPX_CONFIG` para controlar el rendimiento del rendimiento del contenedor NetScaler CPX en tales casos. Debe proporcionar los siguientes valores a la variable de entorno `CPX_CONFIG` en formato JSON:

- Si quiere que el contenedor NetScaler CPX genere CPU en casos inactivos, defina `{ "YIELD" : "Yes" }`
- Si quiere que el contenedor NetScaler CPX evite producir la CPU en casos inactivos para que pueda obtener un rendimiento de alto rendimiento, defina `{ "YIELD" : "No" }`

```
1 docker run -dt --privileged=true --net=host -e NS_NETMODE="HOST" -e
     EULA=yes -e CPX_CORES=5 -e CPX_CONFIG='{
2  "YIELD":"No" }
3  ' -e PLATFORM=CP1000 --name cpx_host cpx:13.0-x.x
4 <!--NeedCopy-->
```

```
1 docker run -dt --privileged=true --net=host -e NS_NETMODE="HOST" -e
     EULA=yes -e CPX_CORES=5 -e CPX_CONFIG='{
2  "YIELD":"Yes" }
3  ' -e PLATFORM=CP1000 --name cpx_host cpx:13.0-x.x
4 <!--NeedCopy-->
```

El `-v` parámetro es un parámetro opcional que especifica el punto de montaje del directorio de montaje de NetScaler CPX, `/cpx`. Un punto de montaje es un directorio en el host, en el que se monta el `/cpx` directorio. El `/cpx` directorio almacena los registros, los archivos de configuración, los certificados SSL y los archivos de volcado de memoria. En el ejemplo, el punto de montaje es `/var/cpx` y el directorio de montaje de NetScaler CPX es `/cpx`.

Si compró una licencia o tiene una licencia de evaluación, puede cargar la licencia en un servidor de licencias y especificar la ubicación del servidor de licencias con el comando docker run, mediante el `-e LS_IP=<LS_IP_ADDRESS> -e LS_PORT=<LS_PORT>` parámetro. En este caso, no necesita aceptar el EULA.

```
1 docker run -dt --privileged=true --net=host -e NS_NETMODE="HOST" -e
     CPX_CORES=5 -e CPX_CONFIG='{
2  "YIELD":"No" }
3  ' -e LS_IP=10.102.38.134 -e PLATFORM=CP1000 --name cpx_host cpx:13.0-x
     .x
4 <!--NeedCopy-->
```

Donde:

- `LS_IP_ADDRESS` es la dirección IP del servidor de licencias.
- `LS_PORT` es el puerto del servidor de licencias.

Puede ver las imágenes que se ejecutan en su sistema y los puertos asignados a los puertos estándar mediante el comando: `docker ps`

Implementación de una versión más ligera de NetScaler CPX mediante el comando docker run

NetScaler proporciona una versión más ligera de NetScaler CPX que consume menos memoria en tiempo de ejecución. La versión más ligera de NetScaler CPX se puede implementar como un sidecar en implementaciones de malla de servicio.

La versión más ligera de NetScaler CPX admite las siguientes funciones:

- Disponibilidad de aplicaciones
  - Equilibrio de carga L4 y conmutación de contenido L7
  - Descarga de SSL
  - Traducción de protocolos IPv6
- Seguridad de las aplicaciones
  - Reescritura y respuesta de L7
- Capacidad de administración sencilla
  - Registro web
  - AppFlow

Para crear una instancia de la versión más ligera de NetScaler CPX, establezca la variable de entorno `NS_CPX_LITE` mientras ejecuta el comando `Docker run`.

```
1 docker run -dt -P --privileged=true -e NS_CPX_LITE=1 -e EULA=yes --name
      <container_name> --ulimit core=-1 <REPOSITORY>:<TAG>
2 <!--NeedCopy-->
```

En el siguiente ejemplo, se crea un contenedor ligero basado en la imagen de NetScaler CPX.

```
1 docker run  – dt -P --privileged=true -e NS_CPX_LITE=1 -e EULA=yes - –
     name lightweight --ulimit core=-1 cpx:latest
2 <!--NeedCopy-->
```

De forma predeterminada, el registro mediante `newnslog` está inhabilitado en la versión más ligera de NetScaler CPX. Para habilitarlo, debe establecer la variable de entorno `NS_ENABLE_NEWNSLOG` en 1 mientras abre la versión más ligera de NetScaler CPX.

El siguiente ejemplo muestra cómo habilitar el registro mediante `newnslog` al implementar la versión más ligera de NetScaler CPX.

```
1 docker run -dt --privileged=true --ulimit core=-1 -e EULA=yes -e
     NS_CPX_LITE=1 -e NS_ENABLE_NEWNSLOG=1 cpx:<tag>
2 <!--NeedCopy-->
```


> **Nota:** La versión más ligera de CPX solo admite un solo núcleo (CPX_CORES=1).

Implementación de instancias de NetScaler CPX mediante Docker Compose

Puede usar la herramienta Redactar de Docker para aprovisionar una sola instancia de NetScaler CPX o varias instancias de NetScaler CPX. Para aprovisionar instancias de NetScaler CPX mediante Docker Compose, primero debe escribir un archivo de redacción. Este archivo especifica la imagen de NetScaler CPX, los puertos que quiere abrir para la instancia de NetScaler CPX y los privilegios de la instancia de NetScaler CPX.

> **Importante**
>
> Asegúrese de haber instalado la herramienta Docker Compose en el host.

Para aprovisionar varias instancias de NetScaler CPX:

1. Escribe un archivo de redacción, donde:

- **<service-name>** es el nombre del servicio que quieres aprovisionar.
- **image::** <repository><tag>indica el repositorio y las versiones de la imagen de NetScaler CPX.
- **privileged: true** proporciona todos los privilegios de root a la instancia de NetScaler CPX.
- **cap_add** proporciona privilegios de red a la instancia de NetScaler CPX.
- ** <host_directory_path>indica el directorio en el host de la ventana acoplable que quiere montar para la instancia de NetScaler CPX.
- ** <number_processing_engine>es el número de motores de procesamiento que quiere que se inicie la instancia de NetScaler CPX. Para cada motor de procesamiento adicional, asegúrese de que el host de Docker contenga la cantidad equivalente de CPU virtuales y la cantidad de memoria en GB. Por ejemplo, si quiere agregar 4 motores de procesamiento, el host de Docker debe contener 4 vCPU y 4 GB de memoria.

El archivo de redacción suele tener un formato similar al siguiente:

```
    <service-name>:
    container_name:
    image: <repository>:<tag>
    ports:
        - 22
        - 9080
        - 9443
        - 161/udp
        - 35021-35030
    tty: true
    cap_add:
        - NET_ADMIN
    ulimits:
        core: -1
```



```
    volumes:
        - <host_directory_path>:/cpx
    environment:
        - EULA=yes
        - CPX_CORES=<number_processing_engine>
        - CPX_CONFIG='{
 "YIELD":"Yes" }
 '
<!--NeedCopy-->
```

```
    CPX_0:
    image: quay.io/citrix/citrix-k8s-cpx-ingress:13.1-37.38
    ports:
        -  9443
        -  22
        -  9080
        -  161/udp
    tty: true
    cap_add:
        - NET_ADMIN
    ulimits:
        core: -1
    volumes:
        - /root/test:/cpx
  environment:
        -  CPX_CORES=2
        -  EULA=yes
<!--NeedCopy-->
```

Agregar instancias de NetScaler CPX a NetScaler ADM

March 21, 2024

Debe agregar las instancias de NetScaler CPX instaladas en un host de Docker al software NetScaler Application Delivery Management (ADM) si quiere administrar y supervisar estas instancias.

Puede agregar instancias al configurar ADM por primera vez o más tarde.

Para agregar instancias, debes crear un perfil de instancia y especificar el nombre de host o la dirección IP de cada instancia, o un intervalo de direcciones IP. Este perfil de instancia contiene el nombre de usuario y la contraseña de las instancias que quiere agregar a NetScaler ADM. Para cada tipo de instancia, está disponible un perfil predeterminado. Por ejemplo, `ns-root-profile` es el perfil predeterminado para las instancias de NetScaler. Este perfil se define mediante las credenciales de administrador de ADC predeterminadas. Si ha cambiado las credenciales de administrador predeterminadas de las instancias, puede definir perfiles de instancia personalizados para esas instancias. Si cambia las credenciales de una instancia después de detectarse la instancia, debe modificar el perfil

de instancia o crear un perfil y, a continuación, volver a descubrir la instancia.

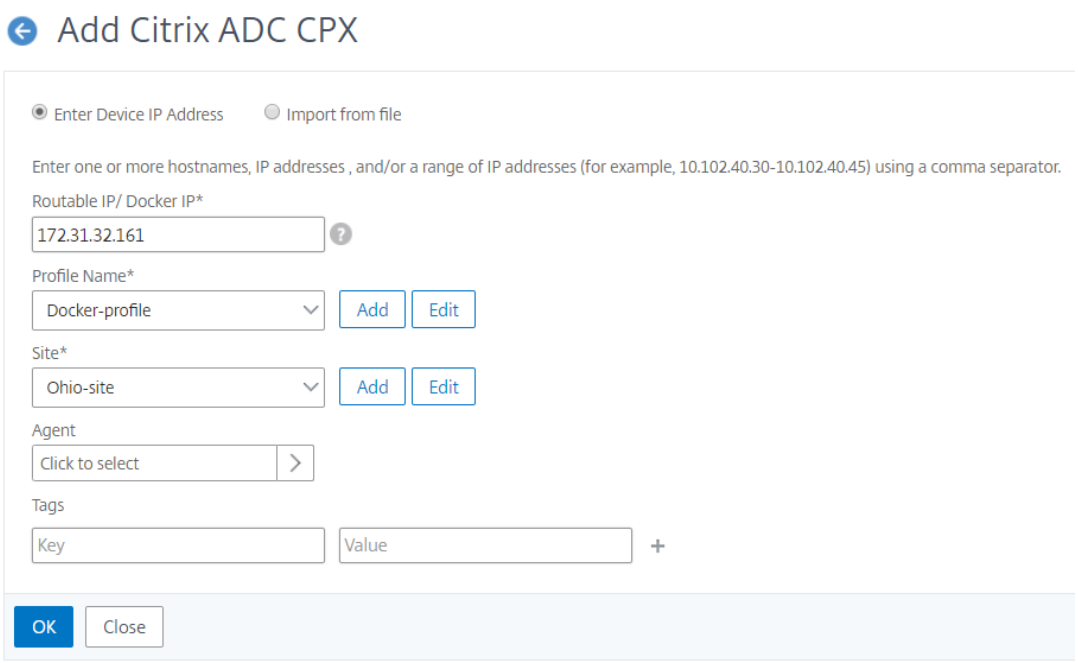
Requisitos previos

Asegúrese de lo siguiente:

- Se instaló el software NetScaler ADM en Citrix XenServer. Para obtener más información, consulte la documentación de NetScaler ADM.
- Se instalaron las instancias de NetScaler CPX en un host de Docker.

Para agregar instancias de NetScaler CPX a ADM:

1. En un explorador web, escriba la dirección IP de **NetScaler Application Delivery Management** (por ejemplo, `http://192.168.100.1`).
2. En los campos **Nombre de usuario** y **Contraseña**, introduzca las credenciales de administrador. Las credenciales de administrador predeterminadas son **nsroot** y **nsroot**.
3. Vaya a **Redes > Instancias** > **NetScaler** y haga clic en la ficha **CPX**.
4. Haga clic en **Agregar** para agregar nuevas instancias CPX en NetScaler ADM.
5. Se abrirá la página **Add NetScaler CPX**. Introduzca los valores de los siguientes parámetros:
    a) Puede agregar instancias de CPX proporcionando la dirección IP accesible de la instancia CPX o la dirección IP del contenedor Docker donde está alojada la instancia de CPX.
    b) Seleccione el perfil de la instancia CPX.
    c) Seleccione el sitio en el que se van a implementar las instancias.
    d) Seleccione el agente.
    e) Como opción, puede introducir el par clave-valor en la instancia. La incorporación de un par clave-valor facilita la búsqueda de la instancia más adelante.



6. Haga clic en **Aceptar**.

> **Nota**
>
> Si quiere volver a descubrir una instancia, elija **Redes > Instancias > NetScaler > CPX**, seleccione la instancia que quiere redescubrir y, a continuación, en la lista desplegable **Seleccionar acción**, haga clic en **Redescubrir**.

Agregar instancias de NetScaler CPX a NetScaler ADM mediante variables de entorno

También puede agregar las instancias de NetScaler CPX a NetScaler ADM mediante variables de entorno. Para agregar instancias, debe configurar las siguientes variables de entorno para la instancia de NetScaler CPX.

- `NS_MGMT_SERVER` - Dirección IP ADM/FQDN
- `HOST` - Dirección IP del nodo
- `NS_HTTP_PORT` - Puerto HTTP asignado en el nodo
- `NS_HTTPS_PORT`- Puerto HTTPS asignado en el nodo
- `NS_SSH_PORT` - Puerto SSH mapeado en el nodo
- `NS_SNMP_PORT` - Puerto SNMP asignado en el nodo
- `NS_ROUTABLE` - (La dirección IP del pod de NetScaler CPX no se puede redirigir desde el exterior).
- `NS_MGMT_USER` —Nombre de usuario de ADM
- `NS_MGMT_PASS` —Contraseña ADM

A continuación, se muestra un comando `docker run` de ejemplo para agregar una instancia de NetScaler CPX a NetScaler ADM.

```
1  docker run -dt --privileged=true -p 9080:9080 -p 9443:9443 -p 9022:22
      -p 9161:161 -e EULA=yes -e NS_MGMT_SERVER=abc-mgmt-server.com -e
      HOST=10.1.1.1 -e NS_HTTP_PORT=9080 -e NS_HTTPS_PORT=9443 -e
      NS_SSH_PORT=9022 -e NS_SNMP_PORT=9161 -e NS_ROUTABLE=0 --ulimit
      core=-1  – name test cpx:latest
2
3 <!--NeedCopy-->
```

Agregar instancias de NetScaler CPX a NetScaler ADM mediante Kubernetes ConfigMaps

NetScaler CPX admite el registro en NetScaler ADM mediante el uso de archivos montados en volumen a través de Kubernetes ConfigMaps. Para habilitar esta forma de registro, NetScaler CPX requiere algunas variables de entorno que se especificarán junto con algunos archivos montados en volúmenes a través de ConfigMaps y Secrets.

A continuación se muestran las variables de entorno requeridas y su descripción:

- `NS_HTTP_PORT` - Especifica el puerto HTTP asignado en el nodo.
- `NS_HTTPS_PORT` - Especifica el puerto HTTPS asignado en el nodo.
- `NS_SSH_PORT` - Especifica el puerto SSH asignado en el nodo.
- `NS_SNMP_PORT`- Especifica el puerto SNMP asignado en el nodo.

Además de las variables de entorno enumeradas, NetScaler CPX requiere información sobre el agente ADM con el que debe registrarse. Esta información contiene la dirección IP o los detalles y credenciales de FQDN del agente ADM. NetScaler CPX adquiere esta información de los archivos montados en el volumen. Un ConfigMap que contiene la dirección IP o el FQDN se monta como un archivo en el sistema de archivos de la instancia de NetScaler CPX. Un secreto de Kubernetes que contiene credenciales para el agente ADM también se monta como un archivo en el sistema de archivos de la instancia de NetScaler CPX. Con toda la información necesaria para el registro, NetScaler CPX intenta registrarse en el agente ADM.

A continuación se muestra un ejemplo de fragmento de archivo YAML de NetScaler CPX con ConfigMap y Secret montados como archivos:

```
1         ...
2         env:
3         - name: "EULA"
4           value: "yes"
5         - name: "NS_HTTP_PORT"
6           value: "9080"
7         - name: "NS_HTTPS_PORT"
8           value: "9443"
```



```
        - name: "NS_SSH_PORT"
          value: "22"
        - name: "NS_SNMP_PORT"
          value: "161"
        - name: "KUBERNETES_TASK_ID"
          value: ""
        ...
        volumeMounts:
          ...
          - mountPath: /var/adminfo/server/
            name: adm-agent-config
          - mountPath: /var/adminfo/credentials/
            name: adm-agent-user
          ...
    volumes:
    ...
    - name: adm-agent-config
      configMap:
        name: adm-agent-config
    - name: adm-agent-user
      secret:
        secretName: adm-secret
```

En el ejemplo anterior, se consumen un nombre ConfigMap `adm-agent-config` y un secreto `adm-agent-user`. A continuación se muestra un ejemplo para crear el ConfigMap y Secret necesarios.

ConfigMap: El ConfigMap se crea a partir de un archivo llamado `adm_reg_envs`. El archivo requiere la dirección IP o el FQDN del agente ADM en el siguiente formato:

```
NS_MGMT_SERVER=adm-agent
```

En el formato anterior, `adm-agent` es el FQDN del agente ADM en el que se debe registrar la instancia de NetScaler CPX.

Use el siguiente comando para crear un ConfigMap:

```
kubectl create configmap adm-agent-config --from-file=adm_reg_envs
```

Nota: El nombre del archivo debe tener la variable `adm_reg_envs` y debe montarse en la ruta: `/var/adminfo/server/`.

Secreto: usa el siguiente comando para crear un secreto de Kubernetes. En el siguiente comando, `user123` es el nombre de usuario del agente ADM y `pass123` es la contraseña.

```
kubectl create secret generic adm-secret --from-literal=NS_MGMT_USER=
    user123 --from-literal=NS_MGMT_PASS=pass123
```

Una instancia de NetScaler CPX se puede implementar en un clúster de Kubernetes con las variables de entorno y los archivos montados en volumen requeridos incluso antes de implementar el agente ADM en el clúster. Si implementa una instancia de NetScaler CPX antes de implementar el agente ADM, NetScaler CPX sigue intentando registrarse hasta que se implemente el agente ADM. Una vez

que se implementa el agente ADM, la instancia de NetScaler CPX utiliza los datos de configuración proporcionados a través de las variables de entorno y los archivos montados en volumen para registrarse en el agente ADM. Le ayuda a evitar la reimplementación de NetScaler CPX con la información de configuración.

Una instancia de NetScaler CPX, que ya está registrada con un agente ADM, puede cambiar dinámicamente el registro a otro agente ADM después de un cambio en la configuración. Para ello, puede actualizar la información de configuración en ConfigMap y el secreto para el NetScaler CPX ya implementado. Debe actualizar el archivo desde el que se creó el ConfigMap con la dirección IP o el FQDN del nuevo agente ADM y eliminar el ConfigMap anterior y, a continuación, crear un nuevo ConfigMap. Del mismo modo, se debe eliminar el secreto existente y se debe crear un nuevo secreto con las credenciales del nuevo agente de ADM.

Agregador de licencias NetScaler CPX

November 23, 2023

Actualmente, los de NetScaler CPX obtienen licencias del servidor NetScaler ADM. En un entorno de Kubernetes, los de NetScaler CPX pueden subir o bajar dinámicamente. Si un NetScaler CPX deja de funcionar inesperadamente, el servidor NetScaler ADM tarda unos minutos en reclamar la licencia. El servidor NetScaler ADM debe poder recuperar dichas licencias inmediatamente cuando los de NetScaler CPX dejen de funcionar, de modo que se pueda asignar la misma licencia a otro NetScaler CPX próximo. Además, si no se puede acceder al servidor NetScaler ADM por algún motivo, no puede obtener licencias de los nuevos de NetScaler CPX en el clúster.

El agregador de licencias de NetScaler CPX es un servicio de Kubernetes proporcionado por NetScaler. Este servicio actúa como proveedor local de licencias de NetScaler CPX dentro de un clúster de Kubernetes. El servicio NetScaler CPX License Aggregator implementado en un clúster de Kubernetes puede actuar como intermediario entre los de NetScaler CPX y el servidor de licencias de ADM y realizar un seguimiento de los de NetScaler CPX y las licencias asignadas. Con el servicio NetScaler CPX License Aggregator, el servidor NetScaler ADM puede reclamar licencias inmediatamente cuando NetScaler CPX deja de funcionar.

En un clúster de Kubernetes, el servicio NetScaler CPX License Aggregator admite NetScaler CPX como sidecar e implementaciones independientes.

> **Nota:**
>
> Para obtener licencias mediante el agregador de licencias de NetScaler CPX se requiere NetScaler CPX 13.1-30.x o una versión posterior. NetScaler CPX License Aggregator no admite la licencia de las versiones anteriores de NetScaler CPX.

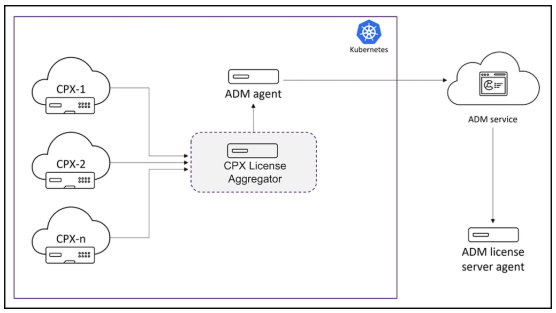
Beneficios clave de NetScaler CPX License Aggregator

Los siguientes son los principales beneficios de usar NetScaler CPX License Aggregator:

- Escalabilidad:
  un servidor de licencias NetScaler ADM solo puede admitir hasta 10 000 implementaciones de NetScaler CPX. Con la introducción del servicio NetScaler CPX License Aggregator, cada clúster de Kubernetes puede actuar como un único cliente para el servidor de licencias de NetScaler ADM. Por lo tanto, puede escalar muchos de NetScaler CPX con un único servidor de licencias NetScaler ADM.
- Optimización de recursos:
  el servicio NetScaler CPX License Aggregator también admite la capacidad de licencias de todo el clúster y también puede extraer licencias del servidor NetScaler ADM según sea necesario. NetScaler CPX License Aggregator puede devolver las licencias al servidor NetScaler ADM.
  NetScaler CPX License Aggregator puede gestionar la terminación no correcta de los de NetScaler CPX y reclamar licencias de dichos de NetScaler CPX después del período de espera configurado.

Topología de implementación

El siguiente diagrama muestra una implementación de NetScaler CPX License Aggregator dentro de un clúster de Kubernetes.



En este diagrama:

- `CPX` significa NetScaler CPX
- `CPX License Aggregator` significa el agregador de licencias NetScaler CPX

En esta implementación de ejemplo, se implementa un servicio NetScaler CPX License Aggregator dentro del clúster de Kubernetes junto con los de NetScaler CPX y el agente NetScaler ADM. El servicio NetScaler CPX License Aggregator actúa como intermediario entre los de NetScaler CPX y el agente NetScaler ADM y supervisa todos los de NetScaler CPX dentro del clúster y administra las licencias para ellos.

Implementar NetScaler CPX License Aggregator mediante gráficos Helm

Requisitos previos

Se aplican los siguientes requisitos previos:

- Necesitas la versión 1.16 y posteriores de Kubernetes.
- Necesitas la versión 3.x o posterior de Helm.
- Debe obtener la dirección IP del servidor de licencias que tiene la licencia para NetScaler CPX.
- Debe proporcionar una contraseña para la base de datos Redis en NetScaler CPX License Aggregator. Puedes proporcionar la contraseña de la base de datos con el secreto de Kubernetes y puedes usar el siguiente comando para crear el secreto:

```
kubectl create secret generic dbsecret --from-literal=password=<
    custom-password>
```

Implementación mediante Helm Charts

Siga los siguientes pasos para implementar el agregador de licencias NetScaler CPX mediante los gráficos de Helm según el tipo de licencia de NetScaler CPX. Para obtener más información sobre los diferentes tipos de licencias de NetScaler CPX, consulte Licencias de NetScaler CPX.

Instalación del gráfico Helm Agregue el repositorio de gráficos Helm de NetScaler CPX License Aggregator con el siguiente comando:

```
helm repo add Citrix https://citrix.github.io/citrix-helm-charts/
```

Instalación del agregador de licencias de NetScaler CPX para administrar las licencias agrupadas en ancho de banda Utilice uno de los siguientes comandos según el tipo de licencia agrupada de NetScaler CPX que tenga. En estos comandos, `my-release` se usa como nombre de la versión.

> **Nota:**
>
> De forma predeterminada, Helm Charts instala los roles y enlaces de roles de RBAC recomendados.

Para la licencia Platinum Bandwidth:


```
helm install my-release citrix/cpx-license-aggregator --set
   licenseServer.address=<License-Server-IP-or-FQDN>,redis.
   secretName=<Kubernetes-Secret-for-DB-password>,licenseAggregator
   .username=<unique-ID-for-CLA>,licenseInfo.instanceQuantum=<
   QUANTUM>,licenseInfo.instanceLowWatermark=<LOW WATERMARK>,
   licenseInfo.bandwidthPlatinumQuantum=<QUANTUM-in-Mbps>,
   licenseInfo.bandwidthPlatinumLowWatermark=<LOW WATERMARK-in-Mbps
   >
```

Para edición de ancho de banda empresarial:

```
helm install my-release citrix/cpx-license-aggregator --set
   licenseServer.address=<License-Server-IP-or-FQDN>,redis.
   secretName=<Kubernetes-Secret-for-DB-password>,licenseAggregator
   .username=<unique-ID-for-CLA>,licenseInfo.instanceQuantum=<
   QUANTUM>,licenseInfo.instanceLowWatermark=<LOW WATERMARK>,
   licenseInfo.bandwidthEnterpriseQuantum=<QUANTUM-in-Mbps>,
   licenseInfo.bandwidthEnterpriseLowWatermark=<LOW WATERMARK-in-
   Mbps>
```

Para edición de ancho de banda estándar:

```
helm install my-release citrix/cpx-license-aggregator --set
   licenseServer.address=<License-Server-IP-or-FQDN>,redis.
   secretName=<Kubernetes-Secret-for-DB-password>,licenseAggregator
   .username=<unique-ID-for-CLA>,licenseInfo.instanceQuantum=<
   QUANTUM>,licenseInfo.instanceLowWatermark=<LOW WATERMARK>,
   licenseInfo.bandwidthStandardQuantum=<QUANTUM-in-Mbps>,
   licenseInfo.bandwidthStandardLowWatermark=<LOW WATERMARK-in-Mbps
   >
```

Estos comandos implementan el agregador de licencias de NetScaler CPX en el clúster de Kubernetes con la configuración predeterminada. Puede configurar los parámetros en el momento de la instalación. Para obtener más información, consulte la sección de **configuración del agregador de licencias de NetScaler CPX** en el repositorio de GitHub de Helm Chart, que enumera los parámetros obligatorios y opcionales que puede configurar durante la instalación.

Instalación del agregador de licencias de NetScaler CPX para administrar las licencias de vCPU

Utilice uno de los siguientes comandos según el tipo de licencia de vCPU de NetScaler CPX que tenga. En estos comandos, `my-release` se usa como nombre de la versión.

Nota:
De forma predeterminada, el gráfico Helm instala las funciones de RBAC recomendadas y los enlaces de funciones.

Para la edición vCPU de platino:


```
helm install my-release citrix/cpx-license-aggregator --set
   licenseServer.address=<License-Server-IP-or-FQDN>,redis.
   secretName=<Kubernetes-Secret-for-DB-password>,licenseAggregator
   .username=<unique-ID-for-CLA>,licenseInfo.vcpuPlatinumQuantum=<
   QUANTUM>,licenseInfo.vcpuPlatinumLowWatermark=<LOW WATERMARK>
```

Para la edición de vCPU empresarial:

```
helm install my-release citrix/cpx-license-aggregator --set
   licenseServer.address=<License-Server-IP-or-FQDN>,redis.
   secretName=<Kubernetes-Secret-for-DB-password>,licenseAggregator
   .username=<unique-ID-for-CLA>,licenseInfo.vcpuEnterpriseQuantum
   =<QUANTUM>,licenseInfo.vcpuEnterpriseLowWatermark=<LOW WATERMARK
   >
```

Para la edición vCPU estándar:

```
helm install my-release citrix/cpx-license-aggregator --set
   licenseServer.address=<License-Server-IP-or-FQDN>,redis.
   secretName=<Kubernetes-Secret-for-DB-password>,licenseAggregator
   .username=<unique-ID-for-CLA>,licenseInfo.vcpuStandardQuantum=<
   QUANTUM>,licenseInfo.vcpuStandardLowWatermark=<LOW WATERMARK>
```

Instalación del agregador de licencias de NetScaler CPX para administrar varias licencias

Si necesita el agregador de licencias de NetScaler CPX para administrar varios tipos de licencias, los argumentos relevantes de esas licencias deben especificarse en el comando Helm.

Por ejemplo:

Para implementar el agregador de licencias `pooled platinum bandwidth edition` y `vCPU platinum edition` licencias de NetScaler CPX:

```
helm install demo citrix/cpx-license-aggregator --set
   licenseServer.address=<License-Server-IP-or-FQDN>,redis.
   secretName=<Kubernetes-Secret-for-DB-password>,
   licenseAggregator.username=<unique-ID-for-CLA>,licenseInfo.
   instanceQuantum=<QUANTUM>,licenseInfo.instanceLowWatermark=<
   LOW WATERMARK>,licenseInfo.bandwidthPlatinumQuantum=<QUANTUM-
   in-Mbps>,licenseInfo.bandwidthPlatinumLowWatermark=<LOW
   WATERMARK-in-Mbps>,licenseInfo.vcpuPlatinumQuantum=<QUANTUM>,
   licenseInfo.vcpuPlatinumLowWatermark=LOW WATERMARK>
```

Configuración de NetScaler CPX para obtener una licencia de NetScaler CPX License Aggregator

Cuando se usa NetScaler CPX License Aggregator para obtener licencias de NetScaler CPX, la variable de entorno `CLA` debe proporcionarse en el YAML de implementación de NetScaler CPX.

La `ipaddress` o el `domainname` mediante el que se puede acceder al Agregador de licencias NetScaler CPX se debe proporcionar en esta variable de entorno de la siguiente manera:

```
        env:
        - name: "CLA"
          value: "192.0.2.2"
```

O bien:

```
        env:
        - name: "CLA"
          value: "local-cla.org"
```

También debe proporcionar las siguientes variables de entorno en el NetScaler CPX YAML.

- `POD_NAME`: Especifica el nombre del pod. El nombre del pod se expone a NetScaler CPX como una variable de entorno.
- `POD_NAMESPACE`: Especifica el espacio de nombres del pod. El espacio de nombres del pod se expone a NetScaler CPX como una variable de entorno.
- `Bandwidth`: Especifica el ancho de banda en Mbps para asignarlo a NetScaler CPX.
- `Edition`: Especifica la edición de la licencia. Los valores admitidos incluyen Standard, Platinum y Enterprise.
- `CPX_CORES`: Especifica la cantidad de núcleos que quiere ejecutar para NetScaler CPX.

Para obtener más información sobre las diferentes opciones de licencia de NetScaler CPX, consulte Licencias de NetScaler CPX.

A continuación, se muestra una configuración de ejemplo con estas variables de entorno:

```
        - name: POD_NAME
          valueFrom:
            fieldRef:
              apiVersion: v1
              fieldPath: metadata.name
        - name: POD_NAMESPACE
          valueFrom:
            fieldRef:
              apiVersion: v1
              fieldPath: metadata.namespace
        - name:  " BANDWIDTH "
          value: 1000
        - name:  " CPX_CORES "
          value: 1
        - name:  " EDITION "
          value: PLATINUM
```

También debe agregar la siguiente etiqueta al NetScaler CPX YAML:

```
      labels:
        adc: citrix
```


Para ver un ejemplo de implementación de NetScaler CPX License Aggregator, consulte NetScaler CPX License Aggregator: Sample deployment.

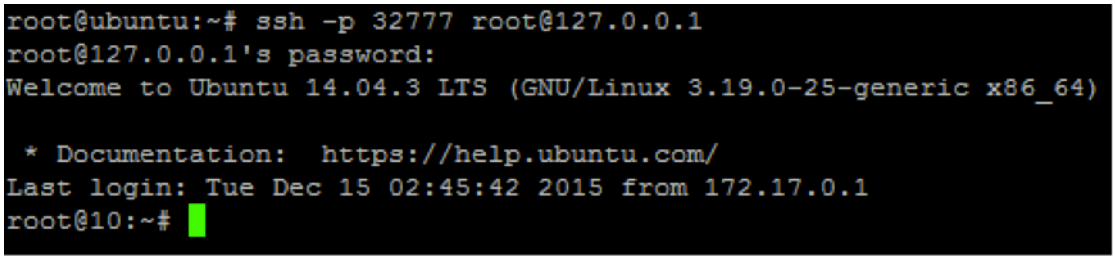
Configuración de NetScaler CPX

November 23, 2023

Puede configurar una instancia de NetScaler CPX accediendo a la solicitud de la CLI a través del host de Linux Docker o mediante las API de NetScaler NITRO.

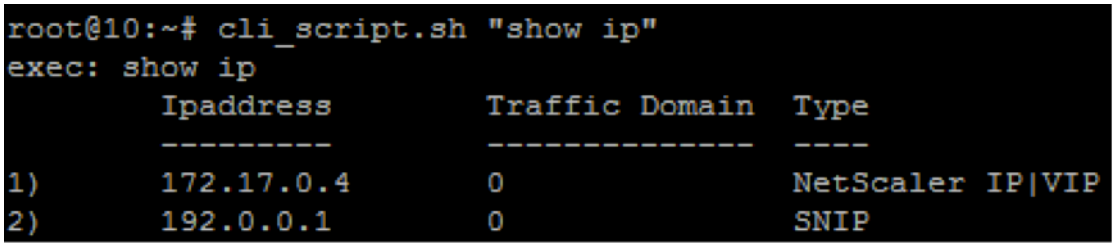
Configuración de una instancia de NetScaler CPX mediante la interfaz de línea de comandos

Acceda al host de Docker e inicie sesión en el indicador SSH de la instancia, como se muestra en esta ilustración. Las credenciales de administrador predeterminadas para iniciar sesión en una instancia de NetScaler CPX son root/linux.



Escriba el siguiente comando para usar el símbolo de línea de comandos de la instancia para ejecutar los comandos de la CLI: **cli_script.sh** "<comando>"

Ejemplo:



Para cerrar sesión en el símbolo de la instancia, escriba **logout**.

Compatibilidad con el uso de una contraseña no predeterminada en NetScaler CPX

NetScaler CPX admite el uso de una contraseña no predeterminada para la cuenta raíz, es decir `nsroot`. Se genera una contraseña predeterminada y se asigna al usuario una vez que se imple-

menta NetScaler CPX. Esta contraseña predeterminada también se actualiza para los usuarios de SSH: `root` y `nsroot`. Puede cambiar esta contraseña predeterminada de forma manual. También puede restablecer la contraseña SSH predeterminada para las cuentas de usuario `root` y `nsroot` de forma manual. Citrix recomienda cambiar esta contraseña manualmente para preservar la seguridad del sistema.

Una vez que restablezca su contraseña, la nueva contraseña se utilizará para las comunicaciones y ejecuciones de `cli_script.sh` de la API de NITRO.
La contraseña de la cuenta raíz predeterminada se almacena en texto sin formato en el archivo */var/deviceinfo/random_id* en el sistema de archivos de NetScaler CPX.

Use la siguiente sintaxis para ejecutar `cli_script.sh` con las credenciales:

```
cli_script.sh "<command>"":<user>:<password>"
```

Por ejemplo, `cli_script.sh` para ejecutar y mostrar direcciones IP con usuario `nsroot` y contraseña `Citrix123`, use lo siguiente:

```
1     cli_script.sh "show ns ip" ":nsroot:Citrix123"
```

Configuración de una instancia de NetScaler CPX mediante la API de NITRO

Puede usar la API NetScaler NITRO para configurar instancias de NetScaler CPX.

Para configurar instancias de NetScaler CPX mediante la API de Nitro, en un explorador web, escriba:

```
http://<host_IP_address>:<port>/nitro/v1/config/<resource-type\
```

Para recuperar estadísticas mediante la API de Nitro, en un explorador web, escriba:

```
http://\<host\\_IP\\_address\>:\<port\>/nitro/v1/stat/\<resource-type
\
```

Para obtener más información sobre el uso de la API de NITRO, consulte Servicios web REST. Para NetScaler CPX, utilice `CPX IP address:port` donde `netscaler-ip-address` se menciona.

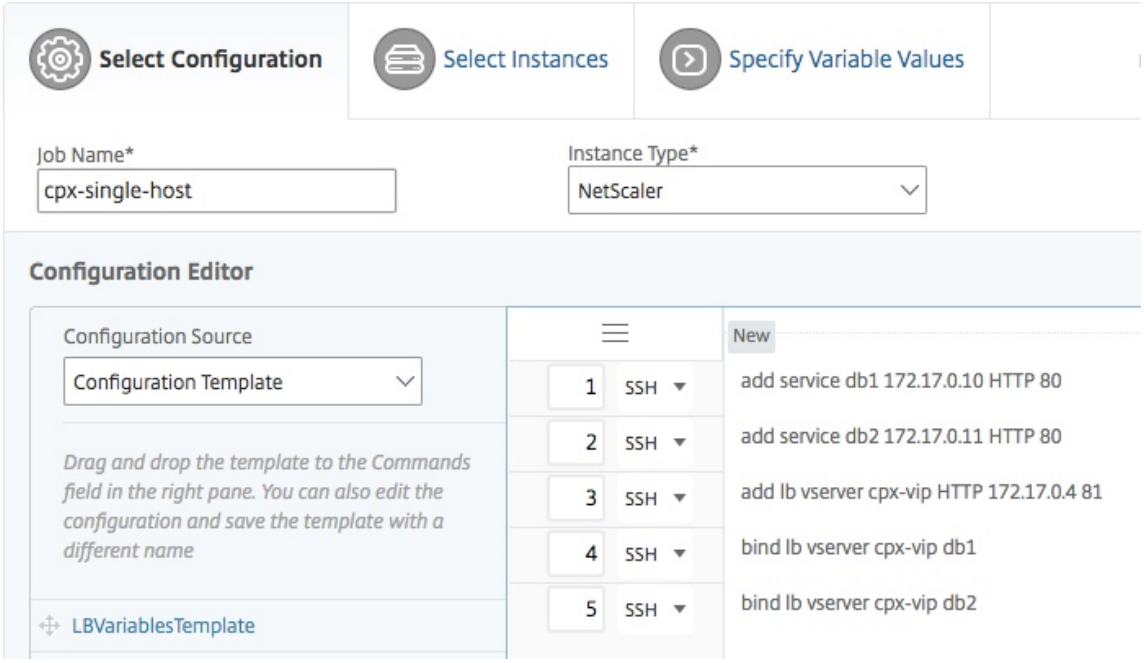
Configuración de una instancia de NetScaler CPX mediante trabajos

Puede configurar instancias de NetScaler CPX creando y ejecutando trabajos en NetScaler ADM. Puede usar las configuraciones de las plantillas de configuración, extraer configuraciones disponibles en otros dispositivos y usar configuraciones guardadas en archivos de texto. También puede registrar las configuraciones realizadas mediante la utilidad de configuración de otras instancias. A continuación, NetScaler ADM muestra los comandos de la CLI correspondientes para que los utilice en la

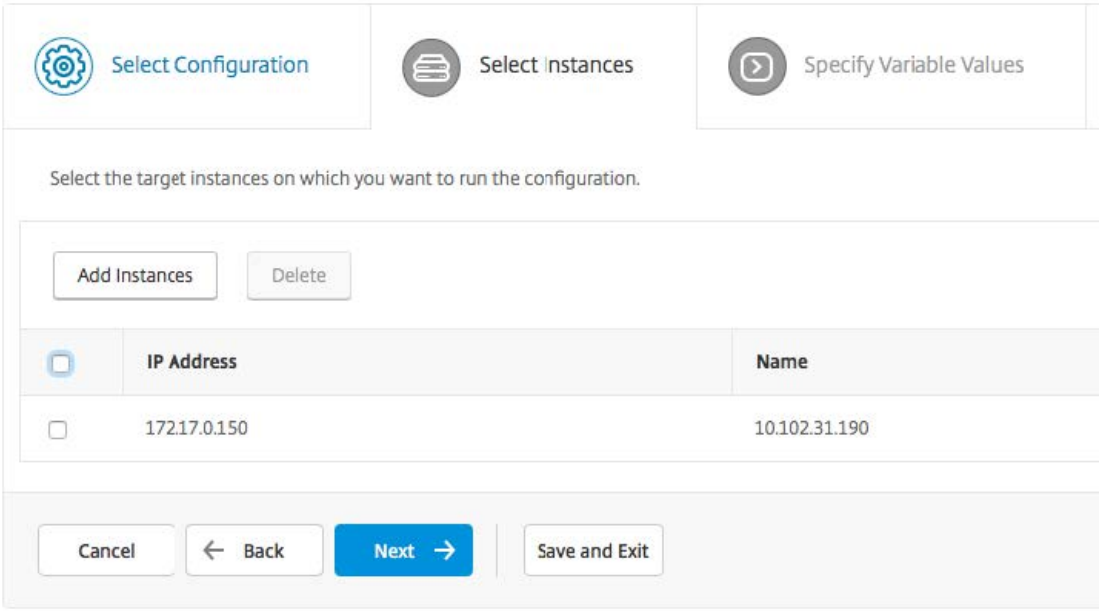
instancia de NetScaler CPX. Después de seleccionar la configuración, debe seleccionar las **instancias de NetScaler CPX en las** que quiere cargar la configuración, especificar los valores de las variables y ejecutar el trabajo.

Para configurar instancias de NetScaler CPX mediante Trabajos:

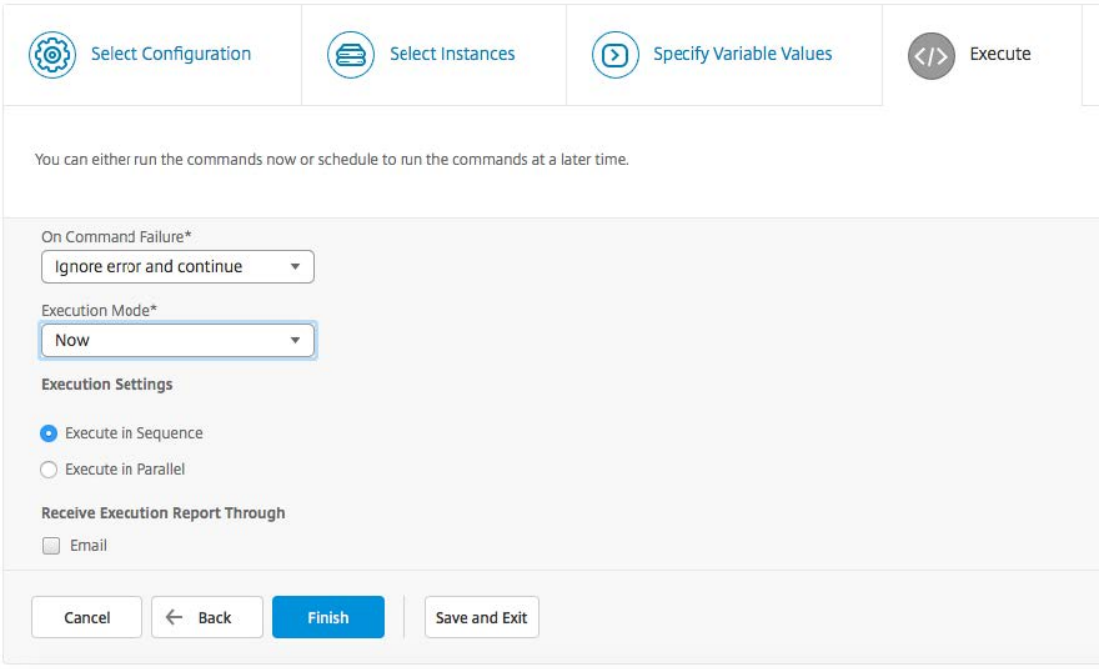
1. Inicie sesión en NetScaler ADM con las credenciales administrativas.
2. Vaya a **Redes > Trabajos de configuración**y, a continuación, haga clic en **Crear trabajo**.
3. Especifique los valores requeridos y seleccione el origen de configuración. También puede escribir los comandos que quiere ejecutar.



4. Seleccione las instancias de NetScaler CPX en las que quiera ejecutar la configuración y haga clic en **Siguiente**.



5. Especifique la configuración de ejecución y haga clic en Finalizar para ejecutar los comandos en la instancia de NetScaler CPX. Si quiere guardar la configuración y ejecutarla más tarde, haga clic en **Guardar y salir**.



Configuración de AppFlow en una instancia de NetScaler CPX

November 23, 2023

Puede configurar la función AppFlow en una instancia de NetScaler CPX para recopilar datos de rendimiento de páginas web, información a nivel de flujo y sesión de usuario e información de base de datos necesaria para la supervisión y el análisis del rendimiento de las aplicaciones. Estos registros de datos se envían a NetScaler ADM, donde puede ver informes históricos y en tiempo real de todas sus aplicaciones.

Para configurar AppFlow, primero, debe habilitar la función AppFlow. A continuación, especifique los recopiladores a los que se envían los registros de flujo. Después de eso, defina acciones, que son conjuntos de recopiladores configurados. A continuación, configure una o varias directivas y asocie una acción a cada directiva. La directiva indica a NetScaler CPX que seleccione las solicitudes cuyos registros de flujo se envían a la acción asociada. Por último, vincula cada directiva globalmente o al servidor virtual específico para ponerla en práctica.

Además, puede configurar los parámetros de AppFlow para especificar el intervalo de actualización de la plantilla y para permitir la exportación de `httpURL`, `httpCookie` y la información de `httpReferer`. En cada recopilador, debe especificar la dirección IP de NetScaler CPX como la dirección del exportador.

La utilidad de configuración proporciona herramientas que ayudan a los usuarios a definir las directivas y acciones. Determina exactamente cómo NetScaler CPX exporta los registros de un flujo determinado a un conjunto de recopiladores (acción). La interfaz de línea de comandos proporciona un conjunto correspondiente de comandos basados en CLI para usuarios experimentados que prefieren una línea de comandos.

Para poder supervisar los registros, debe agregar la instancia de NetScaler CPX a NetScaler ADM. Para obtener más información sobre cómo agregar una instancia de NetScaler CPX a NetScaler ADM, consulte Instalación de una instancia de NetScaler CPX mediante NetScaler ADM.

Habilitar AppFlow

Para usar la función AppFlow, primero debe habilitarla.

Para habilitar la función AppFlow mediante la interfaz de línea de comandos:

Ejecute los comandos siguientes:

```
enable ns feature AppFlow
enable ns mode ulfd
```


Especificar un recopilador

Un recopilador recibe registros de AppFlow generados por NetScaler. Para enviar los registros de AppFlow, debe especificar al menos un recopilador. De forma predeterminada, el recopilador escucha los mensajes IPFIX en el puerto UDP 4739. Puede cambiar el puerto predeterminado al configurar el selector.

Para especificar un recopilador mediante la interfaz de línea de comandos:

Use los siguientes comandos para agregar un recopilador:

```
add appflow collector <name> -IPAddress <ipaddress> -port <port_number>
     -netprofile <netprofile_name> -Transport Logstream
```

Para verificar la configuración, use el siguiente comando:

```
show appflow collector <name>
```

Para especificar varios recopiladores mediante la interfaz de línea de comandos:

Use los siguientes comandos para agregar y enviar los mismos datos a varios recopiladores:

```
add appflow collector <collector1> -IPAddress <IP> -Transport Logstream

add appflow collector <collector2> -IPAddress <IP> -Transport Logstream

add appflow action <action> -collectors <collector1> <collector2> -
   Transport Logstream

add appflow policy <policy> true <action> -Transport Logstream

bind lbvserver <lbvserver> -policy <policy> -priority <priority> -
   Transport Logstream
```

Configuración de una acción de AppFlow

Una acción AppFlow es un recopilador de conjuntos, al que se envían los registros de flujo si coincide la directiva AppFlow asociada.

Use los siguientes comandos para configurar una acción de AppFlow:

```
add appflow action <name> --collectors <string> ... \[-
   clientSideMeasurements \(Enabled|Disabled) ] \[-comment <string>]
```

Para verificar la configuración, use el siguiente comando:

```
show appflow action
```


Configuración de una directiva de AppFlow

Después de configurar una acción de AppFlow, debe configurar una directiva de AppFlow. Una directiva de AppFlow se basa en una regla, que consta de una o más expresiones.

Para configurar una directiva de AppFlow mediante la interfaz de línea de comandos:

En el símbolo del sistema, escriba el siguiente comando para agregar una directiva de AppFlow y verificar la configuración:

```
add appflow policy <name> <rule> <action>

show appflow policy <name>
```

Vinculación de una directiva de AppFlow

Para poner en vigor una directiva, debe vincularla de forma global, de modo que se aplique a todo el tráfico que fluye a través de NetScaler CPX.

Para enlazar de forma global una directiva de AppFlow mediante la interfaz de línea de comandos:

Use el siguiente comando para vincular de forma global una directiva de AppFlow:

```
bind appflow global <policyName> <priority> [<gotoPriorityExpression [-
    type <type>] [-invoke (<labelType> <labelName>)]
```

Verifique la configuración mediante el siguiente comando:

```
show appflow global
```

Configuración de NetScaler CPX mediante un archivo de configuración

November 23, 2023

En lugar de utilizar la interfaz de línea de comandos (`cli_script.sh`), la API de NITRO o los trabajos de configuración de NetScaler ADM para configurar NetScaler CPX, puede configurar NetScaler CPX mediante un archivo de configuración estática al implementar la instancia de NetScaler CPX.

Puede proporcionar un archivo de configuración estática como archivo de entrada mientras implementa el contenedor NetScaler CPX. Durante el inicio del contenedor NetScaler CPX, el contenedor se configura en función de la configuración especificada en el archivo de configuración estática. Esta configuración incluye la configuración específica de NetScaler y los comandos bash shell que se pueden ejecutar dinámicamente en el contenedor NetScaler CPX.

Estructura del archivo de configuración estática

Como se mencionó anteriormente, cuando se implementa NetScaler CPX, se configura en función de las configuraciones especificadas en el archivo de configuración estática.

El archivo de configuración estática es un archivo `.conf` que incluye dos etiquetas `#NetScaler Commands` y `#Shell Commands`. Debajo de la etiqueta `#NetScaler Commands`, debe agregar todos los comandos de NetScaler para configurar la configuración específica de NetScaler en NetScaler CPX. Debajo de la etiqueta `#Shell Commands`, debe agregar los comandos de shell que quiere ejecutar en NetScaler CPX.

Durante la implementación del contenedor NetScaler CPX, los comandos de NetScaler y los comandos de shell se ejecutan en el contenedor en el orden especificado en el archivo de configuración.

> **Importante:**
>
> - Las etiquetas se pueden repetir varias veces en el archivo de configuración.
> - Las etiquetas no distinguen entre mayúsculas y minúsculas.
> - El archivo de configuración debe estar presente en el directorio `/etc` como archivo `cpx.conf` en el sistema de archivos del contenedor.
> - El archivo de configuración también puede incluir comentarios. Debe agregar un carácter "**#**"antes de sus comentarios.
> - Si hay casos de error al implementar el contenedor NetScaler CPX con el archivo de configuración, los errores se registran en el archivo `ns.log` en el contenedor.
> - Al reiniciar el contenedor NetScaler CPX, el archivo de configuración se vuelve a aplicar en el contenedor.

```
#NetScaler Commands

add lb vserver v1 http 1.1.1.1 80

add service s1 2.2.2.2 http 80

bind lb vserver v1 s1

#Shell Commands

touch /etc/a.txt

echo "this is a" > /etc/a.txt

#NetScaler Commands

add lb vserver v2 http

#Shell Commands

echo "this is a 1" >> /etc/a.txt
```



```
22
23 #NetScaler Commands
24
25 add lb vserver v3 http
26
27 #This is a test configuration file
28 <!--NeedCopy-->
```

Para instalar un contenedor NetScaler CPX y configurar dinámicamente el contenedor NetScaler CPX en función de un archivo de configuración, monte el archivo de configuración estática mediante la opción – v del comando `docker run`:

```
1 docker run -dt --privileged=true -e EULA=yes --ulimit core=-1 -v /tmp/
     cpx.conf:/etc/cpx.conf --name mycpx store/citrix/citrixadccpx:13.0-x
     .x
2 <!--NeedCopy-->
```

Compatibilidad con redirección dinámica en NetScaler CPX

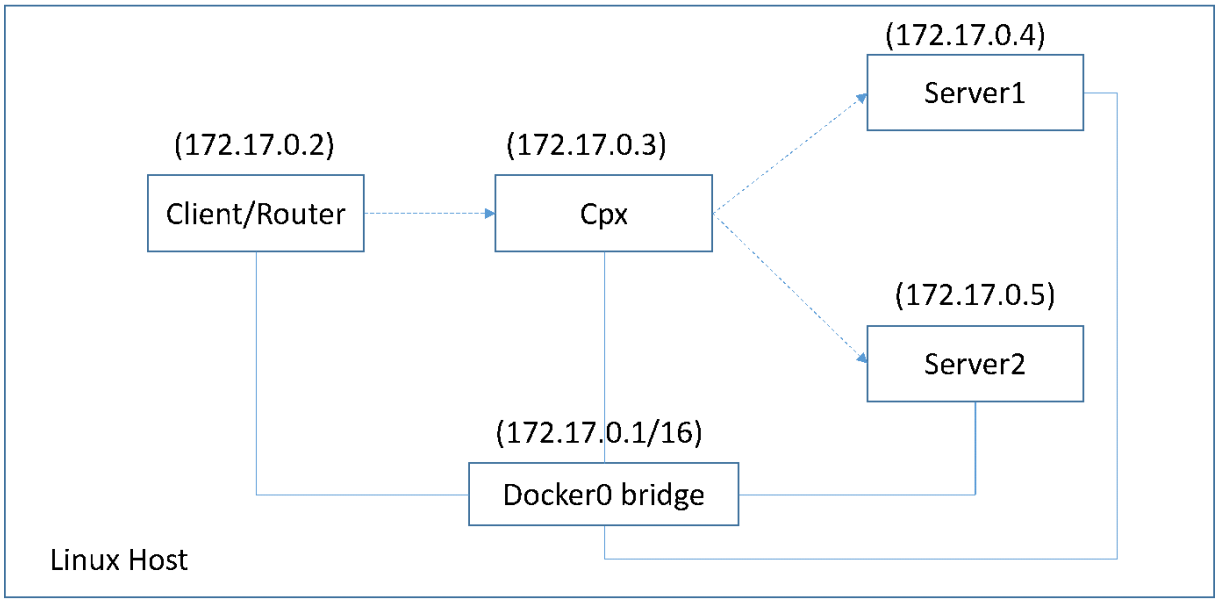
November 23, 2023

NetScaler CPX admite el protocolo de redirección dinámica BGP. El objetivo clave del protocolo de redirección dinámica es anunciar la dirección IP del servidor virtual en función del estado de los servicios, vinculados al servidor virtual. Ayuda a un enrutador ascendente a elegir la mejor entre varias rutas a un servidor virtual distribuido topográficamente.

Para obtener información sobre la contraseña no predeterminada en NetScaler CPX, consulte la sección `Support for using a non-default password in NetScaler CPX` del documento Configuración de NetScaler CPX.

En una red de host único, el cliente, los servidores y la instancia de NetScaler CPX se implementan como contenedores en el mismo host de Docker. Todos los contenedores están conectados a través del puente docker0. En este entorno, la instancia de NetScaler CPX actúa como proxy para las aplicaciones aprovisionadas como contenedores en el mismo host de Docker. Para obtener información sobre la implementación en modo de red de host de NetScaler CPX, consulte Modo de red de host.

Esta ilustración ilustra la topología de host único.



En esta topología, los servidores virtuales se configuran y anuncian (según el estado de los servicios) a la red o enrutador ascendente mediante BGP.

Realice los siguientes pasos para configurar BGP en NetScaler CPX en un solo host Docker con el modo de conexión en red puente.

Configurar la inyección de estado de ruta basada en BGP mediante la API REST en NetScaler CPX

1. Cree un contenedor a partir de la imagen de NetScaler CPX mediante el siguiente comando:

```
docker run -dt --privileged=true -p 22 -p 80 -p 161 -e EULA=yes --
    ulimit core=-1 cpx: <tag>
```

   Por ejemplo:

```
docker run -dt --privileged=true -p 22 -p 80 -p 161 -e EULA=yes --
    ulimit core=-1 cpx:12.1-50.16
```

2. Inicie sesión en el contenedor con el siguiente comando:

```
docker exec -it <container id> bash
```

3. Habilite la función BGP con el siguiente comando:

```
cli_script.sh "enable ns feature bgp"
```

4. Obtenga el NSIP mediante el comando `show ns ip`:

```
cli_script.sh "show ns ip"
```


5. Agregue el servidor virtual mediante el siguiente comando:

```
cli_script.sh "add lb  vserver <vserver_name> http <VIP> <PORT>"
```

6. Agregue servicios y enlace servicios al servidor virtual.
7. Habilite `hostroute` para el VIP mediante el siguiente comando:

```
cli_script.sh "set ns ip <VIP> -hostroute enabled ”
```

Salga del contenedor y envíe comandos BGP NITRO desde el host al NSIP en el puerto 9080.

8. Configure el enrutador BGP:

Por ejemplo, si quieres configurar:

```
router bgp 100
Neighbour 172.17.0.2 remote-as 101
Redistribute kernel
```

Especifique el comando de la siguiente manera:

```
 curl -u username:password http://<NSIP>:9080/nitro/v1/config/ -X
    POST --data 'object={
"routerDynamicRouting": {
"bgpRouter" : {
"localAS":100, "neighbor": [{
 "address": "172.17.0.2", "remoteAS": 101 }
], "afParams":{
"addressFamily": "ipv4", "redistribute": {
"protocol": "kernel" }
 }
 }
 }
 }
'
```

9. Instale las rutas BGP aprendidas en el PE mediante el siguiente comando NITRO:

```
curl -u username:password http://<NSIP>:9080/nitro/v1/config/ --
   data 'object={
 "params":{
 "action":"apply" }
 ,"routerDynamicRouting": {
 "commandstring" : "ns route-install bgp" }
 }
 '
```

10. Verifique el estado de adyacencia de BGP mediante el siguiente comando NITRO:

```
curl -u username:password http://<NSIP>:9080/nitro/v1/config/
   routerDynamicRouting/bgpRouter
```

Salida de muestra:


```
root@ubuntu:~# curl -u username:password http://172.17.0.3:9080/
   nitro/v1/config/routerDynamicRouting/bgpRouter
{
 "errorcode": 0, "message": "Done", "severity": "NONE", "
    routerDynamicRouting":{
"bgpRouter":[{
 "localAS": 100, "routerId": "172.17.0.3", "afParams": [ {
 "addressFamily": "ipv4"  }
, {
 "addressFamily": "ipv6"  }
 ], "neighbor": [ {
 "address": "172.17.0.2", "remoteAS": 101, "ASOriginationInterval
    ": 15, "advertisementInterval": 30, "holdTimer": 90, "
    keepaliveTimer": 30, "state": "Connect", "singlehopBfd":
    false, "multihopBfd": false, "afParams": [ {
 "addressFamily": "ipv4", "activate": true  }
, {
 "addressFamily": "ipv6", "activate": false  }
 ]
```

11. Compruebe que las rutas aprendidas a través de BGP estén instaladas en el motor de paquetes con el siguiente comando:

```
cli_script.sh " show route "
```

12. Guarde la configuración mediante el siguiente comando:

```
cli_script.sh " save config "
```

La configuración de redirección dinámica se guarda en el archivo `/nsconfig/ZebOS.conf`.

Configuración de controladores de registro de Docker

November 23, 2023

Docker incluye mecanismos de registro denominados "controladores de registro"para ayudarlo a obtener información de los contenedores en ejecución. Puede configurar un contenedor NetScaler CPX para que reenvíe los registros que genere a los controladores de registro de Docker. Para obtener más información sobre los controladores de registro de docker, consulte Configurar controladores de registro.

De forma predeterminada, todos los registros generados por el contenedor NetScaler CPX se almacenan en un archivo `/cpx/log/ns.log` en el host de la ventana acoplable. Cuando inicia el contenedor NetScaler CPX con el comando docker run, puede configurarlo para que reenvíe todos los registros generados a un controlador de registro de docker mediante la opción `--log-driver`. Si

el controlador de registro tiene parámetros configurables, puede configurarlos mediante la opción `--log-opt <NAME>=<VALUE>`.

En el siguiente ejemplo, el contenedor NetScaler CPX está configurado para reenviar todos los registros generados mediante syslog como controlador de registro.

```
1 docker run -dt --privileged=true --log-driver syslog --log-opt syslog-
     address=udp://10.106.102.190:514 -e EULA=yes --ulimit core=-1 --name
      test store/citrix/cpx:12.1-48.13
2 <!--NeedCopy-->
```

Del mismo modo, en el siguiente ejemplo, el contenedor NetScaler CPX está configurado para reenviar todos los registros generados mediante Splunk como controlador de registro.

```
1 docker run -dt --privileged=true --log-driver=splunk --log-opt splunk-
     token=176FCEBF-4CF5-4EDF-91BC-703796522D20 --log-opt splunk-url=
     https://splunkhost:8088 -e EULA=yes --ulimit core=-1 --name test
     store/citrix/cpx:12.1-48.13
2 <!--NeedCopy-->
```

Actualización de una instancia de NetScaler CPX

November 23, 2023

Para actualizar una instancia de NetScaler CPX, apáguela, instale la versión más reciente en el mismo punto de montaje y, a continuación, elimine la instancia anterior. Un punto de montaje es un directorio en el que se monta el directorio **/cpx** en el host.

Por ejemplo, para montar el directorio **/cpx** de la instancia de NetScaler CPX existente en el directorio **/var/cpx** del host, el punto de montaje es **/var/cpx** y el directorio de montaje de NetScaler CPX es **/cpx**, como se muestra a continuación:

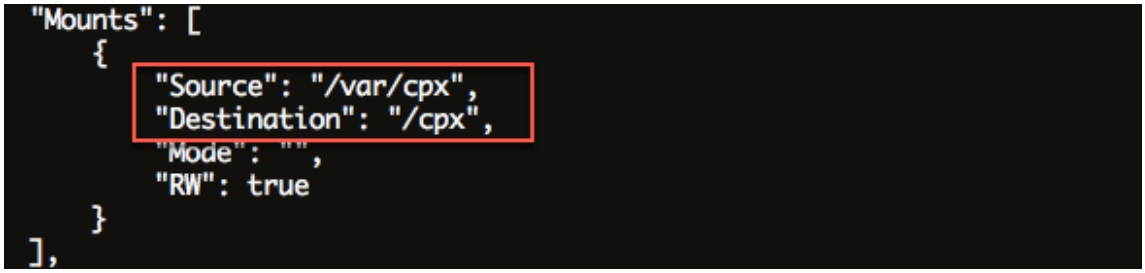
```
1     root@ubuntu:~# docker run -dt -e EULA=yes --name mycpx -v /var/cpx
         :/cpx --ulimit core=-1 cpx:13.0-x.x
2 <!--NeedCopy-->
```

Requisitos previos

Asegúrese de lo siguiente:

- Detalles del directorio host en el que montó el directorio `/cpx` de la instancia de NetScaler CPX existente. Puede usar el comando `docker inspect <containerName>`, donde `<containerName>` es el nombre del contenedor NetScaler CPX, para mostrar información sobre el directorio host.

El resultado del comando proporciona los detalles de las configuraciones del contenedor, incluidos los volúmenes. En la entrada "**Mounts**", la subentrada"**Source**"muestra la ubicación del directorio del host en el host.



- Descargue el archivo de imagen Docker de NetScaler CPX más reciente y cargue la imagen de Docker de NetScaler CPX. Para cargar la imagen, navega hasta el directorio en el que guardaste el archivo de imagen de Docker. Use el comando `docker load -i <image_name>` para cargar la imagen. Después de cargar la imagen de NetScaler CPX, puede introducir el comando docker images para mostrar información sobre la imagen:

```
1     root@ubuntu:~# docker load -i  cpx-13.0-x.x.gz
2
3     root@ubuntu:~# docker images
4
5     REPOSITORY   TAG         IMAGE ID         CREATED     VIRTUAL SIZE
6
7     cpx          13.0-x.x  2e97aadf918b  43 hours ago      414.5 MB
8  <!--NeedCopy-->
```

Para actualizar una instancia de NetScaler CPX

1. Detenga la instancia de NetScaler CPX existente introduciendo el comando `docker stop <containerName>`, donde `<containerName>` es el nombre de la instancia de NetScaler CPX.

   ```
   1  root@ubuntu:~# docker stop mycpx
   2  mycpx
   3  <!--NeedCopy-->
   ```

2. Con el comando `docker run`, implemente la última instancia de NetScaler CPX desde la imagen de NetScaler CPX que cargó en el host. Asegúrese de implementar la instancia en el mismo punto de montaje (por ejemplo, `/var/cpx:/cpx`) que usó para la instancia de NetScaler CPX existente.

   ```
   1  root@ubuntu:~# docker run -dt -P -e CPX_CORES=1 --name latestcpx
         --ulimit core=-1 -e EULA=yes -v /var/cpx:/cpx --cap-add=
         NET_ADMIN cpx:13.0-x.x
   2  <!--NeedCopy-->
   ```


Puede introducir el comando `docker ps` para comprobar que la instancia de NetScaler CPX implementada es la versión más reciente.

```
1 ```
2     root@ubuntu:~# docker ps
3
4     CONTAINER ID        IMAGE               COMMAND
         CREATED             STATUS                          PORTS
                          NAMES
5
6     ead12ec4e965   cpx:13.0-x.x  "/bin/sh -c 'bash -C "   5
         seconds ago       Up 5 seconds        22/tcp, 80/tcp, 443/
         tcp, 161/udp   latestcpx
7 <!--NeedCopy--> ```
```

3. Después de comprobar que implementó la instancia CPX correcta de NetScaler, introduzca el comando **docker rm** <containerName> para eliminar la instancia anterior.

```
1 root@ubuntu:~# docker rm mycpx
2 mycpx
3 <!--NeedCopy-->
```

Uso de servidores virtuales comodín en la instancia de NetScaler CPX

November 23, 2023

Cuando aprovisiona una instancia de NetScaler, el motor de Docker solo asigna una dirección IP privada (una dirección IP) a una instancia de NetScaler CPX. Las tres funciones IP de una instancia de NetScaler se multiplexan en una dirección IP. Esta única dirección IP utiliza diferentes números de puerto para funcionar como NSIP, SNIP y VIP.

La dirección IP única que asigna el motor de Docker es dinámica. Agregue los servidores virtuales de equilibrio de carga (LB) o conmutación de contenido (CS) mediante la dirección IP única o la dirección IP 127.0.0.1. Los servidores virtuales creados con 127.0.0.1 se denominan Servidores virtuales comodín. De forma predeterminada, cuando crea un servidor virtual comodín, NetScaler CPX reemplaza la dirección IP asignada del servidor virtual comodín. La dirección IP asignada es 127.0.0.1, que se reemplaza por el NSIP asignado a la instancia de NetScaler CPX por el motor Docker.

En implementaciones de NetScaler CPX de alta disponibilidad, puede agregar servidores virtuales comodín en la instancia principal de NetScaler CPX. La sincronización de configuración entre nodos configura el servidor virtual comodín en la instancia secundaria de NetScaler CPX. Elimina la necesidad de configurar el servidor virtual en el NSIP asignado por el motor Docker a las instancias de NetScaler CPX.

Puntos a tener en cuenta:

- Asegúrese de que el número de puerto que asigna al servidor virtual comodín no lo utilice ningún otro servidor virtual de la implementación.
- La incorporación de servidor virtual comodín falla si el número de puerto que asigna al servidor virtual comodín ya está siendo utilizado por los servicios internos.
- El servidor virtual comodín no admite el carácter *.

Para crear un servidor virtual de equilibrio de carga comodín, en el símbolo del sistema, introduzca el siguiente comando:

```
1     add lb vserver <name> <serviceType> 127.0.0.1 <port>
2
3     add lb vserver testlbvserver HTTP 127.0.0.1 30000
4 <!--NeedCopy-->
```

Para crear un servidor virtual de conmutación de contenido comodín, en el símbolo del sistema, escriba el siguiente comando:

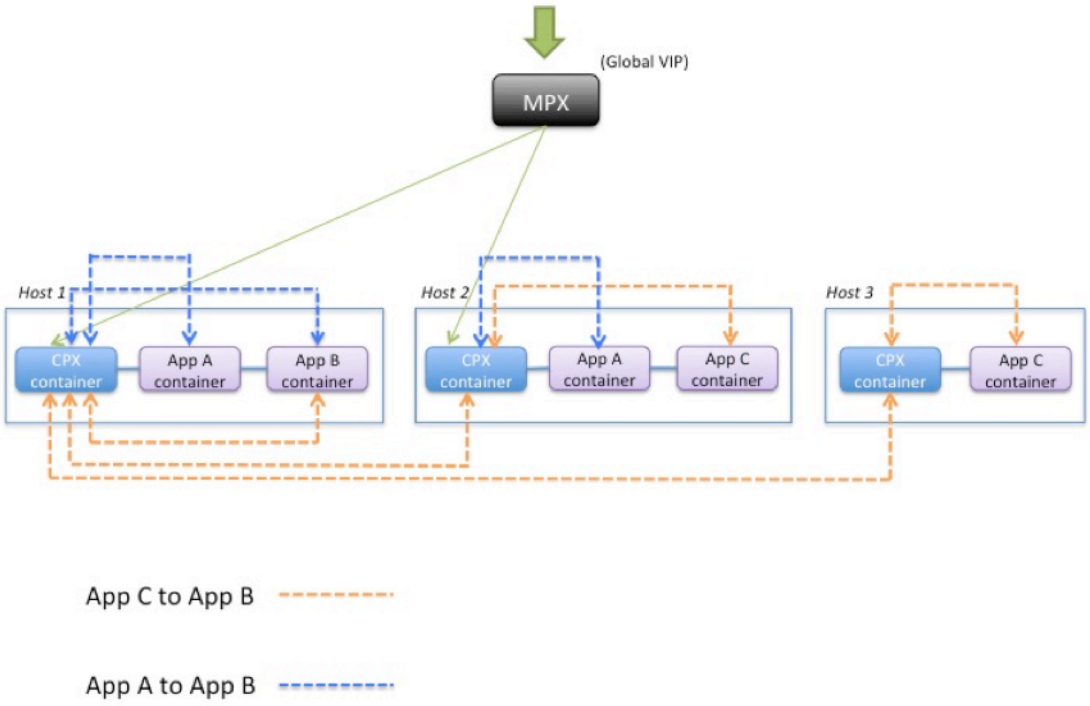
```
1     add cs vserver <name> <serviceType> 127.0.0.1 <port>
2
3     add cs vserver testcsvserver HTTP 127.0.0.1 30000
4 <!--NeedCopy-->
```

Implementar NetScaler CPX como proxy para permitir el flujo de tráfico de este a oeste

November 23, 2023

En esta implementación, la instancia de NetScaler CPX actúa como un proxy para permitir la comunicación entre los contenedores de aplicaciones que residen en varios hosts. La instancia de NetScaler CPX se aprovisiona junto con las aplicaciones en varios hosts y proporciona la ruta más corta para la comunicación.

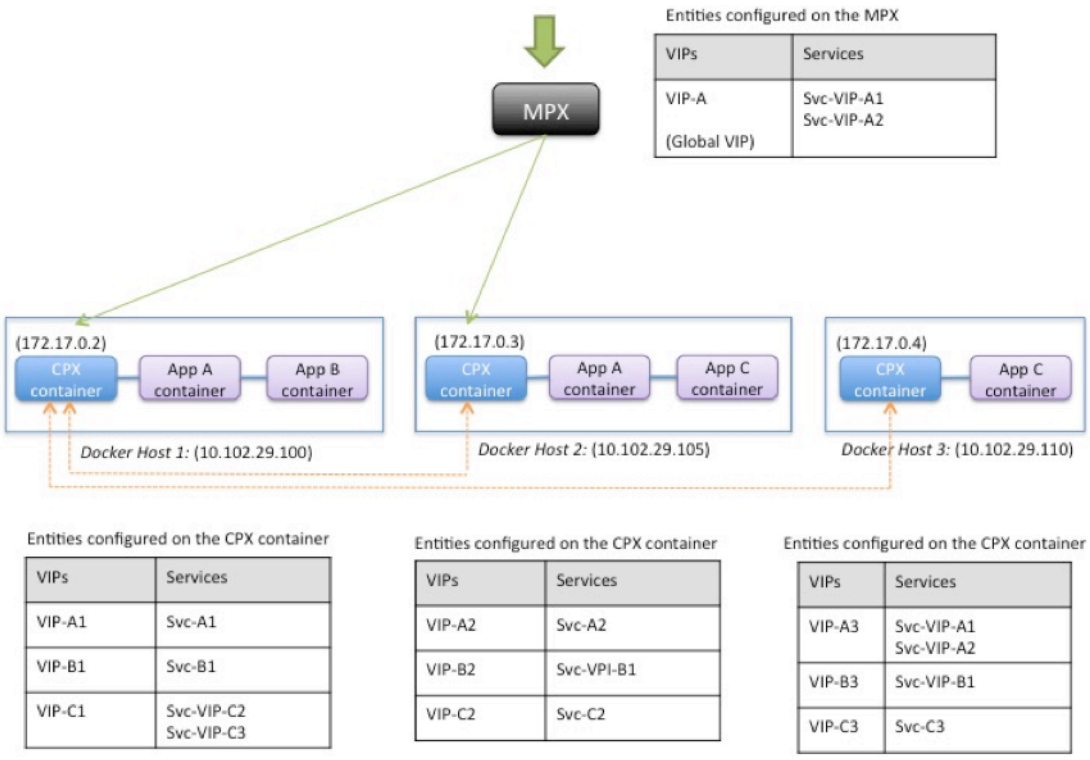
La siguiente imagen ilustra el flujo de tráfico entre dos aplicaciones a través de las instancias de NetScaler CPX.



Esta imagen muestra el flujo de tráfico entre la aplicación C y la aplicación B y entre la aplicación A y la aplicación B. Cuando la aplicación C (en cualquiera de los hosts) envía una solicitud a B, la solicitud se recibe primero en el contenedor NetScaler CPX en el mismo host que la aplicación C. A continuación, el contenedor NetScaler CPX pasa el tráfico al contenedor NetScaler CPX alojado en el mismo host que la aplicación B y, a continuación, el tráfico se reenvía a la aplicación B. Se sigue una ruta de tráfico similar cuando la aplicación A envía una solicitud a la aplicación B.

En este ejemplo, también se implementa un NetScaler MPX para permitir el tráfico a las aplicaciones desde Internet a través de un VIP global. El tráfico de NetScaler MPX se recibe en los contenedores de NetScaler CPX, que luego distribuyen el tráfico entre los contenedores de aplicaciones.

El siguiente diagrama ilustra esta topología con las configuraciones que deben establecerse para que se produzca la comunicación.



En la siguiente tabla se enumeran las direcciones IP y los puertos que se configuran en las instancias de NetScaler CPX en este ejemplo de configuración.

Docker Host 1		Docker Host 2		Docker Host 3	
VIPs	**Services Bound to the VIP**	**VIPs**	**Services Bound to the VIP**	**VIPs**	**Services Bound to the VIP**
VIP-A1 172.17.0.2:30000	SVC-A1 10.102.29.100:80	VIP-A2 172.17.0.3:30000	SVC-A2 10.102.29.105:80	VIP-A3 172.17.0.4:30000	SVC-VIP-A1 10.102.29.100:30000
					SVC-VIP-A2 10.102.29.105:30000
VIP-B1 172.17.0.2:30001	SVC-B1 10.102.29.100:90	VIP-B2 172.17.0.3:30001	SVC-VIP-B1 10.102.29.100:30001	VIP-B3 172.17.0.4:30001	SVC-VIP-B1 10.102.29.100:30001
VIP-C1 172.17.02:30002	SVC-VIP-C2 10.102.29.105:30002	VIP-C2 172.17.0.3:30002	SVC-C2 10.102.29.105:70	VIP-C3 172.17.0.4:30002	SVC-C3 10.102.29.110:70
	SVC-VIP-C3 10.102.29.110:30002				

Para configurar este caso de ejemplo, ejecute el siguiente comando en el símbolo del shell de Linux mientras crea el contenedor NetScaler CPX en los tres hosts de Docker:


```
docker run -dt -p 22 -p 80 -p 161/udp -p 30000-30002: 30000-30002 --
    ulimit core=-1 --privileged=truecpx:6.2
<!--NeedCopy-->
```

Ejecute los siguientes comandos mediante la función Trabajos en NetScaler ADM o mediante las API de NITRO.

En la instancia de NetScaler CPX en Docker Host 1:

```
    add lb vserver VIP-A1 HTTP 172.17.0.2 30000
    add service svc-A1 10.102.29.100 HTTP 80
    bind lb vserver VIP-A1 svc-A1
    add lb vserver VIP-B1 HTTP 172.17.0.2 30001
    add service svc-B1 10.102.29.100 HTTP 90
    bind lb vserver VIP-B1 svc-B1
    add lb vserver VIP-C1 HTTP 172.17.0.2 30002
    add service svc-VIP-C2 10.102.29.105 HTTP 30002
    add service svc-VIP-C3 10.102.29.110 HTTP 30002
    bind lb vserver VIP-C1 svc-VIP-C2
    bind lb vserver VIP-C1 svc-VIP-C3
<!--NeedCopy-->
```

En la instancia de NetScaler CPX en el host 2 de Docker:

```
    add lb vserver VIP-A2 HTTP 172.17.0.3 30000
    add service svc-A2 10.102.29.105 HTTP 80
    bind lb vserver VIP-A2 svc-A2
    add lb vserver VIP-B2 HTTP 172.17.0.3 30001
    add service svc-VIP-B1 10.102.29.100 HTTP 30001
    bind lb vserver VIP-B2 svc-VIP-B1
    add lb vserver VIP-C2 HTTP 172.17.0.3 30002
    add service svc-C2 10.102.29.105 HTTP 70
    bind lb vserver VIP-C2 svc-C2
<!--NeedCopy-->
```

En la instancia de NetScaler CPX en el host 3 de Docker:

```
    add lb vserver VIP-A3 HTTP 172.17.0.4 30000
    add service svc-VIP-A1 10.102.29.100 HTTP 30000
    add service svc-VIP-A2 10.102.29.105 HTTP 30000
    bind lb vserver VIP-A3 svc-VIP-A1
    bind lb vserver VIP-A3 svc-VIP-A2
    add lb vserver VIP-B3 HTTP 172.17.0.4 30001
    add service svc-VIP-B1 10.102.29.100 HTTP 30001
    bind lb vserver VIP-B3 svc-VIP-B1
    add lb vserver VIP-C3 HTTP 172.17.0.4 30002
    add service svc-C3 10.102.29.110 HTTP 70
    bind lb vserver VIP-C3 svc-C3
<!--NeedCopy-->
```


Implementación de NetScaler CPX en una red de host único

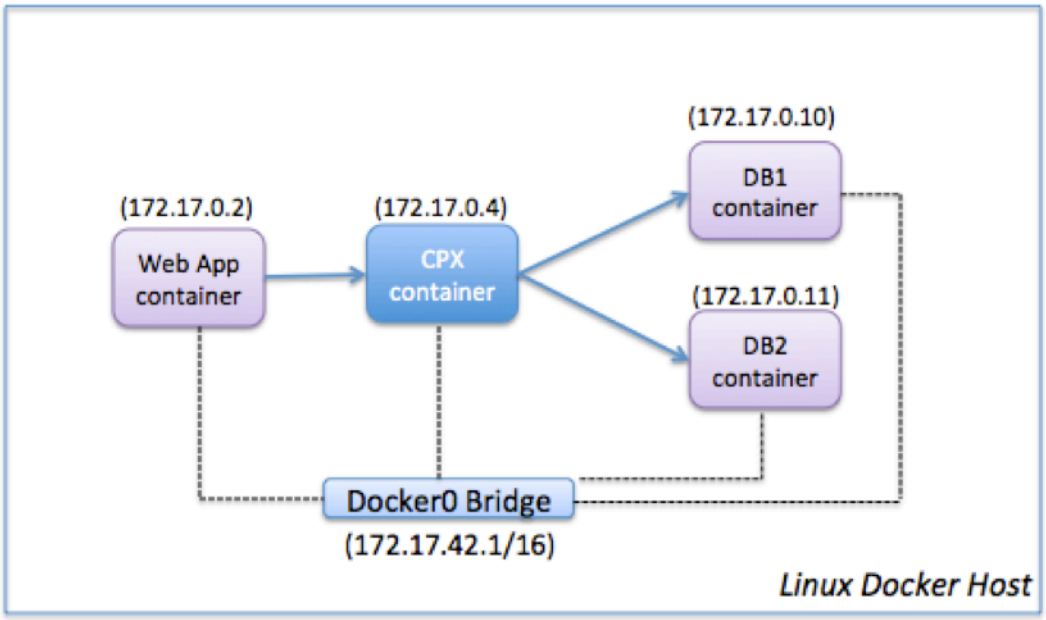
November 23, 2023

En una red de host único, la instancia de NetScaler CPX actúa como proxy entre los contenedores de aplicaciones en el mismo host. En esta capacidad, la instancia de NetScaler CPX proporciona escalabilidad y seguridad a las aplicaciones basadas en contenedores. Además, optimiza el rendimiento y también proporciona información sobre los datos de telemetría.

En una red de host único, el cliente, los servidores y la instancia de NetScaler CPX se implementan como contenedores en el mismo host de Docker. Todos los contenedores están conectados a través del puente docker0.

En este entorno, la instancia de NetScaler CPX actúa como proxy para las aplicaciones aprovisionadas como contenedores en el mismo host de Docker.

Esta ilustración ilustra la topología de host único.



En este ejemplo, un contenedor de aplicaciones web (172.17.0.2) es el cliente y los dos contenedores de base de datos, DB1 (172.17.0.10) y DB2 (172.17.0.11), son los servidores. El contenedor NetScaler CPX (172.17.0.4) se encuentra entre el cliente y los servidores que actúan como proxy.

Para permitir que la aplicación web se comunique con los contenedores de base de datos a través de NetScaler CPX, primero debe configurar dos servicios en el contenedor NetScaler CPX para que representen a los dos servidores. A continuación, configure un servidor virtual mediante la dirección IP de NetScaler CPX y un puerto HTTP no estándar (como 81) porque NetScaler CPX reserva el puerto

HTTP estándar 80 para la comunicación NITRO.

En esta topología, no tiene que configurar ninguna regla de NAT porque el cliente y el servidor están en la misma red.

Para configurar este caso, ejecute los siguientes comandos mediante la función Trabajos en NetScaler ADM o mediante las API de NITRO:

```
1      add service db1 HTTP 172.17.0.10 80
2      add service db2 HTTP 172.17.0.11 80
3      add lb vserver cpx-vip HTTP 172.17.0.4 81
4      bind lb vserver cpx-vip db1
5      bind lb vserver cpx-vip db2
6  <!--NeedCopy-->
```

Implementación de NetScaler CPX en una red de varios hosts

November 23, 2023

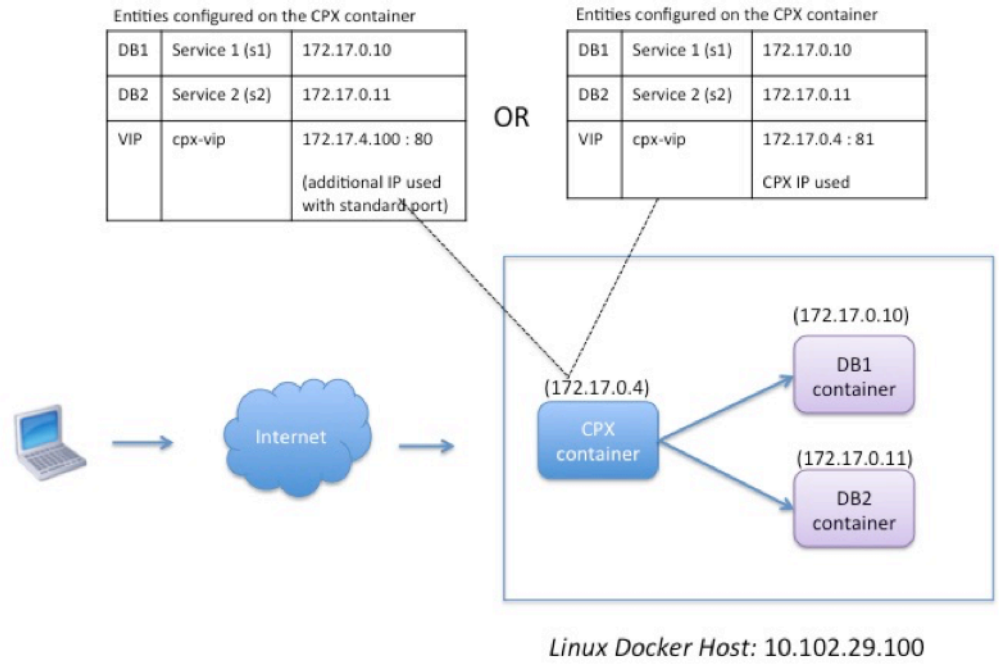
Una instancia de NetScaler CPX en una red multihost se puede configurar en una implementación de producción en el centro de datos, donde proporciona funciones de equilibrio de carga. Además, puede proporcionar funciones de supervisión y datos de análisis.

En una red de varios hosts, las instancias de NetScaler CPX, los servidores backend y los clientes se implementan en hosts diferentes. Puede utilizar topologías de varios hosts en implementaciones de producción en las que la instancia de NetScaler CPX equilibra la carga de un conjunto de aplicaciones y servidores basados en contenedores o incluso servidores físicos.

Topología 1: servidores NetScaler CPX y backend en el mismo host; cliente en una red diferente

En esta topología, la instancia de NetScaler CPX y los servidores de base de datos se aprovisionan en el mismo host de Docker, pero el tráfico del cliente se origina en otro lugar de la red. Esta topología se puede utilizar en una implementación de producción en la que la instancia de NetScaler CPX equilibra la carga de un conjunto de aplicaciones o servidores basados en contenedores.

El siguiente diagrama ilustra esta topología.



En este ejemplo, la instancia de NetScaler CPX (172.17.0.4) y los dos servidores, DB1 (172.17.0.10) y DB2 (172.17.0.11) se aprovisionan en el mismo host de Docker con la dirección IP 10.102.29.100. El cliente reside en otra parte de la red.

Las solicitudes de clientes que se originan en Internet se reciben en el VIP configurado en la instancia de NetScaler CPX, que luego distribuye las solicitudes entre los dos servidores.

Hay dos métodos que puede usar para configurar esta topología:

Método 1: uso de una dirección IP adicional y un puerto estándar para el VIP

1. Configure el VIP en el contenedor NetScaler CPX mediante una dirección IP adicional.
2. Configure una dirección IP adicional para el host de Docker.
3. Configure las reglas NAT para reenviar todo el tráfico recibido en la dirección IP adicional del host Docker a la dirección IP adicional del VIP.
4. Configure los dos servidores como servicios en la instancia de NetScaler CPX.
5. Por último, vincule los servicios al VIP.

Tenga en cuenta que en este ejemplo de configuración, la red 10.x.x.x indica una red pública.

Para configurar este caso de ejemplo, ejecute los siguientes comandos mediante la función Trabajos en NetScaler ADM o mediante las API de NITRO:

```
1     add service s1 172.17.0.10 HTTP 80
```



```
    add service s2 172.17.0.11 HTTP 80
    add lb vserver cpx-vip HTTP 172.17.4.100 80
    bind lb vserver cpx-vip s1
    bind lb vserver cpx-vip s2
<!--NeedCopy-->
```

Configure una dirección IP pública adicional para el host de Docker y una regla NAT ejecutando los siguientes comandos en el símbolo del shell de Linux:

```
    ip addr add 10.102.29.103/24 dev eth0
    iptables -t nat -A PREROUTING -p ip  -d 10.102.29.103 -j DNAT --to-
       destination 172.17.4.100
<!--NeedCopy-->
```

Método 2: utilizar la dirección IP de NetScaler CPX para el VIP y configurar la asignación de puertos:

1. Configure la VIP y los dos servicios en la instancia de NetScaler CPX. Use un puerto no estándar, 81, con el VIP.
2. Vincula los servicios al VIP.
3. Configure una regla NAT para reenviar todo el tráfico recibido en el puerto 50000 del host Docker a la VIP y al puerto 81.

Para configurar este caso de ejemplo, ejecute el siguiente comando en el símbolo del shell de Linux mientras crea el contenedor NetScaler CPX en los tres hosts de Docker:

```
    docker run -dt -p 22 -p 80 -p 161/udp -p 50000:81 --ulimit core=-1
       --privileged=true cpx:6.2

<!--NeedCopy-->
```

Después de aprovisionar la instancia de NetScaler CPX, ejecute los siguientes comandos mediante la función Trabajos en NetScaler ADM o mediante las API de NITRO:

```
    add service s1 172.17.0.10 http 80
    add service s2 172.17.0.11 http 80
    add lb vserver cpx-vip HTTP 172.17.0.4 81
    bind lb vserver cpx-vip s1
    bind lb vserver cpx-vip s2
<!--NeedCopy-->
```

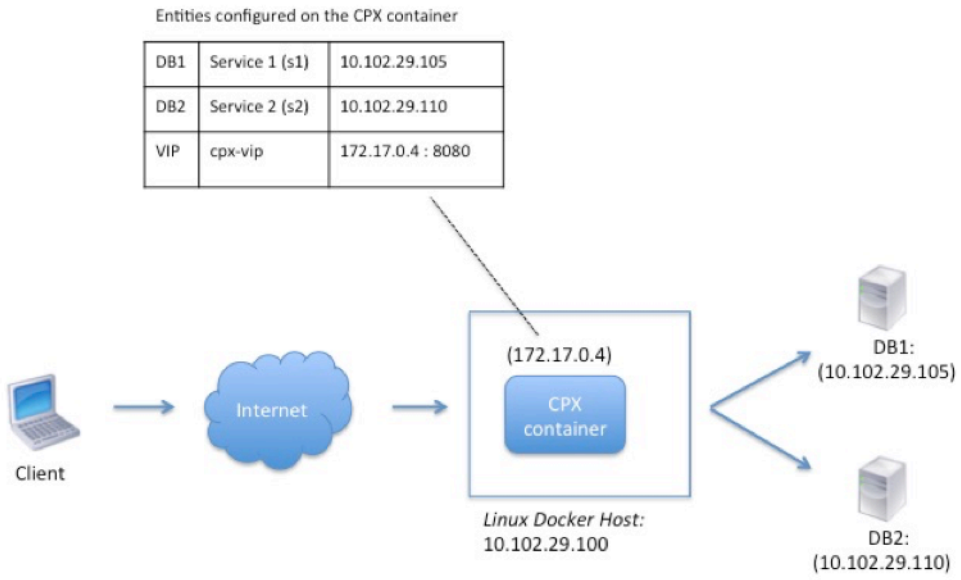
> **Nota:**
>
> Si no ha configurado la asignación de puertos durante el aprovisionamiento de la instancia de NetScaler CPX, configure una regla NAT ejecutando los siguientes comandos en el símbolo del shell de Linux:
>
> **iptables -t nat -A PREROUTING -p tcp -m addrtype –dst-type LOCAL -m tcp –dport** 50000 **-j DNAT –to-destination** 172.17.0.4:81

Topología 2: NetScaler CPX con servidores físicos y cliente

En esta topología, solo la instancia de NetScaler CPX se aprovisiona en un host Docker. El cliente y los servidores no están basados en contenedores y residen en otra parte de la red.

En este entorno, puede configurar la instancia de NetScaler CPX para equilibrar la carga del tráfico en los servidores físicos.

En esta ilustración se ilustra esta topología.



En este ejemplo, el contenedor NetScaler CPX (172.17.0.4) se encuentra entre el cliente y los servidores físicos que actúan como proxy. Los servidores, DB1 (10.102.29.105) y DB2 (10.102.29.110), residen fuera de un host Docker en la red. La solicitud del cliente se origina en Internet y se recibe en NetScaler CPX, que la distribuye en los dos servidores.

Para habilitar esta comunicación entre el cliente y los servidores a través de NetScaler CPX, primero debe configurar la asignación de puertos al crear el contenedor NetScaler CPX. A continuación, configure los dos servicios en el contenedor NetScaler CPX para que representen los dos servidores. Y, por último, configure un servidor virtual mediante la dirección IP de NetScaler CPX y el puerto HTTP asignado no estándar 8080.

Tenga en cuenta que en la configuración de ejemplo, la red 10.x.x.x indica una red pública.

Para configurar este caso de ejemplo, ejecute el siguiente comando en el símbolo del shell de Linux

mientras crea el contenedor NetScaler CPX:

```
    docker run -dt -p 22 -p 80 -p 161/udp -p 8080:8080 --ulimit core=-1
        --privileged=true cpx:6.2
<!--NeedCopy-->
```

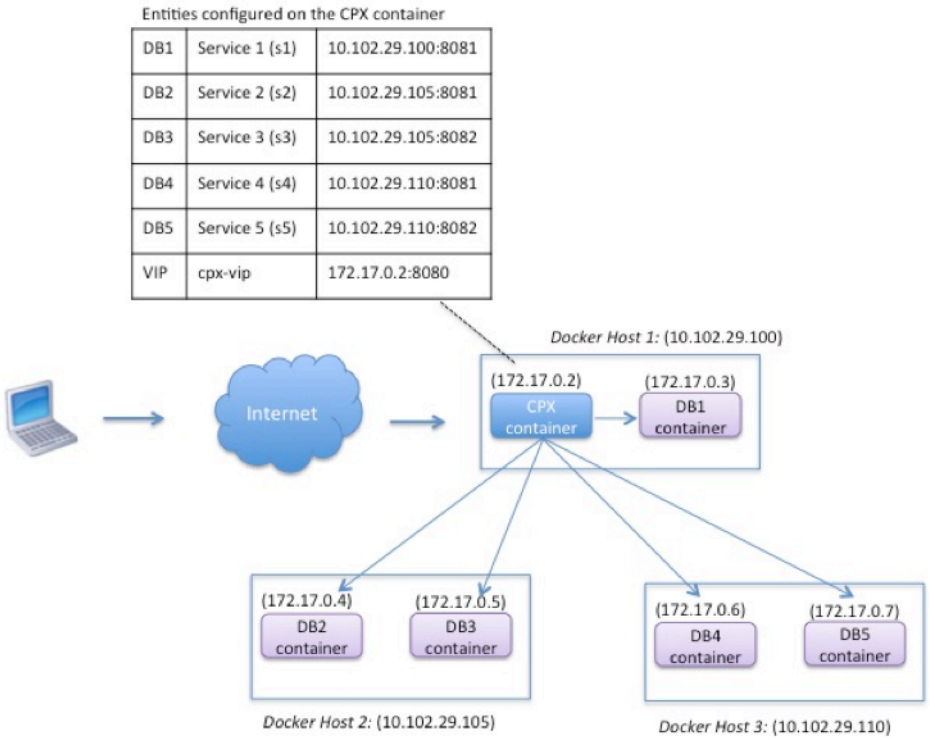
A continuación, ejecute los siguientes comandos mediante la función Trabajos en NetScaler ADM o mediante las API de NITRO:

```
    add service s1 HTTP 10.102.29.105 80
    add service s2 HTTP 10.102.29.110 80
    add lb vserver cpx-vip HTTP 172.17.0.4 8080
    bind lb vserver cpx-vip s1
    bind lb vserver cpx-vip s2
<!--NeedCopy-->
```

Topología 3: NetScaler CPX y servidores aprovisionados en diferentes hosts

En esta topología, la instancia de NetScaler CPX y los servidores de base de datos se aprovisionan en diferentes hosts de Docker y el tráfico del cliente se origina en Internet. Esta topología se puede utilizar en una implementación de producción en la que la instancia de NetScaler CPX equilibra la carga de un conjunto de aplicaciones o servidores basados en contenedores.

El siguiente diagrama ilustra esta topología.



En este ejemplo, la instancia de NetScaler CPX y un servidor (DB1) se aprovisionan en el mismo host de Docker con la dirección IP 10.102.29.100. Otros cuatro servidores (DB2, DB3, DB4 y DB5) se aprovisionan en dos hosts Docker diferentes, 10.102.29.105 y 10.102.29.110.

Las solicitudes de clientes que se originan en Internet se reciben en la instancia de NetScaler CPX, que luego distribuye las solicitudes en los cinco servidores. Para habilitar esta comunicación, debe configurar lo siguiente:

1. Configure la asignación de puertos al crear su contenedor NetScaler CPX. En este ejemplo, esto significa que debe reenviar el puerto 8080 del contenedor al puerto 8080 del host. Cuando la solicitud del cliente llega al puerto 8080 del host, se asigna al puerto 8080 del contenedor CPX.
2. Configure los cinco servidores como servicios en la instancia de NetScaler CPX. Debe usar una combinación de la dirección IP del host Docker respectivo y el puerto asignado para configurar estos servicios.
3. Configure una VIP en la instancia de NetScaler CPX para recibir la solicitud del cliente. Este VIP debe estar representado por la dirección IP de NetScaler CPX y el puerto 8080 que se asignó al puerto 8080 del host.
4. Por último, vincule los servicios al VIP.

Tenga en cuenta que en la configuración de ejemplo, la red 10.x.x.x indica una red pública.

Para configurar este caso de ejemplo, ejecute el siguiente comando en el símbolo del shell de Linux mientras crea el contenedor NetScaler CPX:

```
1     docker run -dt -p 22 -p 80 -p 161/udp -p 8080:8080 --ulimit core=-1
          --privileged=true cpx:6.2
2 <!--NeedCopy-->
```

Ejecute los siguientes comandos mediante la función Trabajos en NetScaler ADM o mediante las API de NITRO:

```
1     add service s1 10.102.29.100 HTTP 8081
2     add service s2 10.102.29.105 HTTP 8081
3     add service s3 10.102.29.105 HTTP 8082
4     add service s4 10.102.29.110 HTTP 8081
5     add service s5 10.102.29.110 HTTP 8082
6     add lb vserver cpx-vip HTTP 172.17.0.2 8080
7     bind lb vserver cpx-vip s1
8     bind lb vserver cpx-vip s2
9     bind lb vserver cpx-vip s3
10    bind lb vserver cpx-vip s4
11    bind lb vserver cpx-vip s5
12 <!--NeedCopy-->
```


Implementar NetScaler CPX con acceso directo a la red

November 23, 2023

En el modo de red puente, puede configurar la instancia de NetScaler CPX para que tenga acceso directo a la red. En este caso, el tráfico entrante se recibe directamente en la IP (VIP) del servidor virtual NetScaler CPX.

Para habilitar esta comunicación, primero debe configurar una dirección IP pública en el puente docker0. A continuación, elimine la dirección IP pública del puerto de red eth0 y vincule el puerto de red al puente docker0.

Configure el equilibrio de carga agregando los dos servicios y, a continuación, configure una dirección IP pública de red como VIP en la instancia de NetScaler CPX. Las solicitudes de los clientes se reciben directamente en el VIP.

En la configuración de ejemplo, la red 10.x.x.x indica una red pública.

Para configurar este caso, ejecute el siguiente comando en el símbolo del shell de Linux:

```
1     ip addr add 10.102.29.100/24 dev docker0;
2     ip addr del 10.102.29.100/24 dev eth0;
3     brctl addif docker0 eth0;
4     ip route del default;
5     ip route add default via 10.102.29.1 dev docker0
6 <!--NeedCopy-->
```

Ya sea mediante la función Trabajos en NetScaler ADM o mediante las API de NITRO, ejecute los siguientes comandos:

```
1     add service s1 172.17.0.8 http 80
2     add service s2 172.17.0.9 http 80
3     add lb vserver cpx-vip HTTP 10.102.29.102 80
4     bind lb vserver cpx-vip s1
5     bind lb vserver cpx-vip s2
6 <!--NeedCopy-->
```

Configurar NetScaler CPX en Kubernetes mediante ConfigMaps

November 23, 2023

En Kubernetes, puede configurar la instancia de NetScaler CPX mediante ConfigMaps. Con ConfigMaps, puede configurar dinámicamente la instancia de NetScaler CPX durante el inicio de la instancia.

Cree un archivo de configuración `cpx.conf` que incluya la configuración específica de NetScaler y los comandos bash shell que quiera ejecutar dinámicamente en la instancia de NetScaler CPX. La estructura del archivo de configuración requiere dos tipos de etiquetas, `#NetScaler Commands` y `#Shell Commands`. Debajo de la etiqueta `#NetScaler Commands`, debe agregar todos los comandos de NetScaler para configurar la configuración específica de NetScaler en la instancia de NetScaler CPX. Debajo de la etiqueta `#Shell Commands`, debe agregar los comandos de shell que quiere ejecutar en la instancia de NetScaler CPX.

> **Importante:**
>
> - Las etiquetas se pueden repetir varias veces en el archivo de configuración.
> - El archivo de configuración también puede incluir comentarios. Agregue un carácter "**#**" antes de los comentarios.
> - Las etiquetas no distinguen entre mayúsculas y minúsculas.
> - Si hay casos de error al implementar el contenedor NetScaler CPX con el archivo de configuración, los errores se registran en el archivo `ns.log`.
> - Una vez iniciada la instancia de NetScaler CPX, si cambia el ConfigMap, la configuración actualizada se aplica solo cuando se reinicia la instancia de NetScaler CPX.

A continuación se muestra un archivo de configuración de ejemplo:

```
#NetScaler Commands
add lb vserver v1 http 1.1.1.1 80
add service s1 2.2.2.2 http 80
bind lb vserver v1 s1
#Shell Commands
touch /etc/a.txt
echo "this is a" > /etc/a.txt
#NetScaler Commands
add lb vserver v2 http
#Shell Commands
echo "this is a 1" >> /etc/a.txt
#NetScaler Commands
add lb vserver v3 http
<!--NeedCopy-->
```

Una vez que haya creado el archivo de configuración, debe crear un ConfigMap a partir del archivo de configuración mediante el comando `kubectl create configmap`.

```
kubectl create configmap cpx-config --from-file=cpx.conf
<!--NeedCopy-->
```

En el ejemplo anterior, puede crear un ConfigMap, `cpx-config` basado en el archivo de configuración cpx.conf. A continuación, puede usar este ConfigMap en el archivo YAML que se utiliza para implementar la instancia de NetScaler CPX.

Puede ver el ConfigMap creado mediante el comando `kubectl get configmap`.


```
root@node1:~/yaml# kubectl get configmap cpx-config -o yaml
```

Muestra:

```
    apiVersion: v1
    data:
        cpx.conf: |
        #NetScaler Commands
            add lb vserver v1 http 1.1.1.1 80
            add service s1 2.2.2.2 http 80
            bind lb vserver v1 s1
        #Shell Commands
            touch /etc/a.txt
            echo "this is a" > /etc/a.txt
            echo "this is the file" >> /etc/a.txt
            ls >> /etc/a.txt
        #NetScaler Commands
            add lb vserver v2 http
        #Shell Commands
            echo "this is a 1" >> /etc/a.txt
        #NetScaler Commands
            add lb vserver v3 http
        #end of file
    kind: ConfigMap
    metadata:
        creationTimestamp: 2017-12-26T06:26:50Z
        name: cpx-config
        namespace: default
        resourceVersion: "8865149"
        selfLink: /api/v1/namespaces/default/configmaps/cpx-config
        uid: c1c7cb5b-ea05-11e7-914a-926745c10b02
<!--NeedCopy-->
```

Puede especificar el ConfigMap creado, `cpx-config`, en el archivo YAML utilizado para implementar la instancia de NetScaler CPX de la siguiente manera:

```
apiVersion: v1
kind: Pod
metadata:
  name: cpx-1
  labels:
    app: cpx-daemon
  annotations:
    NETSCALER_AS_APP: "True"
spec:
  hostNetwork: true
  containers:
  - name: cpx
    image: "quay.io/citrix/citrix-k8s-cpx-ingress:13.1-33.47"
    securityContext:
      privileged: true
    volumeMounts:
    - name: config-volume
```



```
        mountPath: /cpx/bootup_conf
      env:
        - name: "EULA"
          value: "yes"
        - name: "NS_NETMODE"
          value: "HOST"
        - name: "kubernetes_url"
          value: "https://10.90.248.101:6443"
        - name: "NS_MGMT_SERVER"
          value: "10.90.248.99"
        - name: "NS_MGMT_FINGER_PRINT"
          value: "19:71:A3:36:85:0A:2B:62:24:65:0F:7E:72:CC:DC:AD:B8:BF
              :53:1E"
        - name: "NS_ROUTABLE"
          value: "FALSE"
        - name: "KUBERNETES_TASK_ID"
          valueFrom:
            fieldRef:
              fieldPath: metadata.name
      imagePullPolicy: Never
    volumes:
    - name: config-volume
      configMap:
        name: cpx-config
<!--NeedCopy-->
```

Una vez que se implementa la instancia de NetScaler CPX is e inicia la configuración especificada en ConfigMap, `cpx-config` se aplica a la instancia de NetScaler CPX.

Implementar de NetScaler CPX como cachés DNS locales para nodos de Kubernetes

November 23, 2023

Los pods de aplicaciones de un clúster de Kubernetes dependen del DNS para comunicarse con otros pods de aplicaciones. Las solicitudes de DNS de aplicaciones dentro de un clúster de Kubernetes se gestionan mediante DNS de Kubernetes (kube-dns). Debido a la adopción más amplia de arquitecturas de microservicios, las tasas de solicitudes de DNS dentro de un clúster de Kubernetes están aumentando. Como resultado, el DNS de Kubernetes (kube-dns) está sobrecargado. Ahora puede implementar NetScaler CPX como una memoria caché de DNS local en cada nodo de Kubernetes y reenviar solicitudes de DNS desde los pods de aplicaciones en el nodo a NetScaler CPX. Por lo tanto, puede resolver las solicitudes de DNS más rápido y reducir significativamente la carga en el DNS de Kubernetes.

Para implementar NetScaler CPX, se utiliza una entidad de Kubernetes DaemonSet para programar

pods de NetScaler CPX en cada nodo del clúster de Kubernetes. Un DaemonSet de Kubernetes garantiza que haya una instancia de NetScaler CPX en cada nodo de Kubernetes del clúster.
Para que los pods de aplicaciones dirijan el tráfico a los pods DNS de CPX, debe crear un servicio de Kubernetes con puntos finales como pods de NetScaler CPX. La IP de clúster de este servicio se usa como punto de enlace DNS para los pods de la aplicación. Para asegurarse de que los pods de la aplicación usen la dirección IP del clúster de servicios NetScaler CPX para la resolución de DNS, debe actualizar el archivo de configuración de kubelet en cada nodo con la IP del clúster de servicios NetScaler CPX.

Se introducen las siguientes variables de entorno para admitir la implementación de NetScaler CPX como caché DNS de NodeLocal:

- `KUBE_DNS_SVC_IP`: Especifica la dirección IP del clúster del servicio `kube-dns`, que es un argumento obligatorio para desencadenar la configuración en un pod de NetScaler CPX. El pod de NetScaler CPX dirige las consultas DNS a esta dirección IP cuando la respuesta a la consulta DNS no está disponible en la memoria caché de NetScaler CPX.
- `CPX_DNS_SVC_IP`: Especifica la dirección IP del clúster del servicio NetScaler CPX. La variable de entorno `CPX_DNS_SVC_IP` se usa para configurar el DNS local en los nodos. Al configurar esta variable, se agrega una regla `iptables` para dirigir las solicitudes DNS que se originan en los pods de aplicaciones al pod CPX local de NetScaler dentro del nodo.
- `NS_DNS_FORCE_TCP`: Esta variable de entorno fuerza el uso de TCP para las solicitudes de DNS, incluso si las consultas se reciben a través de UDP.
- `NS_DNS_EXT_RESLV_IP`: Especifica la dirección IP del servidor de nombres externo para dirigir las solicitudes de DNS para un dominio específico.
- `NS_DNS_MATCH_DOMAIN`: Especifica la cadena de dominio externo que se va a comparar para dirigir las consultas al servidor de nombres externo.

Implementar de NetScaler CPX como cachés de DNS en nodos

La implementación de NetScaler CPX como caché de DNS local para un clúster de Kubernetes incluye las siguientes tareas:

En el nodo maestro:

- Crear un servicio de Kubernetes con puntos finales como pods de NetScaler CPX
- Crear un ConfigMap para definir variables de entorno para pods de NetScaler CPX
- Programe pods de NetScaler CPX en cada nodo del clúster de Kubernetes mediante un DaemonSet de Kubernetes.

En los nodos trabajadores:

- Modifique el archivo de configuración de kubelet con la dirección IP del clúster del servicio NetScaler CPX para reenviar las solicitudes DNS a NetScaler CPX.

Configuración en el nodo maestro de Kubernetes

Realice los siguientes pasos en el nodo principal de Kubernetes para implementar NetScaler CPX como la memoria caché DNS local para los nodos:

1. Cree un servicio con pods de NetScaler CPX como puntos finales mediante el archivo `cpx_dns_svc.yaml`.

```
kubectl apply -f cpx_dns_svc.yaml
```

   El archivo `cpx_dns_svc.yaml` se proporciona de la siguiente manera:

```
        apiVersion: v1
        kind: Service
        metadata:
          name: cpx-dns-svc
          labels:
            app: cpxd
        spec:
          ports:
          - protocol: UDP
            port: 53
            name: dns
          - protocol: TCP
            port: 53
            name: dns-tcp
          selector:
            app: cpx-daemon
```

2. Obtenga la dirección IP del servicio NetScaler CPX.

```
kubectl get svc cpx-dns-svc
```

3. Obtenga la dirección IP del servicio DNS de Kube.

```
kubectl get svc -n kube-system
```

4. Cree un ConfigMap para definir variables de entorno para pods de NetScaler CPX. Estas variables de entorno se utilizan para pasar las direcciones IP del servicio NetScaler CPX y el servicio DNS de Kube. En este paso, `cpx-dns-cache` se crea un ejemplo de ConfigMap utilizando las variables de entorno especificadas como datos (pares clave-valor) en un archivo.

```
kubectl create configmap cpx-dns-cache --from-file <path-to-file>
```

   A continuación se muestra un archivo de ejemplo con las variables de entorno como pares clave-valor.


```
  CPX_DNS_SVC_IP: 10.111.95.145
  EULA: "yes"
  KUBE_DNS_SVC_IP: 10.96.0.10
  NS_CPX_LITE: "1"
  NS_DNS_EXT_RESLV_IP: 10.102.217.142
  NS_DNS_MATCH_DOMAIN: citrix.com
  PLATFORM: CP1000
```

A continuación se muestra un ejemplo de ConfigMap:

```
apiVersion: v1
data:
  CPX_DNS_SVC_IP: 10.111.95.145
  EULA: "yes"
  KUBE_DNS_SVC_IP: 10.96.0.10
  NS_CPX_LITE: "1"
  NS_DNS_EXT_RESLV_IP: 10.102.217.142
  NS_DNS_MATCH_DOMAIN: citrix.com
  PLATFORM: CP1000
kind: ConfigMap
metadata:
  creationTimestamp: "2019-10-15T07:45:54Z"
  name: cpx-dns-cache
  namespace: default
  resourceVersion: "8026537"
  selfLink: /api/v1/namespaces/default/configmaps/cpx-dns-cache
  uid: 8d06f6ee-133b-4e1a-913c-9963cbf4f48
```

5. Cree un DaemonSet de Kubernetes para NetScaler CPX en el nodo principal.

```
kubectl apply -f cpx_daemonset.yaml
```

El archivo `cpx_daemonset.yaml` se proporciona de la siguiente manera:

```
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: cpx-daemon
  labels:
        app: cpxd
spec:
   selector:
        matchLabels:
          app: cpx-daemon
template:
   metadata:
        labels:
          app: cpx-daemon
   spec:
       containers:
       - name: cpxd
        imagePullPolicy: IfNotPresent
        image: localhost:5000/dev/cpx
```



```
20          volumeMounts:
21           - mountPath: /netns/default/
22          name: test-vol
23          ports:
24          - containerPort: 53
25      envFrom:
26          - configMapRef:
27          name: cpx-dns-cache
28      securityContext:
29      privileged: true
30      allowPrivilegeEscalation: true
31      capabilities:
32       add: ["NET_ADMIN"]
33      volumes:
34         -  name: test-vol
35         hostPath:
36         path: /proc/1/ns
37         type: Directory
```

Configuración en nodos de trabajo en el clúster de Kubernetes

Una vez que haya completado la configuración en el nodo maestro, lleve a cabo el siguiente paso en los nodos de trabajo:

1. Modifique el archivo de configuración de kubelet para que los pods de aplicaciones puedan usar la IP del clúster de servicios de NetScaler CPX para la resolución de DNS mediante uno de los siguientes pasos:

   - Siga los pasos en reconfigurar el kubelet de un nodo y modifique el valor del argumento `--cluster-dns` en el siguiente formato.

     ```
     1       --cluster-dns=<CPX_DNS_SVC_IP>,<KUBE_DNS_SVC_IP>
     ```

     O bien:

   - Modifique el archivo `/etc/systemd/system/kubelet.service.d/10-kubeadm.conf` y modifique el argumento `--cluster-dns` mediante los siguientes pasos.

     a) Modifique la configuración de kubelet y especifique la dirección IP del clúster del servicio y la dirección IP `kube-dns` del servicio NetScaler CPX para el argumento `--cluster-dns`.

     ```
     1   root@node:~# cat /etc/systemd/system/kubelet.service.d/10-
            kubeadm.conf | grep KUBELET\_DNS\_ARGS
     2
     3   Environment="KUBELET_DNS_ARGS=--cluster-dns
            =10.111.95.145,10.96.0.10 --cluster-domain=cluster.
            local"
     ```



```
4   ExecStart=/usr/bin/kubelet $KUBELET_KUBECONFIG_ARGS
       $KUBELET_CONFIG_ARGS $KUBELET_DNS_ARGS
```

b) Vuelva a cargar el kubelet de los nodos con los siguientes comandos:

```
1   # systemctl daemon-reload
2   # service kubelet restart
```

Implementar el proxy de NetScaler CPX en Google Compute Engine

March 21, 2024

En esta guía de implementación, se describe cómo puede implementar NetScaler CPX con Docker en Google Compute Engine (GCE) de Google Cloud con NetScaler ADM ejecutándose dentro de la red empresarial. En esta implementación, NetScaler CPX instalado en GCE equilibra la carga de dos servidores back-end, y NetScaler ADM proporciona soluciones de licencias y análisis.

NetScaler CPX es un proxy basado en contenedores que admite la funcionalidad completa de capa 7, la descarga de SSL, varios protocolos y la API de NITRO. NetScaler ADM proporciona soluciones de administración, licencias y análisis. Como servidor de licencias, NetScaler ADM otorga derechos a instancias de NetScaler CPX que se ejecutan en las instalaciones o en la nube.

CPX y CPX Express son las mismas imágenes. Cuando licencia e instala la imagen CPX con NetScaler ADM, la imagen CPX en Docker App Store (versión 11 o 12) se convierte en una instancia CPX completa. Sin licencia, la imagen CPX se convierte en una instancia CPX Express que admite conexiones SSL de 20 Mbps y 250.

Requisitos previos

- 2 GB de memoria y 1 vCPU dedicada a NetScaler CPX
- Código abierto de Docker disponible en GCE
- NetScaler ADM se ejecuta en las instalaciones con conexión a Internet o VPN a GCE

> **Nota**
>
> Para obtener información sobre cómo implementar NetScaler ADM, consulte Implementación de NetScaler ADM.

Pasos de configuración

Debe realizar los siguientes pasos para configurar esta implementación.

1. Instala Docker en una VM de GCE.
2. Configure la comunicación de API remota con la instancia de Docker.
3. Instale la imagen de NetScaler CPX.
4. Cree una instancia CPX.
5. Licencia de NetScaler CPX a través de NetScaler ADM.
6. Configure los servicios de equilibrio de carga en NetScaler CPX y verifique la configuración.
    a) Instale los servidores web NGINX.
    b) Configure NetScaler CPX para el equilibrio de carga y verifique la distribución de la carga a ambos servicios web.

Paso 1: Instalar Docker en una VM de GCE

Desde GCE, crea una VM Linux Ubuntu. A continuación, instale Docker en la VM mediante los comandos que se muestran en el siguiente ejemplo:

```
$ sudo curl –ssl https://get.docker.com/ | sh
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
0 0 0 0 0 0 0 0 --:--:-- --:--:-- --:--:-- 0curl: (6) Could not resolve
    host: xn--ssl-1n0a
100 17409 100 17409 0 0 21510 0 --:--:-- --:--:-- --:--:-- 21492
apparmor is enabled in the kernel and apparmor utils were already
   installed
+ sudo -E sh -c apt-key add -
+ echo -----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1

mQINBFWln24BEADrBl5p99uKh8+rpvqJ48u4eTtjeXAWbslJotmC/CakbNSqOb9o
ddfzRvGVeJVERt/Q/mlvEqgnyTQy+e6oEYN2Y2kqXceUhXagThnqCoxcEJ3+KM4R
mYdoe/BJ/J/6rHOjq7Omk24z2qB3RU1uAv57iY5VGw5p45uZB4C4pNNsBJXoCvPn
TGAs/7IrekFZDDgVraPx/hdiwopQ8NltSfZCyu/jPpWFK28TR8yfVlzYFwibj5WK
dHM7ZTqlA1tHIG+agyPf3Rae0jPMsHR6q+arXVwMccyOi+ULU0z8mHUJ3iEMIrpT
X+80KaN/ZjibfsBOCjcfiJSB/acn4nxQQgNZigna32velafhQivsNREFeJpzENiG
HOoyC6qVeOgKrRiKxzymj0FIMLru/iFF5pSWcBQB7PYlt8J0G80lAcPr6VCiN+4c
NKv03SdvA69dCOj79PuO9IIvQsJXsSq96HB+TeEmmL+xSdpGtGdCJHHM1fDeCqkZ
hT+RtBGQL2SEdWjxbF43oQopocT8cHvyX6Zaltn0svoGs+wX3Z/H6/8P5anog43U
65c0A+64Jj00rNDr8j31izhtQMRo892kGeQAaaxg4Pz6HnS7hRC+cOMHUU4HA7iM
zHrouAdYeTZeZEQOA7SxtCME9ZnGwe2grxPXh/U/80WJGkzLFNcTKdv+rwARAQAB
tDdEb2NrZXIgUmVsZWFzZSBUb29sIChyZWxlYXNlZG9ja2VyKSA8ZG9ja2VyQGRv
Y2tlci5jb20+iQIcBBABCgAGBQJWw7vdAAoJEFyzYeVS+w0QHysP/i37m4SyoOCV
cnybl18vzwBEcp4VCRbXvHvOXty1gccVIV8/aJqNKgBV97lY3vrpOyiIeB8ETQeg
srxFE7t/Gz0rsLObqfLEHdmn5iBJRkhLfCpzjeOnyB3Z0IJB6UogO/msQVYe5CXJ
l6uwr0AmoiCBLrVlDAktxVh9RWch0l0KZRX2FpHu8h+uM0/zySqIidlYfLa3y5oH
scU+nGU1i6ImwDTD3ysZC5jp9aVfvUmcESyAb4vvdcAHR+bXhA/RW8QHeeMFliWw
7Z2jYHyuHmDnWG2yUrnCqAJTrWV+OfKRIzzJFBs4e88ru5h2ZIXdRepw/+COYj34
```



```
29 LyzxR2cxr2u/xvxwXCkSMe7F4KZAphD+1ws61FhnUMi/PERMYfTFuvPrCkq4gyBj
30 t3fFpZ2NR/fKW87QOeVcn1ivXl9id3MMs9KXJsg7QasT7mCsee2VIFsxrkFQ2jNp
31 D+JAERRn9Fj4ArHL5TbwkkFbZZvSi6fr5h2GbCAXIGhIXKnjjorPY/YDX6X8AaHO
32 W1zblWy/CFr6VFl963jrjJgag0G6tNtBZLrclZgWhOQpeZZ5Lbvz2ZA5CqRrfAVc
33 wPNW1fObFIRtqV6vuVluFOPCMAAnOnqR02w9t17iVQjO3oVN0mbQi9vjuExXh1Yo
34 ScVetiO6LSmlQfVEVRTqHLMgXyR/EMo7iQIcBBABCgAGBQJXSWBlAAoJEFyzYeVS
35 +w0QeH0QAI6btAfYwYPuAjfRUy9qlnPhZ+xt1rnwsUzsbmo8K3XTNh+l/R08nu0d
36 sczw30Q1wju28fh1N8ay223+69f0+yICaXqR18AbGgFGKX7vo0gfEVaxdItUN3eH
37 NydGFzmeOKbAlrxIMECnSTG/TkFVYO9Ntlv9vSN2BupmTagTRErxLZKnVsWRzp+X
38
39 -----END PGP PUBLIC KEY BLOCK-----
40
41 OK
42 + sudo -E sh -c mkdir -p /etc/apt/sources.list.d
43 + dpkg --print-architecture
44 + sudo -E sh -c echo deb \\[arch=amd64\\] https://apt.dockerproject.org
      /repo ubuntu-yakkety main > /etc/apt/sources.list.d/docker.list
45 + sudo -E sh -c sleep 3; apt-get update; apt-get install -y -q docker-
      engine
46 Hit:1 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety InRelease
47 Get:2 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety-updates
      InRelease [102 kB]
48 Get:3 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety-backports
      InRelease [102 kB]
49 Get:4 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety/restricted
      Sources [5,376 B]
50 Get:5 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety/multiverse
      Sources [181 kB]
51 Get:6 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety/universe
      Sources [8,044 kB]
52 Get:7 http://archive.canonical.com/ubuntu yakkety InRelease [11.5 kB]
53 Get:8 http://security.ubuntu.com/ubuntu yakkety-security InRelease [102
       kB]
54 Get:9 https://apt.dockerproject.org/repo ubuntu-yakkety InRelease [47.3
       kB]
55 Get:10 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety/main
      Sources [903 kB]
56 Get:11 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety-updates/
      restricted Sources [2,688 B]
57 Get:12 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety-updates/
      universe Sources [57.9 kB]
58 Get:13 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety-updates/
      multiverse Sources [3,172 B]
59 Get:14 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety-updates/
      main Sources [107 kB]
60 Get:15 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety-updates/
      main amd64 Packages [268 kB]
61 Get:16 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety-updates/
      main Translation-en [122 kB]
62 Get:17 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety-updates/
      universe amd64 Packages [164 kB]
63 Get:18 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety-updates/
      universe Translation-en [92.4 kB]
```



```
64 Get:19 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety-updates/
      multiverse amd64 Packages [4,840 B]
65 Get:20 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety-updates/
      multiverse Translation-en [2,708 B]
66 Get:21 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety-backports/
      universe Sources [2,468 B]
67 Get:22 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety-backports/
      main Sources [2,480 B]
68 Get:23 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety-backports/
      main amd64 Packages [3,500 B]
69 Get:24 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety-backports/
      universe amd64 Packages [3,820 B]
70 Get:25 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety-backports/
      universe Translation-en [1,592 B]
71 Get:26 http://archive.canonical.com/ubuntu yakkety/partner amd64
      Packages [2,480 B]
72 Get:27 http://security.ubuntu.com/ubuntu yakkety-security/main Sources
      [47.7 kB]
73 Get:28 https://apt.dockerproject.org/repo ubuntu-yakkety/main amd64
      Packages [2,453 B]
74 Get:29 http://security.ubuntu.com/ubuntu yakkety-security/universe
      Sources [20.7 kB]
75 Get:30 http://security.ubuntu.com/ubuntu yakkety-security/multiverse
      Sources [1,140 B]
76 Get:31 http://security.ubuntu.com/ubuntu yakkety-security/restricted
      Sources [2,292 B]
77 Get:32 http://security.ubuntu.com/ubuntu yakkety-security/main amd64
      Packages [150 kB]
78 Get:33 http://security.ubuntu.com/ubuntu yakkety-security/main
      Translation-en [68.0 kB]
79 Get:34 http://security.ubuntu.com/ubuntu yakkety-security/universe
      amd64 Packages [77.2 kB]
80 Get:35 http://security.ubuntu.com/ubuntu yakkety-security/universe
      Translation-en [47.3 kB]
81 Get:36 http://security.ubuntu.com/ubuntu yakkety-security/multiverse
      amd64 Packages [2,832 B]
82 Fetched 10.8 MB in 2s (4,206 kB/s)
83 Reading package lists... Done
84 Reading package lists...
85 Building dependency tree...
86 Reading state information...
87 The following additional packages will be installed:
88 aufs-tools cgroupfs-mount libltdl7
89 The following NEW packages will be installed:
90 aufs-tools cgroupfs-mount docker-engine libltdl7
91 0 upgraded, 4 newly installed, 0 to remove and 37 not upgraded.
92 Need to get 21.2 MB of archives.
93 After this operation, 111 MB of additional disk space will be used.
94 Get:1 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety/universe
      amd64 aufs-tools amd64 1:3.2+20130722-1.1ubuntu1 [92.9 kB]
95 Get:2 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety/universe
      amd64 cgroupfs-mount all 1.3 [5,778 B]
96 Get:3 http://us-west1.gce.archive.ubuntu.com/ubuntu yakkety/main amd64
```



```
    libltdl7 amd64 2.4.6-1 [38.6 kB]
97  Get:4 https://apt.dockerproject.org/repo ubuntu-yakkety/main amd64
    docker-engine amd64 17.05.0~ce-0~ubuntu-yakkety [21.1 MB]
98  Fetched 21.2 MB in 1s (19.8 MB/s)
99  Selecting previously unselected package aufs-tools.
100 (Reading database ... 63593 files and directories currently installed.)
101 Preparing to unpack .../aufs-tools_1%3a3.2+20130722-1.1ubuntu1_amd64.
    deb ...
102 Unpacking aufs-tools (1:3.2+20130722-1.1ubuntu1) ...
103 Selecting previously unselected package cgroupfs-mount.
104 Preparing to unpack .../cgroupfs-mount_1.3_all.deb ...
105 Unpacking cgroupfs-mount (1.3) ...
106 Selecting previously unselected package libltdl7:amd64.
107 Preparing to unpack .../libltdl7_2.4.6-1_amd64.deb ...
108 Unpacking libltdl7:amd64 (2.4.6-1) ...
109 Selecting previously unselected package docker-engine.
110 Preparing to unpack .../docker-engine_17.05.0~ce-0~ubuntu-yakkety_amd64
    .deb ...
111 Unpacking docker-engine (17.05.0~ce-0~ubuntu-yakkety) ...
112 Setting up aufs-tools (1:3.2+20130722-1.1ubuntu1) ...
113 Processing triggers for ureadahead (0.100.0-19) ...
114 Setting up cgroupfs-mount (1.3) ...
115 Processing triggers for libc-bin (2.24-3ubuntu2) ...
116 Processing triggers for systemd (231-9ubuntu4) ...
117 Setting up libltdl7:amd64 (2.4.6-1) ...
118 Processing triggers for man-db (2.7.5-1) ...
119 Setting up docker-engine (17.05.0~ce-0~ubuntu-yakkety) ...
120 Created symlink /etc/systemd/system/multi-user.target.wants/docker.
    service → /lib/systemd/system/docker.service.
121 Created symlink /etc/systemd/system/sockets.target.wants/docker.socket
    → /lib/systemd/system/docker.socket.
122 Processing triggers for ureadahead (0.100.0-19) ...
123 Processing triggers for libc-bin (2.24-3ubuntu2) ...
124 Processing triggers for systemd (231-9ubuntu4) ...
125 + sudo -E sh -c docker version
126 Client:
127 Version: 17.05.0-ce
128 API version: 1.29
129 Go version: go1.7.5
130 Git commit: 89658be
131 Built: Thu May 4 22:15:36 2017
132 OS/Arch: linux/amd64
133
134 Server:
135 Version: 17.05.0-ce
136 API version: 1.29 (minimum version 1.12)
137 Go version: go1.7.5
138 Git commit: 89658be
139 Built: Thu May 4 22:15:36 2017
140 OS/Arch: linux/amd64
141 Experimental: false
142
143 If you would like to use Docker as a non-root user, you should now
```



```
    consider
adding your user to the "docker" group with something like:

sudo usermod -aG docker albert_lee

Remember that you will have to log out and back in for this to take
    effect.

WARNING: Adding a user to the "docker" group will grant the ability to
    run
containers which can be used to obtain root privileges on the
docker host.
Refer to https://docs.docker.com/engine/security/security/#docker-
    daemon-attack-surface
for more information.

$

\*\*$ sudo docker info\*\*
Containers: 0
Running: 0
Paused: 0
Stopped: 0
Images: 0
Server Version: 17.05.0-ce
Storage Driver: aufs
Root Dir: /var/lib/docker/aufs
Backing Filesystem: extfs
Dirs: 0
Dirperm1 Supported: true
Logging Driver: json-file
Cgroup Driver: cgroupfs
Plugins:
Volume: local
Network: bridge host macvlan null overlay
Swarm: inactive
Runtimes: runc
Default Runtime: runc
Init Binary: docker-init
containerd version: 9048e5e50717ea4497b757314bad98ea3763c145
runc version: 9c2d8d184e5da67c95d601382adf14862e4f2228
init version: 949e6fa
Security Options:
apparmor
seccomp
Profile: default
Kernel Version: 4.8.0-51-generic
Operating System: Ubuntu 16.10
OSType: linux
Architecture: x86_64
CPUs: 1
Total Memory: 3.613GiB
Name: docker-7
```



```
ID: R5TW:VKXK:EKGR:GHWM:UNU4:LPJH:IQY5:X77G:NNRQ:HWBY:LIUD:4ELQ
Docker Root Dir: /var/lib/docker
Debug Mode (client): false
Debug Mode (server): false
Registry: https://index.docker.io/v1/
Experimental: false
Insecure Registries:
127.0.0.0/8
Live Restore Enabled: false

WARNING: No swap limit support
$

\*\*$ sudo docker images\*\*
REPOSITORY TAG IMAGE ID CREATED SIZE
$

\*\*$ sudo docker ps\*\*
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
$
<!--NeedCopy-->
```

Paso 2: Configurar la comunicación de API remota con la instancia de Docker

Abra el puerto 4243 para la comunicación de la API con la instancia de Docker. Este puerto es necesario para que NetScaler ADM se comunique con la instancia de Docker.

```

 \*\*cd /etc/systemd/system\*\*
 \*\*sudo vi docker-tcp.socket\*\*
 \*\*cat docker-tcp.socket\*\*
[Unit]
\*\*Description=Docker Socket for the API
[Socket]
ListenStream=4243
BindIPv6Only=both
Service=docker.service
[Install]
WantedBy=sockets.target\*\*

$ \*\*sudo systemctl enable docker-tcp.socket\*\*
Created symlink /etc/systemd/system/sockets.target.wants/docker-tcp.
   socket → /etc/systemd/system/docker-tcp.socket.
 \*\*sudo systemctl enable docker.socket\*\*
 \*\*sudo systemctl stop docker\*\*
 \*\*sudo systemctl start docker-tcp.socket\*\*
 \*\*sudo systemctl start docker\*\*
$ \*\*sudo systemctl status docker\*\*
● docker.service - Docker Application Container Engine
Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor
   preset: enabled)
```



```
23 Active: \*\*active (running)\*\* since Wed 2017-05-31 12:52:17 UTC; 2s
      ago
24 Docs: https://docs.docker.com
25 Main PID: 4133 (dockerd)
26 Tasks: 16 (limit: 4915)
27 Memory: 30.1M
28 CPU: 184ms
29 CGroup: /system.slice/docker.service
30  ├─4133 /usr/bin/dockerd -H fd://
31  └─4137 docker-containerd -l unix:///var/run/docker/libcontainerd/docker
      -containerd.sock --metrics-interval=0 --start-timeout 2m -
32
33 May 31 12:52:17 docker-7 dockerd[4133]: time="2017-05-31T12
      :52:17.300890402Z" level=warning msg="Your kernel does not support
      cgroup rt peri
34 May 31 12:52:17 docker-7 dockerd[4133]: time="2017-05-31T12
      :52:17.301079754Z" level=warning msg="Your kernel does not support
      cgroup rt runt
35 May 31 12:52:17 docker-7 dockerd[4133]: time="2017-05-31T12
      :52:17.301681794Z" level=info msg="Loading containers: start."
36 May 31 12:52:17 docker-7 dockerd[4133]: time="2017-05-31T12
      :52:17.417539064Z" level=info msg="Default bridge (docker0) is
      assigned with an I
37 May 31 12:52:17 docker-7 dockerd[4133]: time="2017-05-31T12
      :52:17.465011600Z" level=info msg="Loading containers: done."
38 May 31 12:52:17 docker-7 dockerd[4133]: time="2017-05-31T12
      :52:17.484747909Z" level=info msg="Daemon has completed
      initialization"
39 May 31 12:52:17 docker-7 dockerd[4133]: time="2017-05-31T12
      :52:17.485119478Z" level=info msg="Docker daemon" commit=89658be
      graphdriver=aufs
40 May 31 12:52:17 docker-7 systemd[1]: Started Docker Application
      Container Engine.
41 May 31 12:52:17 docker-7 dockerd[4133]: time="2017-05-31T12
      :52:17.503832254Z" level=info msg="API listen on /var/run/docker.
      sock"
42 May 31 12:52:17 docker-7 dockerd[4133]: time="2017-05-31T12
      :52:17.504061522Z" level=info msg="API listen on [::]:4243"
43 $
44
45 (external)$ \*\*curl 104.199.209.157:4243/version\*\*
46 {
47  "Version":"17.05.0-ce","ApiVersion":"1.29","MinAPIVersion":"1.12","
       GitCommit":"89658be","GoVersion":"go1.7.5","Os":"linux","Arch":"
       amd64","KernelVersion":"4.8.0-52-generic","BuildTime":"2017-05-04
       T22:15:36.071254972+00:00" }
48
49 (external)$
50
51 <!--NeedCopy-->
```


Paso 3: instalar NetScaler CPX Image

Obtenga la imagen de NetScaler CPX en Docker App Store. El CPX Express y el CPX tienen la misma imagen. Sin embargo, cuando licencia e instala la imagen CPX con NetScaler ADM, la imagen se convierte en una instancia CPX completa con un rendimiento de 1 Gbps. Sin licencia, la imagen se convierte en una instancia CPX Express que admite conexiones SSL de 20 Mbps y 250.

```
$ \*\*sudo docker pull store/citrix/citrixadccpx:13.0-36.29\*\*
13.0-36.29: Pulling from store/citrix/citrixadccpx
4e1f679e8ab4: Pull complete
a3ed95caeb02: Pull complete
2931a926d44b: Pull complete
362cd40c5745: Pull complete
d10118725a7a: Pull complete
1e570419a7e5: Pull complete
d19e06114233: Pull complete
d3230f008ffd: Pull complete
22bdb10a70ec: Pull complete
1a5183d7324d: Pull complete
241868d4ebff: Pull complete
3f963e7ae2fc: Pull complete
fd254cf1ea7c: Pull complete
33689c749176: Pull complete
59c27bad28f5: Pull complete
588f5003e10f: Pull complete
Digest: sha256:31
   a65cfa38833c747721c6fbc142faec6051e5f7b567d8b212d912b69b4f1ebe
Status: Downloaded newer image for store/citrix/citrixadccpx:13.0-36.29
$

$ \*\*sudo docker images\*\*
REPOSITORY TAG IMAGE ID CREATED SIZE
store/citrix/citrixadccpx:13.0-36.29 6fa57c38803f 3 weeks ago 415MB
$
<!--NeedCopy-->
```

Paso 4: crear una instancia de NetScaler CPX

Instale la imagen de NetScaler CPX en el host de Docker. Abra puertos para servicios específicos, como se muestra en el siguiente ejemplo, y especifique una dirección IP para NetScaler ADM:

```
bash-2.05b# \*\*CHOST=${
 1:-localhost }
 \*\*
bash-2.05b# \*\*echo | openssl s_client -connect $CHOST:443 | openssl
   x509 -fingerprint -noout | cut -d'=' -f2\*\*
depth=0 C = US, ST = California, L = San Jose, O = NetScaler, OU =
   Internal, CN = Test Only Cert
verify error:num=18:self signed certificate
verify return:1
```



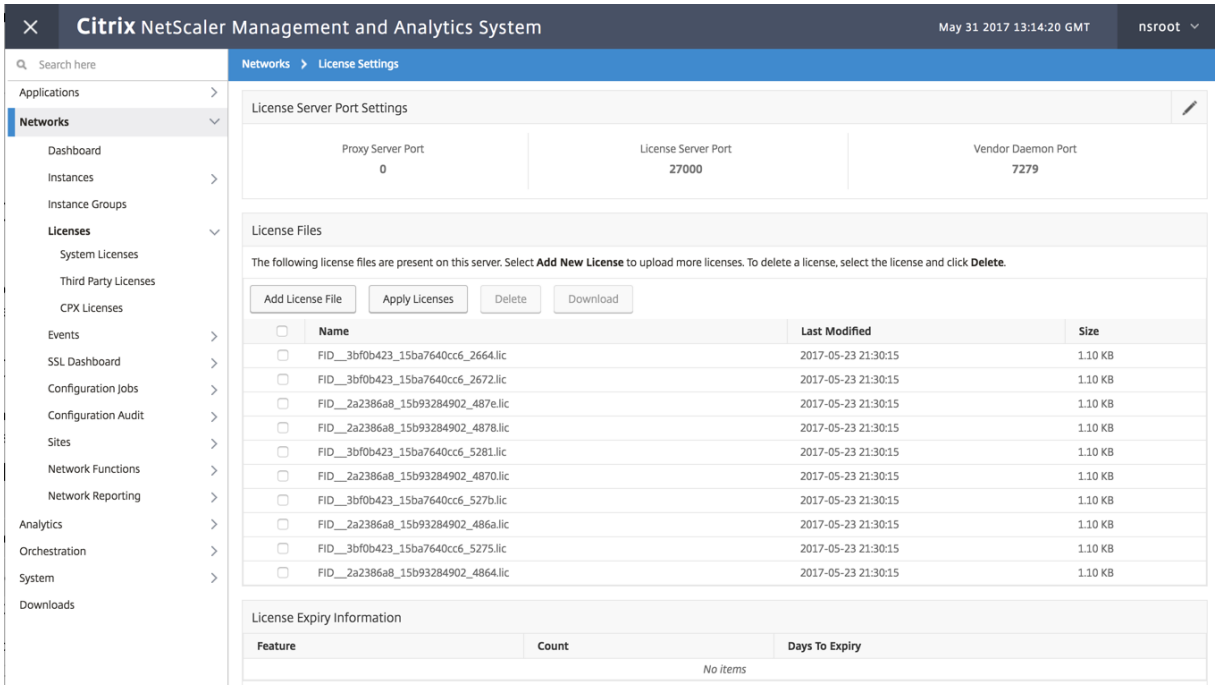
```
depth=0 C = US, ST = California, L = San Jose, O = NetScaler, OU =
   Internal, CN = Test Only Cert
verify return:1
DONE
24:AA:8B:91:7B:72:5E:6E:C1:FD:86:FA:09:B6:42:49:FC:1E:86:A4
bash-2.05b#

$ \*\*sudo docker run -dt -p 50000:88 -p 5080:80 -p 5022:22 -p 5443:443
    -p 5163:161/udp -e NS_HTTP_PORT=5080 -e NS_HTTPS_PORT=5443 -e
   NS_SSH_PORT=5022 -e NS_SNMP_PORT=5163 -e EULA=yes -e LS_IP=xx.xx.xx.
   xx -e PLATFORM=CP1000 --privileged=true --ulimit core=-1 -e
   NS_MGMT_SERVER=xx.xx.xx.xx:xxxx -e NS_MGMT_FINGER_PRINT=24:AA:8B
   :91:7B:72:5E:6E:C1:FD:86:FA:09:B6:42:49:FC:1E:86:A4 --env
   NS_ROUTABLE=false --env HOST=104.199.209.157 store/citrix/
   citrixadccpx:13.0-36.29\*\*
44ca1c6c0907e17a10ffcb9ffe33cd3e9f71898d8812f816e714821870fa3538
$

$ \*\*sudo docker ps\*\*
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
44ca1c6c0907 store/citrix/citrixadccpx:13.0-36.29 "/bin/sh -c 'bash ...
   " 19 seconds ago Up 17 seconds 0.0.0.0:5022->22/tcp,
   0.0.0.0:5080->80/tcp, 0.0.0.0:50000->88/tcp, 0.0.0.0:5163->161/udp,
   0.0.0.0:5443->443/tcp gifted_perlman
$

$ \*\*ssh -p 5022 root@localhost\*\*
root@localhost's password:
Welcome to nsoslx 1.0 (GNU/Linux 4.8.0-52-generic x86_64)

* Documentation: https://www.citrix.com/
Last login: Mon Jun 5 18:58:51 2017 from xx.xx.xx.xx
root@44ca1c6c0907:~#
root@44ca1c6c0907:~#
root@44ca1c6c0907:~# \*\*cli_script.sh 'show ns ip'\*\*
exec: show ns ip
Ipaddress Traffic Domain Type Mode Arp Icmp Vserver State
--------- -------------- ---- ---- --- ---- ------- ------
1) 172.17.0.2 0 NetScaler IP Active Enabled Enabled NA Enabled
2) 192.0.0.1 0 SNIP Active Enabled Enabled NA Enabled
Done
root@44ca1c6c0907:~# \*\*cli_script.sh 'show licenseserver'\*\*
exec: show licenseserver
1) ServerName: xx.xx.xx.xxPort: 27000 Status: 1 Grace: 0 Gptimeleft: 0
Done
root@44ca1c6c0907:~# cli_script.sh 'show capacity'
exec: show capacity
Actualbandwidth: 1000 Platform: CP1000 Unit: Mbps Maxbandwidth: 3000
   Minbandwidth: 20 Instancecount: 0
Done
root@44ca1c6c0907:~#

$ \*\*sudo iptables -t nat -L -n\*\*
```

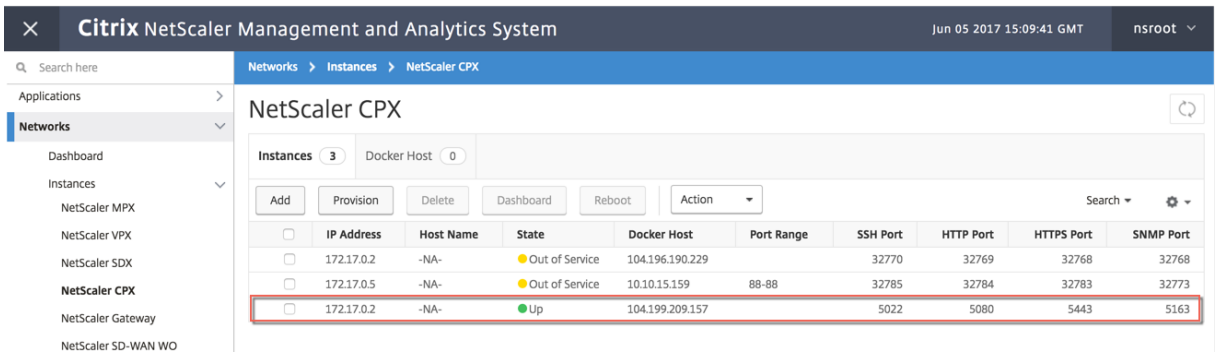


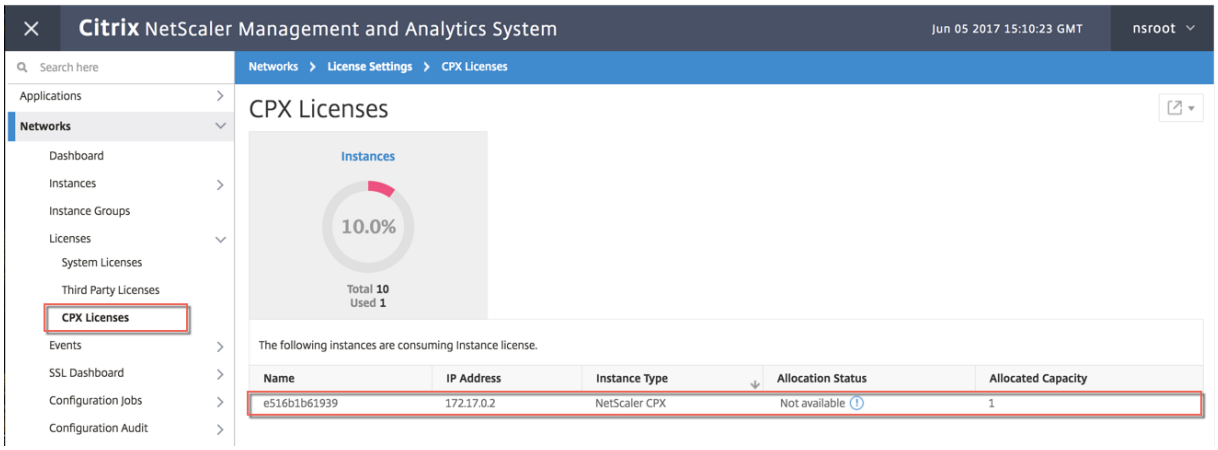
```
Chain PREROUTING (policy ACCEPT)
target prot opt source destination
DOCKER all -- 0.0.0.0/0 0.0.0.0/0 ADDRTYPE match dst-type LOCAL

Chain INPUT (policy ACCEPT)
target prot opt source destination

Chain OUTPUT (policy ACCEPT)
target prot opt source destination
DOCKER all -- 0.0.0.0/0 !127.0.0.0/8 ADDRTYPE match dst-type LOCAL

Chain POSTROUTING (policy ACCEPT)
target prot opt source destination
MASQUERADE all -- 172.17.0.0/16 0.0.0.0/0
MASQUERADE tcp -- 172.17.0.2 172.17.0.2 tcp dpt:443
MASQUERADE udp -- 172.17.0.2 172.17.0.2 udp dpt:161
MASQUERADE tcp -- 172.17.0.2 172.17.0.2 tcp dpt:88
MASQUERADE tcp -- 172.17.0.2 172.17.0.2 tcp dpt:80
MASQUERADE tcp -- 172.17.0.2 172.17.0.2 tcp dpt:22

Chain DOCKER (2 references)
target prot opt source destination
RETURN all -- 0.0.0.0/0 0.0.0.0/0
DNAT tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpt:5443 to:172.17.0.2:443
DNAT udp -- 0.0.0.0/0 0.0.0.0/0 udp dpt:5163 to:172.17.0.2:161
DNAT tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpt:50000 to:172.17.0.2:88
DNAT tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpt:5080 to:172.17.0.2:80
DNAT tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpt:5022 to:172.17.0.2:22
$
<!--NeedCopy-->
```

Paso 5: Licencia de NetScaler CPX a través de NetScaler ADM

Suponiendo que NetScaler ADM se ejecute localmente, debería poder validar que NetScaler CPX se está comunicando con NetScaler ADM y enviando información. En las siguientes imágenes se muestra a NetScaler CPX recuperando una licencia de NetScaler ADM.







Paso 6: Configurar los servicios de equilibrio de carga en NetScaler CPX y verificar la configuración

Primero, instale los servidores web NGINX en el host de Docker. A continuación, configure el equilibrio de carga en NetScaler CPX para equilibrar la carga de los dos servidores web y pruebe la configuración.

Instalar servidores web NGINX Use los comandos que se muestran en el siguiente ejemplo para instalar servidores web NGINX.

```
$ sudo docker pull nginx
Using default tag: latest
latest: Pulling from library/nginx
Digest: sha256:41
    ad9967ea448d7c2b203c699b429abe1ed5af331cd92533900c6d77490e0268
Status: Image is up to date for nginx:latest


\*\*$ sudo docker run -d -p 81:80 nginx\*\*
098a77974818f451c052ecd172080a7d45e446239479d9213cd4ea6a3678616f


\*\*$ sudo docker run -d -p 82:80 nginx\*\*
bbdac2920bb4085f70b588292697813e5975389dd546c0512daf45079798db65


\*\*$ sudo iptables -t nat -L -n\*\*
Chain PREROUTING (policy ACCEPT)
target prot opt source destination
DOCKER all -- 0.0.0.0/0 0.0.0.0/0 ADDRTYPE match dst-type LOCAL

Chain INPUT (policy ACCEPT)
target prot opt source destination

Chain OUTPUT (policy ACCEPT)
target prot opt source destination
DOCKER all -- 0.0.0.0/0 !127.0.0.0/8 ADDRTYPE match dst-type LOCAL

Chain POSTROUTING (policy ACCEPT)
target prot opt source destination
MASQUERADE all -- 172.17.0.0/16 0.0.0.0/0
MASQUERADE tcp -- 172.17.0.2 172.17.0.2 tcp dpt:443
MASQUERADE udp -- 172.17.0.2 172.17.0.2 udp dpt:161
MASQUERADE tcp -- 172.17.0.2 172.17.0.2 tcp dpt:88
MASQUERADE tcp -- 172.17.0.2 172.17.0.2 tcp dpt:80
MASQUERADE tcp -- 172.17.0.2 172.17.0.2 tcp dpt:22
MASQUERADE tcp -- 172.17.0.3 172.17.0.3 tcp dpt:80
MASQUERADE tcp -- 172.17.0.4 172.17.0.4 tcp dpt:80

Chain DOCKER (2 references)
target prot opt source destination
```



```
RETURN all -- 0.0.0.0/0 0.0.0.0/0
DNAT tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpt:5443 to:172.17.0.2:443
DNAT udp -- 0.0.0.0/0 0.0.0.0/0 udp dpt:5163 to:172.17.0.2:161
DNAT tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpt:50000 to:172.17.0.2:88
DNAT tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpt:5080 to:172.17.0.2:80
DNAT tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpt:5022 to:172.17.0.2:22
DNAT tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpt:81 to:172.17.0.3:80
DNAT tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpt:82 to:172.17.0.4:80
$
<!--NeedCopy-->
```

Configurar NetScaler CPX para el equilibrio de carga y verificar la distribución de la carga a ambos servicios web

```
$ \*\*ssh -p 5022 root@localhost\*\*
root@localhost's password:
Welcome to nsoslx 1.0 (GNU/Linux 4.8.0-52-generic x86_64)

* Documentation: https://www.citrix.com/
Last login: Mon Jun 5 18:58:54 2017 from 172.17.0.1
root@44ca1c6c0907:~#
root@44ca1c6c0907:~#
root@44ca1c6c0907:~#
root@44ca1c6c0907:~#
root@44ca1c6c0907:~# \*\*cli_script.sh "add service web1 172.17.0.3
   HTTP 80"\*\*
exec: add service web1 172.17.0.3 HTTP 80
Done
root@44ca1c6c0907:~# \*\*cli_script.sh "add service web2 172.17.0.4
   HTTP 80"\*\*
exec: add service web2 172.17.0.4 HTTP 80
Done
root@44ca1c6c0907:~# \*\*cli_script.sh "add lb vserver cpx-vip HTTP
   172.17.0.2 88"\*\*
exec: add lb vserver cpx-vip HTTP 172.17.0.2 88
Done
root@44ca1c6c0907:~# \*\*cli_script.sh "bind lb vserver cpx-vip web1
   "\*\*
exec: bind lb vserver cpx-vip web1
Done
root@44ca1c6c0907:~# \*\*cli_script.sh "bind lb vserver cpx-vip web2
   "\*\*
exec: bind lb vserver cpx-vip web2
Done
root@44ca1c6c0907:~#

root@44ca1c6c0907:~# \*\*cli_script.sh 'show lb vserver cpx-vip'\*\*
exec: show lb vserver cpx-vip

cpx-vip (172.17.0.2:88) - HTTP Type: ADDRESS
State: UP
Last state change was at Mon Jun 5 19:01:49 2017
Time since last state change: 0 days, 00:00:42.620
Effective State: UP
```



```
Client Idle Timeout: 180 sec
Down state flush: ENABLED
Disable Primary Vserver On Down : DISABLED
Appflow logging: ENABLED
Port Rewrite : DISABLED
No. of Bound Services : 2 (Total) 2 (Active)
Configured Method: LEASTCONNECTION
Current Method: Round Robin, Reason: A new service is bound
    BackupMethod: ROUNDROBIN
Mode: IP
Persistence: NONE
Vserver IP and Port insertion: OFF
Push: DISABLED Push VServer:
Push Multi Clients: NO
Push Label Rule: none
L2Conn: OFF
Skip Persistency: None
Listen Policy: NONE
IcmpResponse: PASSIVE
RHIstate: PASSIVE
New Service Startup Request Rate: 0 PER_SECOND, Increment Interval: 0
Mac mode Retain Vlan: DISABLED
DBS_LB: DISABLED
Process Local: DISABLED
Traffic Domain: 0
TROFS Persistence honored: ENABLED
Retain Connections on Cluster: NO

2) web1 (172.17.0.3: 80) - HTTP State: UP Weight: 1
3) web2 (172.17.0.4: 80) - HTTP State: UP Weight: 1
Done
root@44ca1c6c0907:~#

(external)$ \*\*curl 104.199.209.157:50000\*\*
\\<\\!DOCTYPE html\\>
\<html\>
\<head\>
\<title\>Welcome to nginx\!\</title\>
\<style\>
body {

width: 35em;
margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif;
 }

\\</style\\>
\\</head\\>
\<body\>
\<h1\>Welcome to nginx\!\</h1\>
\<p\>If you see this page, the nginx web server is successfully
    installed and
working. Further configuration is required.\\</p\\>
```



```
87
88  \<p\>For online documentation and support please refer to
89  \\<a href="http://nginx.org/"\\>nginx.org\\</a\\>.\\<br/\\>
90  Commercial support is available at
91  \\<a href="http://nginx.com/"\\>nginx.com\\</a\\>.\\</p\\>
92
93  \<p\>\<em\>Thank you for using nginx.\</em\>\</p\>
94  \\</body\\>
95  \\</html\\>
96  (external)$
97
98
99  (external)$ for i in {
100  1..100 }
101   ; \*\*do curl http://104.199.209.157:50000 -o /dev/null ; done\*\*
102
103   % Total    % Received % Xferd  Average Speed   Time    Time     Time
       Current
104
105                                  Dload  Upload   Total   Spent    Left
       Speed
106
107 100   612  100   612    0     0   1767      0 --:--:-- --:--:--
      --:--:--  1768
108
109   % Total    % Received % Xferd  Average Speed   Time    Time     Time
       Current
110
111                                  Dload  Upload   Total   Spent    Left
       Speed
112
113 100   612  100   612    0     0   1893      0 --:--:-- --:--:--
      --:--:--  1894
114
115   % Total    % Received % Xferd  Average Speed   Time    Time     Time
       Current
116
117                                  Dload  Upload   Total   Spent    Left
       Speed
118
119 100   612  100   612    0     0   1884      0 --:--:-- --:--:--
      --:--:--  1883
120
121   % Total    % Received % Xferd  Average Speed   Time    Time     Time
       Current
122
123                                  Dload  Upload   Total   Spent    Left
       Speed
124
125 100   612  100   612    0     0   1917      0 --:--:-- --:--:--
      --:--:--  1924
126
127   % Total    % Received % Xferd  Average Speed   Time    Time     Time
```



```
      Current
128
129                                 Dload  Upload   Total   Spent    Left
      Speed
130
131 100   612  100   612    0     0   1877      0 --:--:-- --:--:--
    --:--:--  1883
132
133   % Total    % Received % Xferd  Average Speed   Time    Time     Time
      Current
134
135                                 Dload  Upload   Total   Spent    Left
      Speed
136
137 100   612  100   612    0     0   1852      0 --:--:-- --:--:--
    --:--:--  1848
138
139   % Total    % Received % Xferd  Average Speed   Time    Time     Time
      Current
140
141                                 Dload  Upload   Total   Spent    Left
      Speed
142
143 100   612  100   612    0     0   1860      0 --:--:-- --:--:--
    --:--:--  1865
144
145   % Total    % Received % Xferd  Average Speed   Time    Time     Time
      Current
146
147                                 Dload  Upload   Total   Spent    Left
      Speed
148
149 100   612  100   612    0     0   1887      0 --:--:-- --:--:--
    --:--:--  1888
150
151   % Total    % Received % Xferd  Average Speed   Time    Time     Time
      Current
152
153                                 Dload  Upload   Total   Spent    Left
      Speed
154
155 100   612  100   612    0     0   1802      0 --:--:-- --:--:--
    --:--:--  1800
156
157   % Total    % Received % Xferd  Average Speed   Time    Time     Time
      Current
158
159                                 Dload  Upload   Total   Spent    Left
      Speed
160
161 100   612  100   612    0     0   1902      0 --:--:-- --:--:--
    --:--:--  1906
162
```



```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time
    Current
                                 Dload  Upload   Total   Spent    Left
    Speed
100   612  100   612    0     0   1843      0 --:--:-- --:--:--
   --:--:--  1848


  % Total    % Received % Xferd  Average Speed   Time    Time     Time
    Current
                                 Dload  Upload   Total   Spent    Left
    Speed
100   612  100   612    0     0   1862      0 --:--:-- --:--:--
   --:--:--  1860
  % Total    % Received % Xferd  Average Speed   Time    Time     Time
    Current
                                 Dload  Upload   Total   Spent    Left
    Speed
100   612  100   612    0     0   1806      0 --:--:-- --:--:--
   --:--:--  1810
  % Total    % Received % Xferd  Average Speed   Time    Time     Time
    Current
                                 Dload  Upload   Total   Spent    Left
    Speed
100   612  100   612    0     0   1702      0 --:--:-- --:--:--
   --:--:--  1704

(external)$



root@44ca1c6c0907:~# \*\*cli_script.sh 'stat lb vserver cpx-vip'\*\*

exec: stat lb vserver cpx-vip


Virtual Server Summary

                     vsvrIP  port     Protocol        State   Health
```



```
    actSvcs
204
205 cpx-vip           172.17.0.2    88         HTTP           UP      100
            2
206
207
208
209           inactSvcs
210
211 cpx-vip            0
212
213
214
215 Virtual Server Statistics
216
217                                              Rate (/s)
    Total
218
219 Vserver hits                                        0
    101
220
221 Requests                                           0
    101
222
223 Responses                                          0
              101
224
225 Request bytes                                      0
    8585
226
227 Response bytes                                     0
    85850
228
229 Total Packets rcvd                                 0
    708
230
231 Total Packets sent                                 0
    408
232
233 Current client connections                        --
     0
234
235 Current Client Est connections                    --
     0
236
237 Current server connections                        --
                      0
238
239 Current Persistence Sessions                      --
     0
240
241 Requests in surge queue                           --
     0
```



```
242
243 Requests in vserver's surgeQ                          --
        0
244
245 Requests in service's surgeQs                        --
        0
246
247 Spill Over Threshold                                 --
        0
248
249 Spill Over Hits                                      --
        0
250
251 Labeled Connection                                   --
        0
252
253 Push Labeled Connection                              --
        0
254
255 Deferred Request                                      0
        0
256
257 Invalid Request/Response                             --
        0
258
259 Invalid Request/Response Dropped                     --
        0
260
261 Vserver Down Backup Hits                             --
        0
262
263 Current Multipath TCP sessions                       --
        0
264
265 Current Multipath TCP subflows                       --
        0
266
267 Apdex for client response times.                     --
       1.00
268
269 Average client TTLB                                  --
        0
270
271 web1                172.17.0.3    80          HTTP            UP        51
            0/s
272
273 web2                172.17.0.4    80          HTTP            UP        50
            0/s
274
275 Done
276
277 root@44ca1c6c0907:~#
278 <!--NeedCopy-->
```


Solución de problemas de NetScaler CPX

November 23, 2023

En este documento se explica cómo solucionar los problemas que puede encontrar al utilizar NetScaler CPX. Con este documento, puede recopilar registros para determinar las causas y aplicar soluciones para algunos de los problemas comunes relacionados con la instalación y la configuración de NetScaler CPX.

- ¿Cómo puedo ver los registros de NetScaler CPX?

  Puede ver los registros de NetScaler CPX mediante el comando `kubectl logs` si NetScaler CPX se implementa con la opción `tty:true`. Puede ejecutar el siguiente comando para mostrar los registros:

  ```
  kubectl logs <pod-name> [-c <container-name>] [-n <namespace-name
     >]
  ```

  Por ejemplo,

  ```
  kubectl logs cpx-ingress1-69b9b8c648-t8bgn -c cpx -n citrix-adc
  ```

  A continuación, se muestra un ejemplo de la implementación de pods de NetScaler CPX con la opción `tty:true`:

  ```
    containers:
      - name: cpx-ingress
        image: "quay.io/citrix/citrix-k8s-cpx-ingress:13.0-58.30"
        tty: true
        securityContext:
           privileged: true
        env:

<!--NeedCopy-->
  ```

  Puede encontrar más registros de arranque en el archivo */cpx/log/boot.log* del sistema de archivos NetScaler CPX.

  Nota: Para obtener el nombre del pod, ejecute el comando `kubectl get pods -o wide`.

- ¿Cómo puedo recopilar el paquete de soporte técnico de NetScaler CPX?

  Puede ejecutar el siguiente comando en la interfaz shell del nodo principal de Kubernetes para recopilar el paquete de soporte técnico de NetScaler CPX:

  ```
  kubectl exec <cpx-pod-name> [-c <cpx-container-name>] [-n <
     namespace-name>] /var/netscaler/bins/cli_script.sh "show
     techsupport"
  ```


Puede ver el paquete de soporte técnico en el directorio */var/tmp/support* del sistema de archivos de NetScaler CPX. Use `scp` o `kubectl cp` para copiar el paquete de soporte técnico de NetScaler CPX al destino deseado.

Ejemplo:

```
root@localhost# kubectl exec cpx-ingress1-55b9b6fc75-t5kc6 -c cpx
    -n citrix-adc /var/netscaler/bins/cli_script.sh "show
   techsupport"
exec: show techsupport
  Scope:  NODE
Done
root@localhost# kubectl cp cpx-ingress1-55b9b6fc75-t5kc6:var/tmp/
   support/collector_P_192.168.29.232_31Aug2020_07_30.tar.gz /tmp
   /collector_P_192.168.29.232_31Aug2020_07_30.tar.gz -c cpx
root@localhost# ll /tmp/collector_P_192.168.29.232
   _31Aug2020_07_30.tar.gz
-rw-r--r-- 1 root root 1648109 Aug 31 13:23 /tmp/collector_P_192
   .168.29.232_31Aug2020_07_30.tar.gz
```

- ¿Por qué se bloquea el pod de NetScaler CPX durante el arranque?

  Puede comprobar el estado del pod con el comando `kubectl describe pods`. Ejecute el siguiente comando para conocer el estado del pod:

  ```
  kubectl describe pods <pod-name> [-c <container-name>] [-n <
     namespace-name>]
  ```

  Ejemplo:

  ```
  kubectl describe pods cpx-ingress1-69b9b8c648-t8bgn
  ```

  Si los eventos del pod muestran que el contenedor se inició, debe comprobar los registros del pod.

- ¿Cómo copio archivos entre el pod de NetScaler CPX y el nodo maestro de Kubernetes?

  Se recomienda utilizar la función de montaje de volumen de docker para montar el directorio `/cpx` en el sistema de archivos del host. Si un contenedor NetScaler CPX sale de los volcados de núcleo, los registros y otros datos importantes están disponibles en el punto de montaje.

  Puede usar cualquiera de los siguientes comandos para copiar archivos entre el pod de NetScaler CPX y el nodo maestro de Kubernetes:

  kubectl cp: Puede ejecutar este comando para copiar archivos del pod al nodo:

  ```
  kubectl cp <pod-name>:<absolute-src-path> <dst-path> [-c <
     container-name>] [-n <namespace-name>]
  ```

  Ejemplo:


```
root@localhost:~# kubectl cp cpx-ingress-596d56bb6-zbx6h:cpx/log/
    boot.log /tmp/cpx-boot.log -c cpx-ingress
root@localhost:~# ll /tmp/cpx-boot.log
-rw-r--r-- 1 root root 7880 Sep 11 00:07 /tmp/cpx-boot.log
```

scp: puede usar el comando para copiar archivos entre el pod de NetScaler CPX y el nodo de Kubernetes. Ejecute el siguiente comando para copiar archivos del pod al nodo. Cuando se le pida la contraseña, proporcione la contraseña para el usuario SSH:

```
scp <user>@<pod-ip>:<absolute-src-path> <dst-path>
```

Ejemplo:

```
root@localhost:~# scp nsroot@192.168.29.198:/cpx/log/boot.log /
    tmp/cpx-boot.log
nsroot@192.168.29.198's password:
boot.log
100% 7880     5.1MB/s   00:00
root@localhost:~#
```

- ¿Cómo capturo paquetes en NetScaler CPX?

  Para capturar paquetes en NetScaler CPX, inicie la interfaz shell de NetScaler CPX mediante el comando `kubectl exec`. Ejecute el siguiente comando para iniciar la interfaz shell del pod de NetScaler CPX:

  ```
  kubectl exec -it pod-name [-c container-name] [-n namespace-
      name] bash
  ```

  Ejemplo:

  ```
  kubectl exec -it cpx-ingress1-69b9b8c648-t8bgn -c cpx -n
      citrix-adc bash
  ```

  Y ejecute el siguiente comando para iniciar la captura de paquetes:

  ```
  cli_script.sh " start nstrace -size 0 "
  ```

  Si quiere detener la captura de paquetes en curso, ejecute el siguiente comando:

  ```
  cli_script.sh " stop nstrace "
  ```

  Puede ver los paquetes capturados en un archivo *.cap* en el directorio */cpx/nstrace/time-stamp* en el sistema de archivos NetScaler CPX.

- ¿Por qué el servidor de licencias no está configurado incluso cuando NetScaler CPX se implementa con la variable de entorno `LS_IP=<ADM-IP>`?

  Asegúrese de que se pueda acceder al servidor de licencias desde el nodo en el que se implementa NetScaler CPX. Puede usar el comando `ping <ADM-IP>` para verificar la conectividad del nodo NetScaler CPX a NetScaler ADM.

Si se puede acceder a NetScaler ADM desde el nodo, debe verificar los registros de configuración del servidor de licencias en el archivo */cpx/log/boot.log*. También puede comprobar la configuración del servidor de licencias mediante el siguiente comando en la interfaz shell del pod de NetScaler CPX:

```
cli_script.sh "show licenseserver"
```

Ejemplo:

```
root@cpx-ingress-596d56bb6-zbx6h:/cpx/log# cli_script.sh "show
    licenseserver"
exec: show licenseserver
ServerName:  10.106.102.199Port:  27000 Status:  1   Grace:  0
    Gptimeleft:  720
Done
```

- ¿Por qué la licencia agrupada no se configura en NetScaler CPX incluso después de una configuración correcta del servidor de licencias en NetScaler CPX?

  Compruebe los registros de configuración de licencias en el archivo */cpx/log/boot.log*. También puede verificar la licencia agrupada configurada en NetScaler CPX mediante el siguiente comando en la interfaz shell del pod de NetScaler CPX:

  ```
  cli_script.sh " show capacity "
  ```

  Por ejemplo,

  ```
  root@cpx-ingress-596d56bb6-zbx6h:/cpx/log# cli_script.sh "show
      capacity"
  exec: show capacity
  Actualbandwidth:  1000  MaxVcpuCount:  2  Edition:  Platinum
          Unit:  Mbps   Bandwidth:  0`  `Maxbandwidth:  40000
            Minbandwidth:  20       Instancecount:  1
  Done
  ```

  Además, asegúrese de que los archivos de licencia requeridos se carguen en el servidor de licencias. También puede verificar las licencias disponibles en el servidor de licencias una vez que se haya configurado correctamente en NetScaler CPX mediante el siguiente comando. Ejecute el comando en la interfaz shell del pod de NetScaler CPX:

  ```
  cli_script.sh " sh licenseserverpool "
  ```

  Ejemplo:

  ```
  root@cpx-ingress-596d56bb6-zbx6h:/cpx/log# cli_script.sh "show
      licenseserverpool"
  exec: show licenseserverpool
      Instance Total                    : 5
      Instance Available                : 4
      Standard Bandwidth Total          : 0 Mbps
  ```



```
      Standard Bandwidth Availabe         : 0 Mbps
      Enterprise Bandwidth Total          : 0 Mbps
      Enterprise Bandwidth Available      : 0 Mbps
      Platinum Bandwidth Total            : 10.00 Gbps
      Platinum Bandwidth Available        : 9.99 Gbps
      CP1000 Instance Total               : 100
      CP1000 Instance Available           : 100
    Done
<!--NeedCopy-->
```

- ¿Por qué las llamadas a la API NITRO obtienen *una respuesta de rechazo de conexión* de NetScaler CPX?

  El puerto predeterminado para las API de NITRO es 9080 (no seguro) y 9443 (seguro) a partir de la versión 12.1 de NetScaler CPX en adelante. Asegúrese de que el puerto NITRO de NetScaler CPX al que intenta acceder esté expuesto en el pod. Puede ejecutar el comando `kubectl describe` para ver el puerto expuesto y asignado del contenedor NetScaler CPX en la sección contenedor NetScaler CPX:

  ```
  kubectl describe pods <pod-name> | grep -i port
  ```

  Ejemplo:

  ```
      ng472 | grep -i port
         Ports:          80/TCP, 443/TCP, 9080/TCP, 9443/TCP
         Host Ports:     0/TCP, 0/TCP, 0/TCP, 0/TCP
           NS_HTTP_PORT:            9080
           NS_HTTPS_PORT:           9443
         Port:          <none>
         Host Port:     <none>
           NS_PORT:                 80
<!--NeedCopy-->
  ```

- ¿Por qué el proceso NSPPE en NetScaler CPX consume la mayor parte del uso de la CPU incluso cuando no hay tráfico o hay poco tráfico?

  Si NetScaler CPX se implementa con la variable de entorno `CPX_CONFIG='{ "YIELD":"NO" } '`, el proceso NSPPE consume el 100 por ciento de uso de la CPU incluso cuando no hay tráfico o hay poco tráfico. Si quiere que el proceso NSPPE no consuma el uso de la CPU, debe implementar NetScaler CPX sin la variable de entorno `CPX_CONFIG='{ "YIELD":"NO" }`. De forma predeterminada, el proceso NSPPE en CPX está configurado para no acaparar ni consumir el uso de la CPU.

- ¿Por qué NetScaler CPX no figura en NetScaler ADM incluso cuando se implementó con las variables de entorno necesarias para registrarse en NetScaler ADM?

  Puede encontrar los registros para el registro de NetScaler CPX con NetScaler ADM en el archivo */cpx/log/boot.log* del sistema de archivos NetScaler CPX.

Puede comprobar la accesibilidad de la dirección IP de NetScaler ADM desde el pod de NetScaler CPX mediante el comando `ping`. Además, asegúrese de que todas las variables de entorno necesarias para el registro de NetScaler ADM estén configuradas para el contenedor NetScaler CPX.

- `NS_MGMT_SERVER`: Especifica la dirección ADM-IP o el FQDN.
- `HOST`: Especifica la dirección IP del nodo.
- `NS_HTTP_PORT`: Especifica el puerto HTTP- asignado en el nodo.
- `NS_HTTPS_PORT`: Especifica el puerto HTTPS asignado en el nodo.
- `NS_SSH_PORT`: Especifica el puerto SSH asignado en el nodo.
- `NS_SNMP_PORT`: Especifica el puerto SNMP asignado en el nodo.
- `NS_ROUTABLE`: La dirección IP del pod de NetScaler CPX no se puede redirigir desde el exterior.
- `NS_MGMT_USER`: Especifica el nombre de usuario de ADM.
- `NS_MGMT_PASS`: Especifica la contraseña de ADM.

- ¿Por qué `cli_script.sh` se muestra un mensaje de error de *nombre de usuario o contraseña no válidos* después de cambiar la contraseña del usuario nsroot?

  El comando `cli_script.sh` es una utilidad de envoltura para NSCLI en NetScaler CPX. Ejecuta el primer argumento como cadena de comandos o ruta de archivo y el segundo argumento es opcional, que son las credenciales. Si se cambia la contraseña del usuario nsroot, debe proporcionar credenciales como segundo argumento para `cli_script.sh`. Puede ejecutar el siguiente comando para ejecutar NSCLI con credenciales:

  ```
  cli_script.sh " <command> "  " :<username>:<password> "
  ```

  Ejemplo:

  ```
  root@087a1e34642d:/# cli_script.sh "show ns ip"
  exec: show ns ip

  ERROR: Invalid username or password

  root@087a1e34642d:/# cli_script.sh "show ns ip" ":nsroot:
     nsroot123"

  exec: show ns ip

  Ipaddress        Traffic Domain         Type             Mode
         Arp     Icmp     Vserver  State
  --------------   ----                   ----             ---
     ----     -------  ------   -----
  172.17.0.3       0                  NetScaler IP     Active
     Enabled  Enabled  NA       Enabled
  192.0.0.1        0                  SNIP             Active
     Enabled  Enabled  NA       Enabled
  Done
  ```


- ¿Por qué falla SSH a NetScaler CPX con un usuario `root` y `nsroot`?

  A partir de la versión 13.0-64.35, NetScaler CPX genera una contraseña predeterminada y la actualiza para los usuarios SSH, `root` y `nsroot`. Si no ha cambiado la contraseña manualmente, la contraseña para los usuarios de SSH se puede encontrar `/var/deviceinfo/random_id` en el sistema de archivos de NetScaler CPX.