# NetScaler CPX 13.1

**Machine translated content**

## Disclaimer

# Contents



# Über NetScaler CPX

November 23, 2023

NetScaler CPX ist ein Container-basierter Anwendungsbereitstellungscontroller, der auf einem Docker-Host bereitgestellt werden kann. NetScaler CPX ermöglicht es Kunden, die Docker-Engine-Funktionen zu nutzen und NetScaler Load Balancing- und Traffic-Management-Funktionen für Container-basierte Anwendungen zu nutzen. Sie können eine oder mehrere NetScaler CPX-Instanzen als eigenständige Instanzen auf einem Docker-Host bereitstellen.

Eine NetScaler CPX-Instanz bietet einen Durchsatz von bis zu 1 Gbit/s.

Als containerisierter Formfaktor von NetScaler lässt sich NetScaler CPX gut in die Kubernetes-Umgebung integrieren und ist ein integraler Bestandteil der nativen Cloud-Lösung von NetScaler. Die native Cloud-Lösung von NetScaler hilft Ihnen dabei, Softwareanwendungen schnell, agil und effizient in einer Kubernetes-Umgebung zu erstellen und bereitzustellen. Mit der nativen Cloud-Lösung von NetScaler können Sie die Zuverlässigkeit und Sicherheit Ihrer Kubernetes-Umgebung auf Unternehmensebene gewährleisten.
Weitere Informationen finden Sie unter NetScaler Cloud Native Solution.

In diesem Dokument wird davon ausgegangen, dass Sie mit Docker vertraut sind und wie es funktioniert. Informationen zu Docker finden Sie in der Docker-Dokumentation unter https://docs.docker.com.

## Unterstützte Funktionen

NetScaler CPX unterstützt die folgenden Funktionen:

- Verfügbarkeit der Anwendung
  - L4-Loadbalancing und L7-Content Switching
  - SSL-Offloading
  - IPv6-Protokollübersetzung
  - Microsoft SQL, MySQL-Lastausgleich
  - AppExpert Ratensteuerung
  - Abonnenten-bewusste Verkehrssteuerung
  - Überspannungsschutz und Priority Queuing
  - Dynamische Routing-Protokolle
- Beschleunigung der Anwendung
  - Client- und Server-TCP-Optimierungen
  - Cache-Umleitung

  - AppCompress
  - AppCache

- Anwendungssicherheit
  - L7 Rewrite und Responder
  - L4-DoS-Abwehr
  - L7-DoS-Abwehr
  - Webanwendungsfirewall (WAF). NetScaler CPX unterstützt alle WAF-Funktionen, die von anderen NetScaler-Formfaktoren unterstützt werden. Informationen zu den unterstützten WAF-Features finden Sie unter NetScaler Web App Firewall.
  - Authentifizierung, Autorisierung und Überwachung (AAA) für den Anwendungsverkehr
- TCP-Protokolloptimierung
  - Mehrpfad-TCP
  - Binär erhöhen Congestion Control (BIC) und kubisches TCP
- Einfache Verwaltbarkeit
  - Webprotokollierung
  - AppFlow
  - NetScaler Application Delivery Management
  - Action-Analytik
- Optimierung der Anwendung
  - Integriertes Caching
- BGP-Routing und Route Health Injection (RHI)
- Hochverfügbarkeit (sowohl Layer 2 als auch Layer 3)

> **Hinweis:**
>
> Schnittstellenfunktionen wie Rx, Tx, GRO, GSO und LRO sind für Schnittstellen (Linux-Host) deaktiviert, die der NetScaler CPX-Appliance zugewiesen sind. Diese Funktionen bleiben auch nach dem Stoppen der NetScaler CPX-Appliance im deaktivierten Zustand. Außerdem wird die MTU für solche Schnittstellen auf 1500 Byte geändert.

### Unterstützte Plattformen

NetScaler CPX wird auf den folgenden Plattformen unterstützt:

- Kubernetes
- Red Hat OpenShift
- Öffentliche Clouds

  - Amazon Elastic Kubernetes-Dienst (EKS)
  - Azure Kubernetes-Dienst (AKS)
  - Google Kubernetes Engine (GKE)

- Rancher
- Pivotal-Containerdienst (PKS)
- Docker Version 1.12 und höher

## Architektur und Verkehrsfluss

November 23, 2023

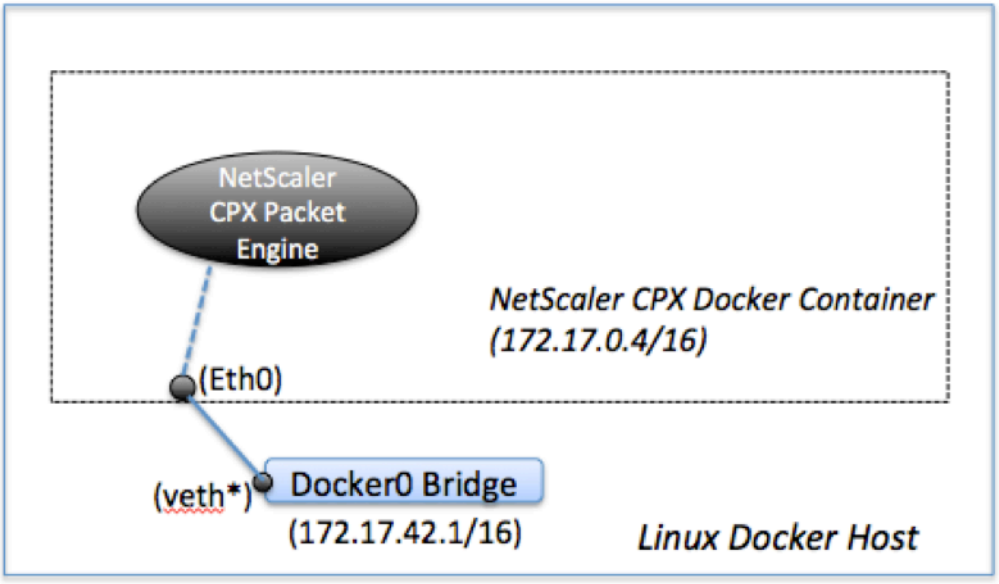
In diesem Abschnitt werden die Architektur und der Verkehrsfluss im NetScaler CPX-Brückenmodus beschrieben. NetScaler CPX kann auch im Hostmodus bereitgestellt werden.

Wenn Sie eine NetScaler CPX-Instanz auf einem Docker-Host bereitstellen, erstellt die Docker-Engine eine virtuelle Schnittstelle, eth0, auf der CPX-Instanz. Diese eth0-Schnittstelle ist direkt mit einer virtuellen Schnittstelle (veth*) auf der Docker0-Brücke verbunden. Die Docker-Engine weist der NetScaler CPX-Instanz im Netzwerk 172.17.0.0/16 auch eine IP-Adresse zu.

Das Standard-Gateway für die CPX-Instanz ist die IP-Adresse der Docker0-Brücke, was bedeutet, dass die Kommunikation mit der NetScaler CPX-Instanz über das Docker-Netzwerk erfolgt. Der gesamte eingehende Datenverkehr, der von der Docker0-Brücke empfangen wird, wird von der eth0-Schnittstelle der NetScaler CPX-Instanz empfangen und von der NetScaler CPX-Paket-Engine verarbeitet.

Die folgende Abbildung veranschaulicht die Architektur einer NetScaler CPX-Instanz auf einem Docker-Host.




### So funktioniert eine einzelne IP-Adresse auf NetScaler CPX

Eine reguläre NetScaler MPX- oder VPX-Appliance benötigt mindestens drei IP-Adressen, um zu funktionieren:

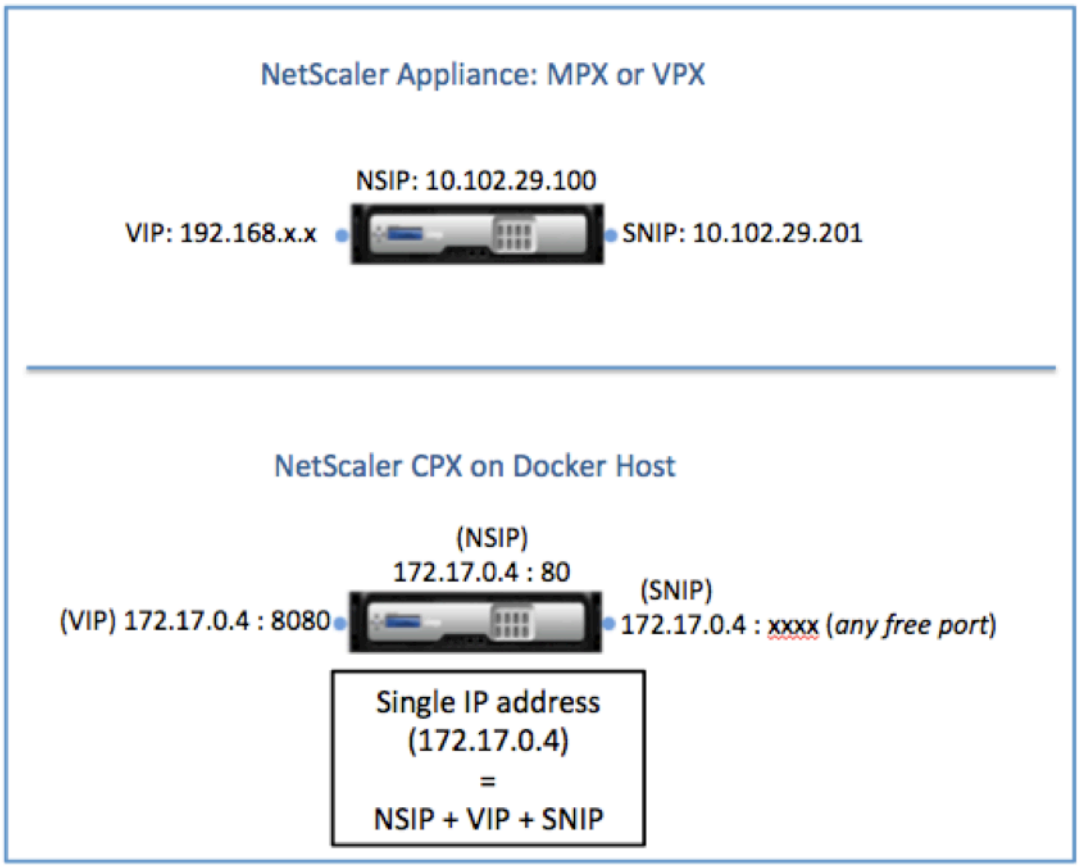
- Management-IP-Adresse, die als NetScaler IP (NSIP) -Adresse bezeichnet wird
- Subnetz-IP (SNIP) -Adresse für die Kommunikation mit der Serverfarm
- Virtuelle Server-IP (VIP) -Adresse (n) zum Akzeptieren von Clientanfragen

Eine NetScaler CPX-Instanz arbeitet mit einer einzigen IP-Adresse, die sowohl für die Verwaltung als auch für den Datenverkehr verwendet wird.

Während der Bereitstellung wird nur eine private IP-Adresse (einzelne IP-Adresse) einer NetScaler CPX-Instanz von der Docker-Engine zugewiesen. Die drei IP-Funktionen einer NetScaler-Instanz werden auf eine IP-Adresse gemultiplext. Diese einzelne IP-Adresse verwendet unterschiedliche Portnummern, um als NSIP, SNIP und VIP (s) zu fungieren.

Das folgende Bild zeigt, wie eine einzelne IP-Adresse verwendet wird, um die Funktionen von NSIP, SNIP und VIP (s) auszuführen.

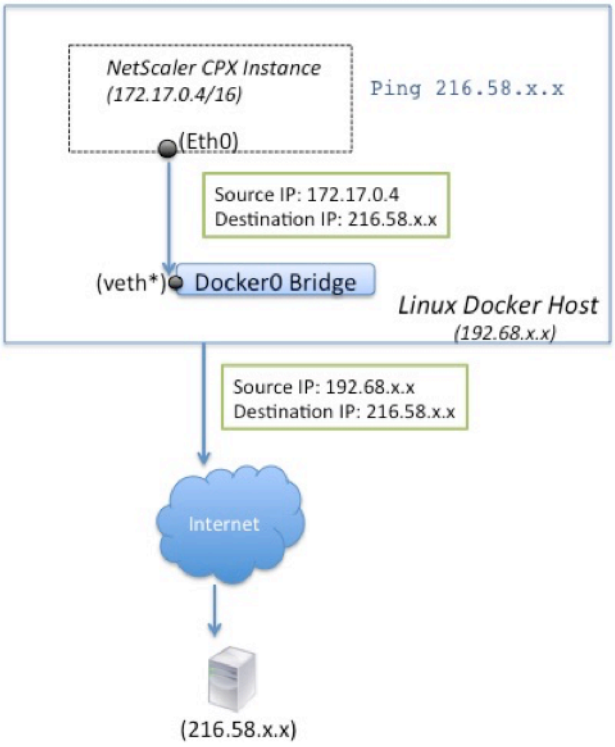



### Verkehrsfluss für Anforderungen, die von der NetScaler CPX-Instanz stammen

Docker konfiguriert implizit IP-Tabellen und eine NAT-Regel, um den Datenverkehr von der NetScaler CPX-Instanz an die Docker0-IP-Adresse zu leiten.

Die folgende Abbildung zeigt, wie eine Ping-Anforderung, die von einer NetScaler CPX-Instanz stammt, das Ziel erreicht.




In diesem Beispiel wird die Ping-Anforderung von der Paket-Engine auf der eth0-Schnittstelle mit der Quell-IP-Adresse als NetScaler CPX-IP-Adresse (172.17.0.4) gesendet. Der Docker-Host führt dann die Netzwerkadressübersetzung (NAT) durch, um die Host-IP-Adresse (192.68.x.x) als Quell-IP-Adresse hinzuzufügen, und sendet die Anforderung an das Ziel (216.58.x.x). Die Antwort von der Ziel-IP-Adresse folgt umgekehrt demselben Pfad. Der Docker-Host führt NAT für die Antwort aus und leitet die Antwort an die NetScaler CPX-Instanz auf der eth0-Schnittstelle weiter.

### Verkehrsfluss für Anforderungen aus dem externen Netzwerk

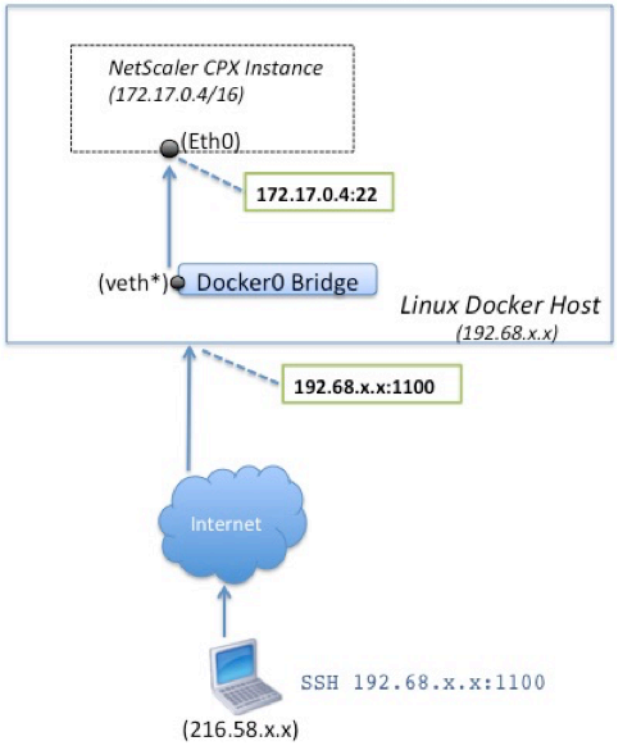
Um die externe Kommunikation zu ermöglichen, müssen Sie bei der Bereitstellung von NetScaler CPX Parameter festlegen, sodass Docker bestimmte Ports wie 80, 22 und jeden anderen gewünschten Port verfügbar macht. Wenn Sie keinen Port festgelegt haben, der während der Bereitstellung verfügbar gemacht werden soll, müssen Sie NAT-Regeln auf dem Docker-Host konfigurieren, um diese Ports verfügbar zu machen.

Die Clientanforderung, die aus dem Internet stammt, wird vom Docker-Host empfangen, der dann die Portadressübersetzung (PAT) durchführt, um die öffentliche IP-Adresse und den Port der einzelnen IP-Adresse und dem Port der NetScaler CPX-Instanz zuzuordnen und den Datenverkehr an die Instanz weiterzuleiten.

Die folgende Abbildung zeigt, wie der Docker-Host die Portadressübersetzung durchführt, um den

Datenverkehr an die einzelne IP-Adresse und den Port von NetScaler CPX zu leiten.




In diesem Beispiel lautet die Docker-Host-IP-Adresse 192.68.x.x und die einzelne IP-Adresse der NetScaler CPX-Instanz ist 172.17.0.4. Der SSH-Port 22 der NetScaler CPX-Instanz ist Port 1100 auf dem Docker-Host zugeordnet. Die SSH-Anforderung vom Client wird auf der IP-Adresse 192.68.x.x an Port 1100 empfangen. Der Docker-Host führt die Portadressübersetzung durch, um diese Adresse und den Port der einzelnen IP-Adresse 172.17.0.4 auf Port 22 zuzuordnen, und leitet die Clientanfrage weiter.

## NetScaler CPX-Lizenzierung

March 21, 2024

NetScaler CPX ist ein Container-basierter Anwendungsbereitstellungscontroller, der auf einem Docker-Host bereitgestellt werden kann, um Microservice-basierte Anwendungen zu Lastenausgleich zu ermöglichen. Sie benötigen lizenziertes CPX für eine bessere Leistung der Anwendungsbereitstellung. NetScaler CPX unterstützt die Poollizenzierung. NetScaler ADM kann als Lizenzserver fungieren, um Ihre NetScaler CPX-Instanzen zu lizenzieren.

NetScaler ADM ist sowohl on-premises als auch als Cloud-Dienst verfügbar. Sie können NetScaler ADM

verwenden, um gepoolte Kapazitätslizenzen für alle NetScaler-Formfaktoren zu verwalten.

Informationen zu NetScaler ADM on-premises finden Sie unter NetScaler ADM on-premises. Informationen zum NetScaler ADM Service finden Sie unter NetScaler ADM Service.

## Arten der NetScaler CPX-Lizenzierung

NetScaler CPX unterstützt Bandbreite und virtuelle CPU (Kern) Pool-Lizenzierung für lokale und Cloud-basierte Bereitstellungen.

**Bandbreitenpool**: NetScaler CPX-Lizenzen können basierend auf dem Bandbreitenverbrauch der Instanzen zugewiesen werden. Sie können die gepoolte Lizenzierung verwenden, um die Bandbreitennutzung zu maximieren, indem Sie die erforderliche Bandbreitenzuweisung zu einer Instanz sicherstellen und nicht mehr als Derzeit unterstützt NetScaler CPX nur die Premium-Bandbreiten-Poollizenzierung.

**vCPU-Pool**: In der nutzungsbasierten Lizenzierung der virtuellen CPU gibt die Lizenz die Anzahl der CPUs an, auf die eine bestimmte NetScaler CPX-Instanz berechtigt ist. Der NetScaler CPX kann also Lizenzen nur für die Anzahl der virtuellen CPUs vom Lizenzserver auschecken. NetScaler CPX checkt Lizenzen je nach Anzahl der im System ausgeführten CPUs aus. Weitere Informationen zum vCPU-Pool finden Sie unter NetScaler Virtual CPU-Lizenzierung.

### Unterstützte gepoolte Kapazität für NetScaler CPX-Instanzen

| Produkt | Maximale Bandbreite | Minimale Bandbreite | Minimale Instanzen | Maximale Anzahl Instanzen | Einheit für minimale Bandbreite |
|---|---|---|---|---|---|
| NetScaler CPX | 40.000<br>**Hinweis**: Dies hängt von der CPU-Frequenz, der Generierung usw. ab. | 20 Mbit/s | 1 | 16 | 10 MBit/s |

**Hinweis:** Citrix arbeitet derzeit an einem nutzungsbasierten oder nutzungsbasierten NetScaler CPX-Lizenzmodell für Public Cloud-basierte Angebote. Sobald es fertig ist, wird es auf dem Public Cloud-Marktplatz zum Konsumieren verfügbar sein.

## Wie funktioniert die NetScaler CPX-Lizenzierung?

**NetScaler CPX gepoolte Kapazität:** Ein gemeinsamer Lizenzpool, aus dem Ihre NetScaler CPX-Instanz eine Instanzlizenz und nur so viel Bandbreite auschecken kann, wie sie benötigt. Wenn die Instanz diese Ressourcen nicht mehr benötigt, checkt sie sie wieder in den gemeinsamen Pool ein und stellt die Ressourcen anderen Instanzen zur Verfügung, die diese Lizenzen benötigen.

**NetScaler CPX Check-In- und Check-Out-Lizenzierung:** NetScaler ADM weist bei Bedarf Lizenzen für NetScaler CPX-Instanzen zu. Eine NetScaler CPX-Instanz kann die Lizenz von NetScaler ADM auschecken, wenn eine NetScaler CPX-Instanz bereitgestellt wird, und ihre Lizenz für NetScaler ADM erneut einchecken, wenn eine Instanz zerstört wird.

**NetScaler CPX-Verhalten:** Eine einzelne NetScaler CPX-Instanz, die einen Durchsatz von bis zu 1 Gbit/s auscheckt und nur aus dem Instanzpool und nicht aus dem Bandbreitenlizenzpool auscheckt. NetScaler CPX arbeitet auf diese Weise bis zu 1 Gbit/s Bandbreitennutzung. Wenn eine CPX-Instanz beispielsweise eine Bandbreite von 200 Mbit/s verbraucht, verwendet sie den Instanz-Lizenzpool anstelle des Bandbreitenpools. Wenn eine NetScaler CPX-Instanz jedoch einen Durchsatz von 1200 Mbit/s verbraucht, werden die ersten 1000 Mbit/s aus dem Instanzpool verwendet und die verbleibenden 200 Mbit/s werden aus dem Bandbreitenpool verbraucht.

### NetScaler CPX Express

NetScaler CPX Express ist eine Software-Edition, die für on-premises und Cloud-Bereitstellungen kostenlos ist. Wenn Sie die NetScaler CPX-Instanz aus dem Quay-Repository herunterladen, ist dies die Standardkapazität, die für POCs verfügbar ist, die keine Lizenzdatei benötigen, und sie verfügt über die folgenden Funktionen:

- 20 Mbit/s Bandbreite
- Maximal 250 SSL-Sitzungen
- 20 Mbit/s SSL-Durchsatz

Sie müssen Ihre NetScaler CPX-Instanz lizenzieren, um ein Upgrade für bessere Leistung und Produktionsbereitstellungen zu erzielen.

## NetScaler CPX-Lizenzmodelle

NetScaler bietet eine Reihe von Produktlizenzierungsmodellen für NetScaler CPX, um die Anforderungen Ihres Unternehmens zu erfüllen. Sie können Optionen wie vCPU oder Bandbreite und on-premises oder Cloud auswählen.

Je nach Ihren Anforderungen können Sie eines der folgenden Modelle wählen:

- Bandbreitenbasierte Lizenzierung für NetScaler CPX vom ADM-Dienst

- vCPU-basierte Lizenzierung für NetScaler CPX vom ADM-Dienst
- Bandbreitenbasierte Lizenzierung für NetScaler CPX von ADM on-premises
- vCPU-basierte Lizenzierung für NetScaler CPX von ADM on-premises

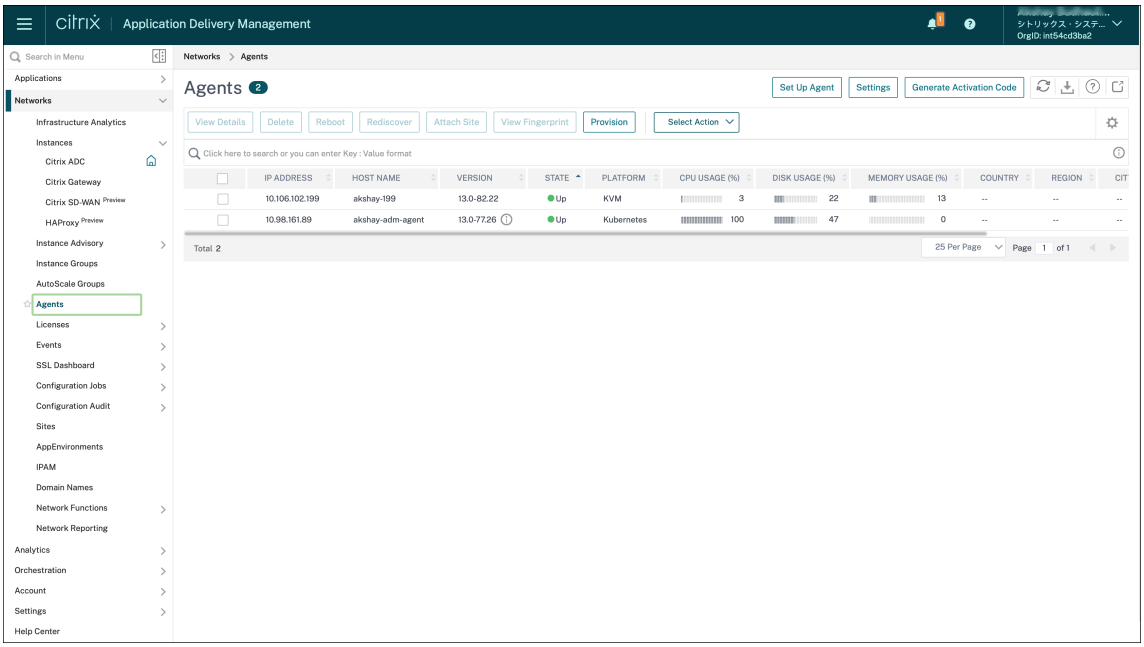
### Bereitstellen der bandbreitenbasierten und vCPU-basierten Lizenzierung vom NetScaler ADM Service für NetScaler CPX

Führen Sie die folgenden Schritte aus, um eine bandbreitenbasierte Lizenz und eine vCPU-basierte Lizenz für NetScaler CPX vom NetScaler ADM Service bereitzustellen.

1. Richten Sie NetScaler ADM ein.

   Stellen Sie sicher, dass das NetScaler ADM Service-Setup mit dem NetScaler ADM Agenten betriebsbereit ist. Sie müssen ein NetScaler ADM Service- und ein NetScaler ADM Agent-Konto haben, damit die NetScaler CPX-Lizenzierung funktionsfähig ist. Informationen zum Einrichten von NetScaler ADM Service und NetScaler ADM Agent finden Sie unter NetScaler ADM Service.

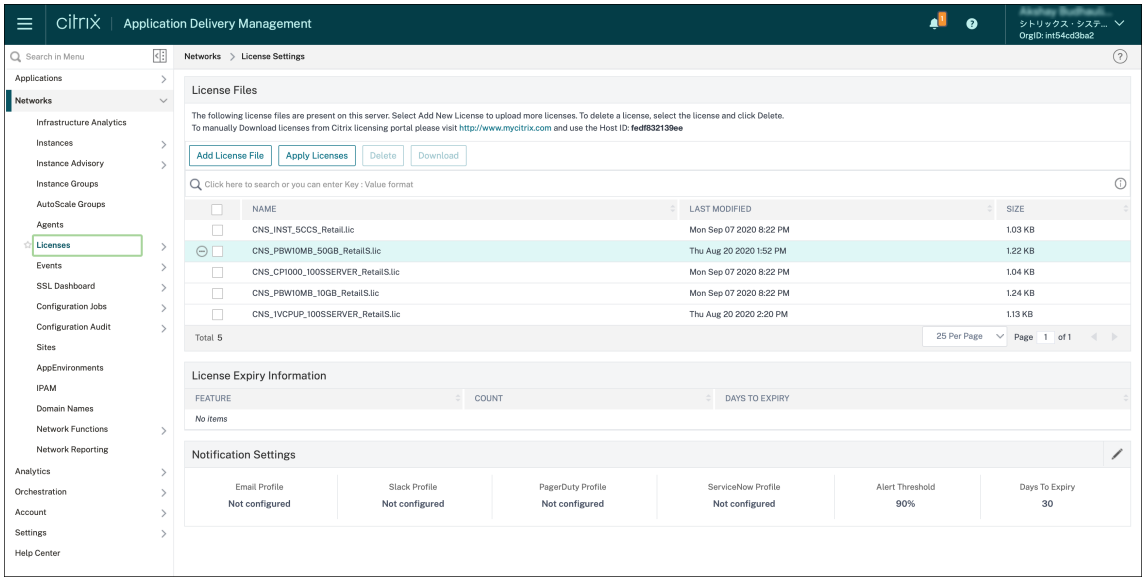
   **Hinweis:** In diesem Verfahren wird ein NetScaler ADM Agent-Setup für Hypervisor (lokal) verwendet. In der folgenden Abbildung ist `10.106.102.199` der on-premises Agent, der für die Lizenzierung von NetScaler CPX verwendet wird.




2. Fügen Sie den NetScaler-Instanzlizenzpool zum NetScaler ADM Service hinzu.

   Es wird davon ausgegangen, dass Sie einen Pool von Bandbreitenlizenzen für ADM Service haben. Informationen zum Hochladen einer Lizenzdatei auf NetScaler ADM finden Sie unter Konfigurieren der gepoolten Kapazität. In der folgenden Abbildung `CNS_INST_200CC_Retail.lic` wird als Bandbreite- und Instanzlizenzpool verwendet.




3. Stellen Sie die NetScaler CPX-Instanz im Kubernetes-Cluster bereit. Stellen Sie sicher, dass die folgenden Umgebungsvariablen zur NetScaler CPX YAML-Datei hinzugefügt werden, um die NetScaler CPX-Instanz zu lizenzieren.

   Geben Sie für die bandbreitenbasierte Lizenzierung vom NetScaler ADM Service die folgenden Umgebungsvariablen in der YAML-Datei an:

   - name: “LS_IP”
     -Wert: “10.105.158.166”//ADM-Agent-IP wie in Schritt 1 erwähnt
   - name: “LS_PORT”
     Wert: “27000”//Port, auf dem der ADM-Lizenzserver lauscht
   - name: “
     BANDWIDTH”-Wert: “3000”//Kapazität in Mbit/s möchte CPX zuweisen
   - name: “EDITION”
     -Wert: “Standard”oder “Enterprise”//um eine bestimmte Lizenzversion auszuwählen, die Standard, Platinum und Enterprise umfasst. Standardmäßig ist Platin ausgewählt.

   Geben Sie für die vCPU-basierte Lizenzierung vom NetScaler ADM Service die folgenden Umgebungsvariablen in der YAML-Datei an:

   - name: “LS_IP”
     -Wert: “10.102.216.173”//ADM-Agent-IP wie in Schritt 1 erwähnt
   - name: “LS_PORT”
     Wert: “27000”//Port, auf dem der ADM-Lizenzserver hört
   - name: “CPX_CORES”
     Wert: “4”//Anzahl der Kerne, die Sie zuweisen möchten
   - name: “PLATFORM”
     -Wert: “CP1000”//Anzahl der Kerne. Die Anzahl der Check-outs entspricht der Anzahl der

Kerne.

4. Laden Sie die Datei `cpx-bandwidth-license-adm-service.yaml` mit dem folgenden Befehl herunter:

```
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. Stellen Sie die bearbeitete YAML mit dem folgenden Befehl im Kubernetes-Cluster bereit:

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

6. Melden Sie sich mit dem folgenden Befehl bei NetScaler CPX an, um die Instanzinformationen zu überprüfen:

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

7. Führen Sie die folgenden Befehle aus, um die Lizenzinformationen für die angegebene NetScaler CPX-Instanz anzuzeigen:

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

Sie können die zugewiesene Bandbreite und vCPU-Kapazität im ADM-Dienstportal verfolgen.

### Bereitstellen bandbreitenbasierter Lizenzierung und vCPU-basierter Lizenzierung für NetScaler CPX von NetScaler ADM on-premises

Führen Sie die folgenden Schritte aus, um bandbreitenbasiert und vCPU-basiert für NetScaler CPX von NetScaler ADM on-premises bereitzustellen.

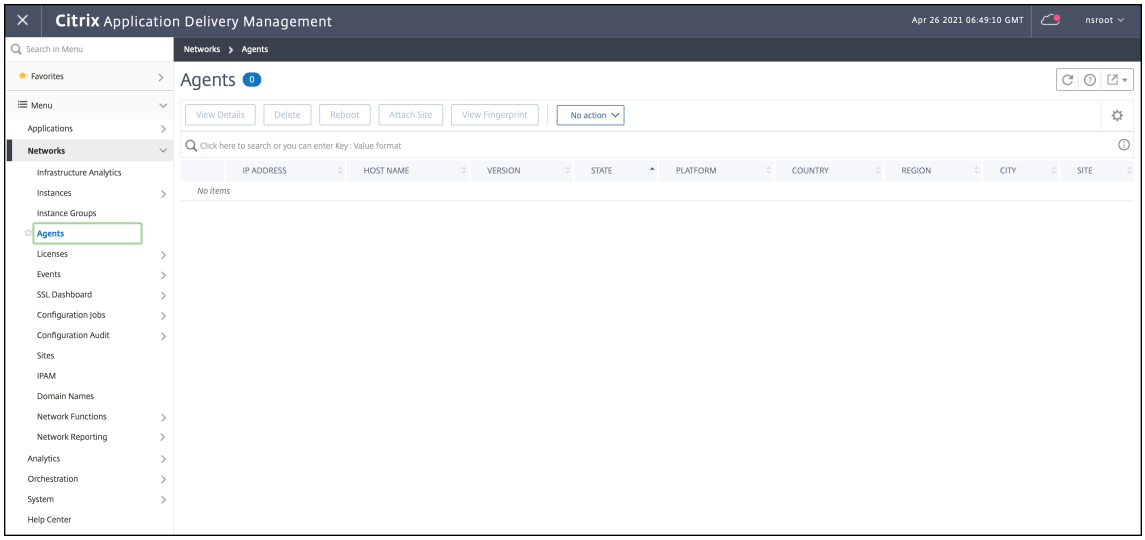
1. Richten Sie NetScaler ADM ein.

   Stellen Sie sicher, dass das ADM-Setup on-premises bereit ist. Stellen Sie sicher, dass NetScaler ADM on-premises mit oder ohne ADM-Agentenbereitstellung für die NetScaler CPX-Lizenzierung funktioniert.

   Informationen zum on-premises Einrichten von NetScaler ADM und NetScaler ADM Agent finden Sie unter NetScaler ADM Service.

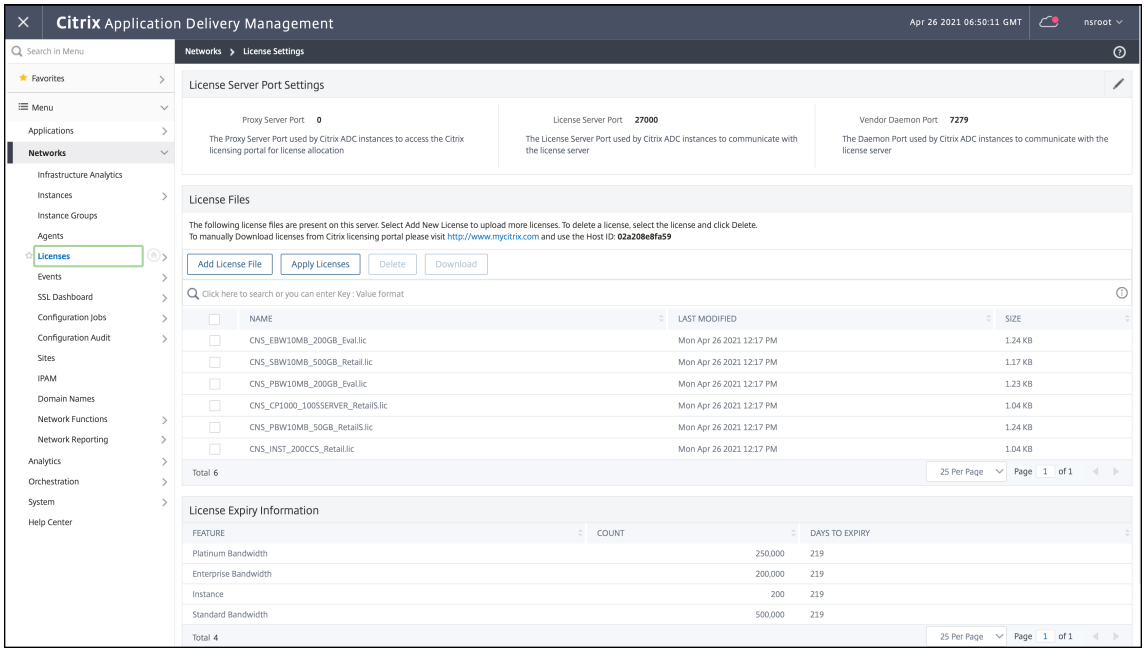
   **Hinweis:** In diesem Beispiel wird ein integrierter ADM-Agent mit ADM on-premises verwendet. In der folgenden Abbildung sehen Sie, dass kein Agent bereitgestellt ist.



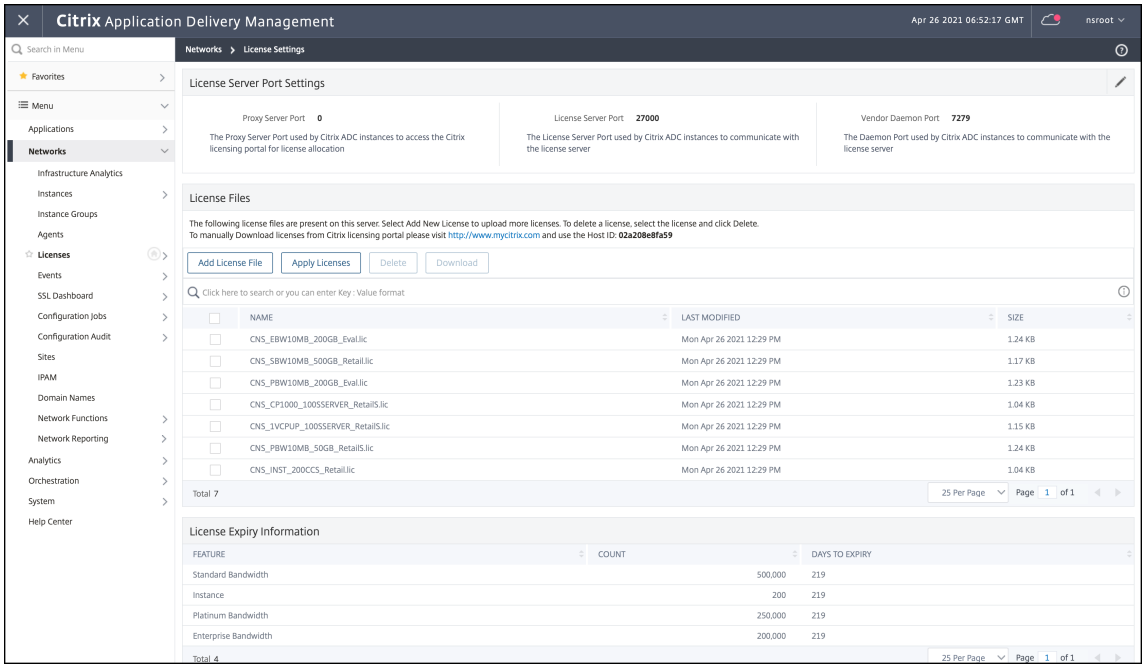

2. Fügen Sie den NetScaler-Instanzlizenzpool on-premises zu ADM hinzu.

   Es wird davon ausgegangen, dass Sie über einen Pool mit Bandbreitenlizenzen verfügen, die on-premises für ADM verfügbar sind. Informationen zum Hochladen einer Lizenzdatei auf NetScaler ADM finden Sie unter Lizenzierung. In der folgenden Abbildung `CNS_INST_200CC_Retail.lic` wird als Bandbreite- und Instanzlizenzpool verwendet.




   In der folgenden Abbildung wird CP1000 als vCPU-Lizenzpool verwendet.




3. Stellen Sie die NetScaler CPX-Instanz im Kubernetes-Cluster bereit. Stellen Sie sicher, dass die folgenden Umgebungsvariablen zur NetScaler CPX YAML-Datei hinzugefügt werden, um die NetScaler CPX-Instanz zu lizenzieren.

   Geben Sie für die bandbreitenbasierte Lizenzierung von NetScaler ADM on-premises die folgenden Umgebungsvariablen in der YAML-Datei an:

   - name: “LS_IP”
     -Wert: “10.105.158.144”//ADM-On-Prem-Instanz-IP, Wenn Sie den ADM-Agenten bereitgestellt haben, ist dies Ihre Agent-IP-Adresse, wie in Schritt 1 beschrieben
   - name: “LS_PORT”
     Wert: “27000”//Port, auf dem der ADM-Lizenzserver lauscht
   - name: “
     BANDWIDTH”-Wert: “3000”//Kapazität in Mbit/s möchte CPX zuweisen

   Geben Sie für die vCPU-basierte Lizenzierung von NetScaler ADM on-premises die folgenden Umgebungsvariablen in der YAML-Datei an:

   - name: “LS_IP”
     -Wert: “10.105.158.144”//ADM-On-Prem-Instanz-IP, Wenn Sie über eine ADM-Agentenausbringung verfügen, ist dies Ihre Agent-IP, wie in Schritt 1 beschrieben
   - name: “LS_PORT”
     Wert: “27000”//Port, auf dem der ADM-Lizenzserver hört
   - name: “CPX_CORES”
     Wert: “4”//die Anzahl der Kerne, die Sie zuweisen möchten
   - name: “PLATFORM”

-Wert: “CP1000”//Anzahl der Kerne. Die Anzahl der Check-outs entspricht der Anzahl der Kerne.

4. Laden Sie die Datei `cpx-bandwidth-license-adm-onprem.yaml` mit dem folgenden Befehl herunter:

```
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. Stellen Sie die bearbeitete YAML mit dem folgenden Befehl im Kubernetes-Cluster bereit:

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

6. Melden Sie sich mit dem folgenden Befehl bei NetScaler CPX an, um die Instanzinformationen zu überprüfen:

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

7. Führen Sie die folgenden Befehle aus, um die Lizenzinformationen für die NetScaler CPX-Instanz anzuzeigen:

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

Sie können die zugewiesene Bandbreite und vCPU-Kapazität im on-premises ADM-Portal verfolgen.

#### Befehle zum Reinigen der Bereitstellungen

Sie können die folgenden Befehle verwenden, um die verschiedenen YAML-Bereitstellungen zu bereinigen:

```
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
```

## Bereitstellen einer NetScaler CPX-Instanz in Docker

November 23, 2023

NetScaler CPX-Instanzen sind als Docker-Imagedatei in der Quay-Container-Registrierung verfügbar. Um eine Instanz bereitzustellen, laden Sie das NetScaler CPX-Image aus der Quay-Container-Registrierung herunter und stellen Sie die Instanz dann mit dem Befehl `docker run` oder mit Docker Compose Tool bereit.

### Voraussetzungen

Stellen Sie sicher, dass:

- Das Docker-Hostsystem verfügt über mindestens:
  - 1 CPU
  - 2 GB RAM

    > **Hinweis:** Für eine bessere NetScaler CPX-Leistung können Sie die Anzahl der Verarbeitungs-Engines definieren, die die NetScaler CPX-Instanz starten soll. Stellen Sie für jede zusätzliche Verarbeitungsengine, die Sie hinzufügen, sicher, dass der Docker-Host die entsprechende Anzahl von vCPUs und die Menge an Arbeitsspeicher in GB enthält. Wenn Sie beispielsweise 4 Verarbeitungsmodule hinzufügen möchten, muss der Docker-Host 4 vCPUs und 4 GB Speicher enthalten.

- Auf dem Docker-Hostsystem wird Linux Ubuntu Version 14.04 oder höher ausgeführt.
- Docker Version 1.12 ist auf dem Hostsystem installiert. Informationen zur Docker-Installation unter Linux finden Sie in derDocker-Dokumentation.
- Docker-Host hat Internetkonnektivität.

  **Hinweis**: NetScaler CPX hat Probleme beim Ausführen auf Ubuntu Version 16.04.5, Kernelversion 4.4.0-131-generic. Es wird daher nicht empfohlen, NetScaler CPX auf Ubuntu Version 16.04.5, Kernelversion 4.4.0-131-generic auszuführen.

  **Hinweis**: Die folgenden Versionen von Kubelet und Kube-Proxy weisen einige Sicherheitslücken auf. Es wird nicht empfohlen, Citrix NetScaler CPX mit diesen Versionen zu verwenden:

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

  Informationen zur Minderung dieser Sicherheitsanfälligkeit finden Sie unter Mindern dieses Sicherheitsrisikos.

### NetScaler CPX-Image von Quay herunterladen

Sie können das NetScaler CPX-Image mit dem `docker pull` Befehl aus der Quay-Container-Registrierung herunterladen und in Ihrer Umgebung bereitstellen. Verwenden Sie den folgenden Befehl, um das NetScaler CPX-Image aus der Quay-Container-Registrierung herunterzuladen:

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

Wenn Sie beispielsweise die Version 13.0-64.35 herunterladen möchten, verwenden Sie den folgenden Befehl:

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

Verwenden Sie den folgenden Befehl, um zu überprüfen, ob das NetScaler CPX-Image in Docker-Images installiert ist:

```
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
```

Sie können das neueste NetScaler CPX-Image aus der Quay-Container-Registrierungbereitstellen.

### Bereitstellen der NetScaler CPX-Instanz mithilfe des Docker-Run-Befehls

Auf dem Host können Sie eine NetScaler CPX-Instanz im Docker-Container installieren, indem Sie das NetScaler CPX Docker-Image verwenden, das Sie auf den Host geladen haben. Installieren Sie mit dem Befehl `docker run` die NetScaler CPX-Instanz mit der NetScaler CPX-Standardkonfiguration.

> **Wichtig:**
>
> Wenn Sie NetScaler CPX Express von CPX Expressheruntergeladen haben, lesen und verstehen Sie die Endbenutzer-Lizenzvereinbarung (EULA), verfügbar unter:CPX Express, und akzeptieren Sie die Endbenutzer-Lizenzvereinbarung, während Sie die NetScaler CPX-Instanz bereitstellen.

Installieren Sie die NetScaler CPX-Instanz im Docker-Container, indem Sie den folgenden ***Docker-Run-Befehl*** verwenden:

```
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='{
```

```
2   "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-->
```

In diesem Beispiel wird ein Container erstellt, der auf dem NetScaler CPX Docker-Image`mycpx` basiert.

Der`-P`Parameter ist zwingend erforderlich. Docker soll die Ports zuordnen, die vom NetScaler CPX Docker-Image im Container verfügbar sind. Das bedeutet, dass die Ports 9080, 22, 9443 und 161/UDP den Ports auf dem Docker-Host zugeordnet werden, die zufällig aus dem benutzerdefinierten Bereich ausgewählt werden. Dieses Mapping wird durchgeführt, um Konflikte zu vermeiden. Wenn Sie später mehrere NetScaler CPX-Container auf demselben Docker-Host erstellen. Die Portzuordnungen sind dynamisch und werden bei jedem Start oder Neustart des Containers festgelegt. Die Ports werden wie folgt verwendet:

- 9080 wird für HTTP verwendet
- 9443 wird für HTTPs verwendet
- 22 für SSH verwendet
- 161/UDP wird für SNMP verwendet.

Wenn Sie statische Portzuordnungen wünschen, verwenden Sie den Parameter -p, um sie manuell festzulegen.

Die `--privileged=true` Option wird verwendet, um den Container im privilegierten Modus auszuführen. Wenn Sie NetScaler CPX im Host-Bereitstellungsmodus ausführen, müssen Sie dem NetScaler CPX alle Systemberechtigungen bereitstellen. Wenn Sie den NetScaler CPX im Bridge-Modus mit einem oder mehreren Kernen ausführen möchten, können Sie anstelle dieser Option die `--cap-add=NET_ADMIN` Option verwenden. Mit `--cap-add=NET_ADMIN` dieser Option können Sie den NetScaler CPX-Container mit vollen Netzwerkberechtigungen ausführen.

Der`**--net=host` ist eine standardmäßige Docker-Ausführungsbefehlsoption, die angibt, dass der Container im Host-Netzwerkstapel ausgeführt wird und Zugriff auf alle Netzwerkgeräte hat.

> **Hinweis**
>
> Ignorieren Sie diese Option, wenn Sie NetScaler CPX in einem Bridge- oder keinem Netzwerk ausführen.

Das `-e NS_NETMODE="HOST"` ist eine NetScaler CPX-spezifische Umgebungsvariable, mit der Sie angeben können, dass der NetScaler CPX im Hostmodus gestartet wird. Sobald NetScaler CPX im Hostmodus gestartet wurde, konfiguriert es 4 Standard-iptables-Regeln auf einem Hostcomputer für den Verwaltungszugriff auf den NetScaler CPX. Es verwendet die folgenden Ports:

- 9995 für HTTP
- 9996 für HTTPS

- 9997 für SSH
- 9998 für SNMP

Wenn Sie verschiedene Ports angeben möchten, können Sie die folgenden Umgebungsvariablen verwenden:

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

> **Hinweis**
>
> Ignorieren Sie diese Umgebungsvariable, wenn Sie NetScaler CPX in einem Bridge- oder keinem Netzwerk ausführen.

Der`-e CPX_CORES`ist eine optionale NetScaler CPX-spezifische Umgebungsvariable. Sie können damit die Leistung der NetScaler CPX-Instanz verbessern, indem Sie die Anzahl der Verarbeitungsmodule definieren, die der NetScaler CPX-Container starten soll.

**Hinweis**: NetScaler CPX kann 1 bis 16 Kerne unterstützen.

> **Hinweis**
>
> Stellen Sie für jede zusätzliche Verarbeitungsengine, die Sie hinzufügen, sicher, dass der Docker-Host die entsprechende Anzahl von vCPUs und die Menge an Arbeitsspeicher in GB enthält. Wenn Sie beispielsweise 4 Verarbeitungsmodule hinzufügen möchten, muss der Docker-Host 4 vCPUs und 4 GB Speicher enthalten.

Die`-e EULA = yes`ist eine obligatorische NetScaler CPX-spezifische Umgebungsvariable, die erforderlich ist, um zu überprüfen, ob Sie die Endbenutzer-Lizenzvereinbarung (EULA) gelesen und verstanden haben, verfügbar unter:CPX Express.

Der`-e PLATFORM=CP1000`Parameter gibt den NetScaler CPX-Lizenztyp an.

Wenn Sie Docker in einem Host-Netzwerk ausführen, können Sie dem NetScaler CPX-Container mithilfe der `-e CPX_NW_DEV` Umgebungsvariablen dedizierte Netzwerkschnittstellen zuweisen. Sie müssen die Netzwerkschnittstellen durch ein Leerzeichen getrennt definieren. Die von Ihnen definierten Netzwerkschnittstellen werden vom NetScaler CPX-Container gespeichert, bis Sie den NetScaler CPX-Container deinstallieren. Wenn der NetScaler CPX-Container bereitgestellt wird, werden alle zugewiesenen Netzwerkschnittstellen dem NetScaler-Netzwerknamespace hinzugefügt.

> **Hinweis**
>
> Wenn Sie NetScaler CPX im Bridge-Netzwerk ausführen, können Sie das Container-Netzwerk ändern, z. B. eine andere Netzwerkverbindung zum Container konfigurieren oder ein vorhandenes

> Netzwerk entfernen. Stellen Sie dann sicher, dass Sie den NetScaler CPX-Container neu starten, um das aktualisierte Netzwerk zu verwenden.

```
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
<!--NeedCopy-->
```

Das `-e CPX_CONFIG` ist eine NetScaler CPX-spezifische Umgebungsvariable, mit der Sie die Durchsatzleistung des NetScaler CPX-Containers steuern können. Wenn das NetScaler CPX keinen eingehenden Datenverkehr zur Verarbeitung empfängt, liefert es die CPU während dieser Leerlaufzeit, was zu einer geringen Durchsatzleistung führt. In solchen Szenarien können Sie die `CPX_CONFIG` Umgebungsvariable verwenden, um die Durchsatzleistung des NetScaler CPX-Containers zu steuern. Sie müssen der `CPX_CONFIG` Umgebungsvariablen folgende Werte im JSON-Format angeben:

- Wenn Sie möchten, dass der NetScaler CPX-Container in Leerlaufszenarien CPU liefert, definieren Sie `{ "YIELD" : "Yes" }`
- Wenn Sie möchten, dass der NetScaler CPX-Container die CPU in Leerlaufszenarien nicht nachgibt, sodass Sie eine hohe Durchsatzleistung erzielen können, definieren Sie `{ "YIELD" : "No" }`

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

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

Der `-v`Parameter ist ein optionaler Parameter, der den Bereitstellungspunkt des NetScaler CPX-Mount-Verzeichnisses angibt `/cpx`. Ein Einhängepunkt ist ein Verzeichnis auf dem Host, in dem Sie das `/cpx`Verzeichnis mounten. Das `/cpx`Verzeichnis speichert die Protokolle, Konfigurationsdateien, SSL-Zertifikate und Core-Dump-Dateien. Im Beispiel ist der Bereitstellungspunkt `/var/cpx` und das NetScaler CPX-Mount-Verzeichnis ist `/cpx`.

Wenn Sie eine Lizenz erworben haben oder über eine Evaluierungslizenz verfügen, können Sie die Lizenz auf einen Lizenzserver hochladen und den Speicherort des Lizenzservers mit dem Befehl Docker Run angeben, indem Sie den`-e LS_IP=<LS_IP_ADDRESS> -e LS_PORT=<LS_PORT >`Parameter verwenden. In diesem Fall müssen Sie die Endbenutzer-Lizenzvereinbarung nicht akzeptieren.

```
docker run -dt --privileged=true --net=host -e NS_NETMODE="HOST" -e
    CPX_CORES=5 -e CPX_CONFIG='{
```

```
 "YIELD":"No" }
 ' -e LS_IP=10.102.38.134 -e PLATFORM=CP1000 --name cpx_host cpx:13.0-x
    .x
<!--NeedCopy-->
```

Dabei gilt:

- `LS_IP_ADDRESS` ist die IP-Adresse des Lizenzservers.
- `LS_PORT` ist der Port des Lizenzservers.

Sie können die Images, die auf Ihrem System ausgeführt werden, und die Ports, die den Standardports zugeordnet sind, mithilfe des folgenden Befehls anzeigen: `docker ps`

### Bereitstellen einer leichteren Version von NetScaler CPX mithilfe des Docker-Befehls run

NetScaler bietet eine leichtere Version von NetScaler CPX, die weniger Laufzeitspeicher verbraucht. Die leichtere Version von NetScaler CPX kann als Beiwagen in Service-Mesh-Bereitstellungen bereitgestellt werden.

Die leichtere Version von NetScaler CPX unterstützt die folgenden Funktionen:

- Verfügbarkeit der Anwendung
  - L4-Loadbalancing und L7-Content Switching
  - SSL Offloading
  - IPv6-Protokollübersetzung
- Anwendungssicherheit
  - L7 Rewrite und Responder
- Einfache Verwaltbarkeit
  - Webprotokollierung
  - AppFlow

Um die leichtere Version von NetScaler CPX zu instanziieren, legen Sie die `NS_CPX_LITE` Umgebungsvariable fest, während Sie den `Docker run` Befehl ausführen.

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

Im folgenden Beispiel wird ein Lightweight-Container basierend auf dem NetScaler CPX-Image erstellt.

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

Standardmäßig `newnslog` ist die Protokollierung mit in der leichteren Version von NetScaler CPX deaktiviert. Um es zu aktivieren, müssen Sie die `NS_ENABLE_NEWNSLOG` Umgebungsvariable auf 1 setzen, während Sie die leichtere Version von NetScaler CPX aufrufen.

Das folgende Beispiel zeigt, wie Sie die `newnslog` Protokollierung mithilfe der leichteren Version von NetScaler CPX aktivieren.

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

> **Hinweis:** Die leichtere Version von CPX unterstützt nur Single-Core (CPX_CORES=1).

### Bereitstellen von NetScaler CPX-Instanzen mithilfe von Docker Compose

Sie können das Compose-Tool von Docker verwenden, um eine einzelne NetScaler CPX-Instanz oder mehrere NetScaler CPX-Instanzen bereitzustellen. Um NetScaler CPX-Instanzen mithilfe von Docker Compose bereitzustellen, müssen Sie zuerst eine Compose-Datei schreiben. Diese Datei gibt das NetScaler CPX-Image, die Ports an, die Sie für die NetScaler CPX-Instanz öffnen möchten, und die Berechtigungen für Ihre NetScaler CPX-Instanz.

> **Wichtig**
>
> Stellen Sie sicher, dass Sie das Docker Compose-Tool auf dem Host installiert haben.

**So stellen Sie mehrere NetScaler CPX-Instanzen bereit:**

1. Schreiben Sie eine Compose-Datei, wobei:

- **<service-name>** ist der Name des Dienstes, den Sie bereitstellen möchten.
- **image::** <repository><tag>bezeichnet das Repository und die Versionen des NetScaler CPX-Images.
- **privileged: true** bietet alle Rootrechte für die NetScaler CPX-Instanz.
- **cap_add** bietet Netzwerkberechtigungen für die NetScaler CPX-Instanz.
- ** <host_directory_path>bezeichnet das Verzeichnis auf dem Docker-Host, das Sie für die NetScaler CPX-Instanz mounten möchten.
- **<number_processing_engine>ist die Anzahl der Verarbeitungsmodule, die die NetScaler CPX-Instanz starten soll. Stellen Sie für jede zusätzliche Verarbeitungsengine sicher, dass der Docker-Host die entsprechende Anzahl von vCPUs und die Menge an Arbeitsspeicher in GB enthält.

Wenn Sie beispielsweise 4 Verarbeitungsmodule hinzufügen möchten, muss der Docker-Host 4 vCPUs und 4 GB Speicher enthalten.

Die Compose-Datei folgt im Allgemeinen einem Format ähnlich:

```
    <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-->
```

# NetScaler CPX-Instanzen zu NetScaler ADM hinzufügen

March 21, 2024

Sie müssen die auf einem Docker-Host installierten NetScaler CPX-Instanzen zur NetScaler Application Delivery Management (ADM)-Software hinzufügen, wenn Sie diese Instanzen verwalten und überwachen möchten.

Sie können Instanzen hinzufügen, während Sie ADM zum ersten Mal oder später einrichten.

Um Instanzen hinzuzufügen, müssen Sie ein Instanzprofil erstellen und entweder den Hostnamen oder die IP-Adresse jeder Instanz oder einen Bereich von IP-Adressen angeben. Dieses Instanzprofil enthält den Benutzernamen und das Kennwort der Instanzen, die Sie zu NetScaler ADM hinzufügen möchten. Für jeden Instanztyp ist ein Standardprofil verfügbar. Zum Beispiel `ns-root-profile` ist das Standardprofil für NetScaler-Instanzen. Dieses Profil wird durch die Standard-ADC-Administratoranmeldeinformationen definiert. Wenn Sie die standardmäßigen Administratoranmeldeinformationen Ihrer Instanzen geändert haben, können Sie benutzerdefinierte Instanzprofile für diese Instanzen definieren. Wenn Sie die Anmeldeinformationen einer Instanz ändern, nachdem die Instanz erkannt wurde, müssen Sie das Instanzprofil bearbeiten oder ein Profil erstellen und dann die Instanz neu ermitteln.

## Voraussetzungen

Stellen Sie sicher, dass Sie:

- Die NetScaler ADM-Software wurde auf Citrix XenServer installiert. Weitere Informationen finden Sie in der NetScaler ADM-Dokumentation.
- Die NetScaler CPX-Instanzen wurden auf einem Docker-Host installiert.

**So fügen Sie NetScaler CPX-Instanzen zu ADM hinzu:**

1. Geben Sie in einem Webbrowser die IP-Adresse von **NetScaler Application Delivery Management** ein (z. B. `http://192.168.100.1)`).
2. Geben Sie in den Feldern **Benutzername** und **Kennwort** die Administratoranmeldeinformationen ein. Die standardmäßigen Administratoranmeldeinformationen sind **nsroot** und **nsroot**.
3. Navigieren Sie zu **Netzwerke > Instanzen** > **NetScaler** und klicken Sie auf die Registerkarte **CPX**.
4. Klicken Sie auf **Hinzufügen**, um neue CPX-Instanzen in NetScaler ADM hinzuzufügen.
5. Die Seite **NetScaler CPX** hinzufügen wird geöffnet. Geben Sie die Werte für die folgenden Parameter ein:

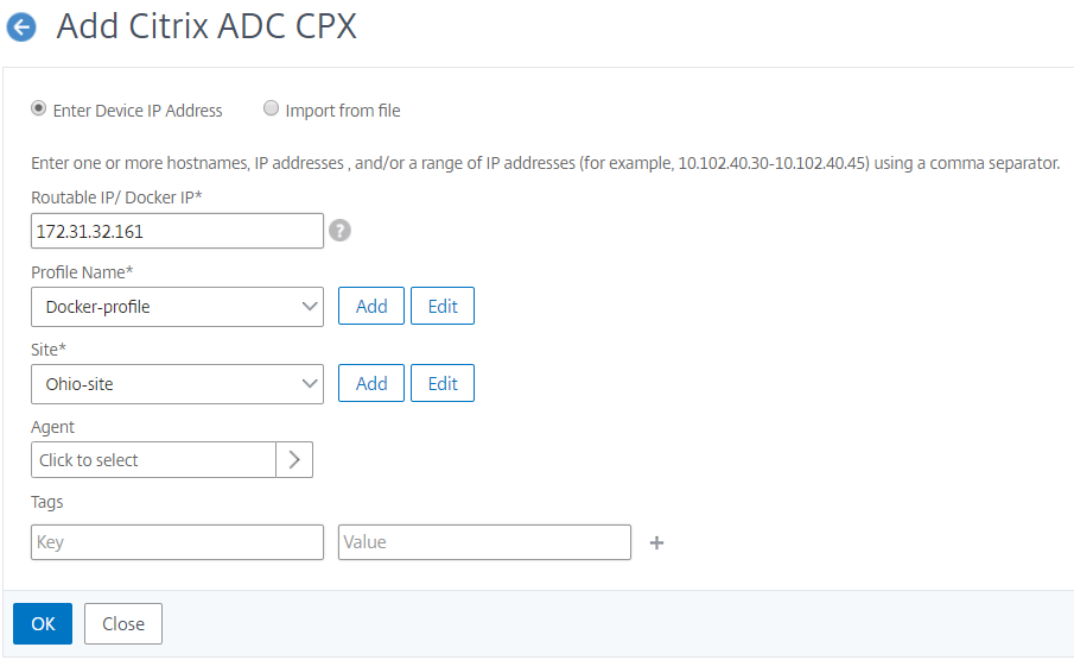
a) Sie können CPX-Instanzen hinzufügen, indem Sie entweder die erreichbare IP-Adresse der CPX-Instanz oder die IP-Adresse des Docker-Containers angeben, in dem die CPX-Instanz gehostet wird.

b) Wählen Sie das Profil der CPX-Instanz aus.

c) Wählen Sie den Standort aus, an dem die Instanzen bereitgestellt werden sollen.

d) Wählen Sie den Agenten aus.

e) Optional können Sie das Schlüssel-Wert-Paar für die Instanz eingeben. Das Hinzufügen eines Schlüssel-Wert-Paares erleichtert Ihnen die spätere Suche nach der Instanz.




6. Klicken Sie auf **OK**.

> **Hinweis**
>
> Wenn Sie eine Instanz erneut ermitteln möchten, wählen Sie **Netzwerke > Instanzen > NetScaler > CPX**aus, wählen Sie die Instanz aus, die Sie erneut ermitteln möchten, und klicken Sie dann in der Dropdownliste **Aktion auswählen** auf **Erneut ermitteln**.

## Hinzufügen von NetScaler CPX-Instanzen zu NetScaler ADM mithilfe von Umgebungsvariablen

Sie können die NetScaler CPX-Instanzen auch mithilfe von Umgebungsvariablen zu NetScaler ADM hinzufügen. Um Instanzen hinzuzufügen, müssen Sie die folgenden Umgebungsvariablen für die NetScaler CPX-Instanz konfigurieren.

- NS_MGMT_SERVER - ADM-IP-Adresse/FQDN
- HOST - Node-IP-Adresse
- NS_HTTP_PORT - Zugeordneter HTTP-Port auf Knoten
- NS_HTTPS_PORT- Zugeordneter HTTPS-Port auf Knoten
- NS_SSH_PORT - Zugeordneter SSH-Port auf Knoten
- NS_SNMP_PORT - Zugeordneter SNMP-Port auf Knoten
- NS_ROUTABLE - (Die NetScaler CPX-Pod-IP-Adresse kann nicht von außen weitergeleitet werden.)
- NS_MGMT_USER —ADM-Benutzername
- NS_MGMT_PASS —ADM-Kennwort

Im Folgenden finden Sie einen `docker run` Beispielbefehl zum Hinzufügen einer NetScaler CPX-Instanz zu 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-->
```

### Hinzufügen von NetScaler CPX-Instanzen zu NetScaler ADM mithilfe von Kubernetes ConfigMaps

NetScaler CPX unterstützt die Registrierung bei NetScaler ADM, indem Volume-Mount-Dateien über Kubernetes ConfigMaps verwendet werden. Um diese Art der Registrierung zu ermöglichen, benötigt NetScaler CPX einige Umgebungsvariablen, die zusammen mit einigen vom Volume bereitgestellten Dateien über ConfigMaps und Secrets angegeben werden müssen.

Im Folgenden sind die erforderlichen Umgebungsvariablen und ihre Beschreibung aufgeführt:

- NS_HTTP_PORT - Gibt den zugeordneten HTTP-Port auf dem Knoten an.
- NS_HTTPS_PORT - Gibt den zugeordneten HTTPS-Port auf dem Knoten an.
- NS_SSH_PORT - Gibt den zugeordneten SSH-Port auf dem Knoten an.
- NS_SNMP_PORT- Gibt den zugeordneten SNMP-Port auf dem Knoten an.

Abgesehen von den aufgelisteten Umgebungsvariablen benötigt NetScaler CPX Informationen über den ADM-Agenten, bei dem es sich registrieren muss. Diese Informationen enthalten die IP-Adresse des ADM-Agenten oder FQDN-Details und Anmeldeinformationen. NetScaler CPX bezieht diese Informationen aus den vom Volume bereitgestellten Dateien. Eine ConfigMap, die die IP-Adresse oder den FQDN enthält, wird als Datei im Dateisystem der NetScaler CPX-Instanz eingehängt. Ein Kubernetes-Geheimnis, das Anmeldeinformationen für den ADM-Agenten enthält, wird auch als Datei im Dateisys-

tem der NetScaler CPX-Instanz eingehängt. Mit allen für die Registrierung erforderlichen Informationen versucht NetScaler CPX, sich beim ADM-Agenten zu registrieren.

Im Folgenden finden Sie ein Beispiel für ein NetScaler CPX YAML-Datei-Snippet, bei dem ConfigMap und Secret als Dateien eingehängt sind:

```
        ...
        env:
        - name: "EULA"
          value: "yes"
        - name: "NS_HTTP_PORT"
          value: "9080"
        - name: "NS_HTTPS_PORT"
          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
```

Im vorherigen Beispiel werden eine ConfigMap mit dem Namen `adm-agent-config` und ein Secret `adm-agent-user` verbraucht. Im Folgenden finden Sie ein Beispiel für das Erstellen der erforderlichen ConfigMap und Secret.

**ConfigMap:** Die ConfigMap wird aus einer Datei namens erstellt `adm_reg_envs`. Die Datei benötigt die IP-Adresse oder den FQDN des ADM-Agenten im folgenden Format:

```
NS_MGMT_SERVER=adm-agent
```

Im vorherigen Format `adm-agent` ist der der FQDN des ADM-Agenten, für den die NetScaler CPX-Instanz registriert werden muss.

Verwenden Sie den folgenden Befehl, um eine ConfigMap zu erstellen:

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

**Hinweis:** Der Dateiname muss die `adm_reg_envs` Variable haben und sie muss in den Pfad eingehängt werden: `/var/adminfo/server/`.

**Secret:** Verwenden Sie den folgenden Befehl, um einen Kubernetes-Schlüssel zu erstellen. Im folgenden Befehl `user123` ist der Benutzername des ADM-Agenten und `pass123` das Kennwort.

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

Eine NetScaler CPX-Instanz kann in einem Kubernetes-Cluster mit den erforderlichen Umgebungsvariablen und Volume-bereitgestellten Dateien bereitgestellt werden, noch bevor der ADM-Agent im Cluster bereitgestellt wird. Wenn Sie vor der Bereitstellung des ADM-Agenten eine NetScaler CPX-Instanz bereitstellen, versucht NetScaler CPX weiterhin, sich registrieren zu lassen, bis der ADM-Agent bereitgestellt ist. Sobald der ADM-Agent bereitgestellt wurde, verwendet die NetScaler CPX-Instanz die Konfigurationsdaten, die über die Umgebungsvariablen und vom Volume bereitgestellten Dateien bereitgestellt werden, um sich beim ADM-Agent zu registrieren. Es hilft Ihnen, die erneute Bereitstellung von NetScaler CPX mit den Konfigurationsinformationen zu vermeiden.

Eine NetScaler CPX-Instanz, die bereits bei einem ADM-Agenten registriert ist, kann die Registrierung nach einer Änderung der Konfiguration dynamisch zu einem anderen ADM-Agenten ändern. Dazu können Sie die Konfigurationsinformationen in der ConfigMap und im Secret für das bereits bereitgestellte NetScaler CPX aktualisieren. Sie müssen die Datei, aus der die ConfigMap erstellt wurde, mit der IP-Adresse oder dem FQDN des neuen ADM-Agenten aktualisieren und die alte ConfigMap löschen und dann eine neue ConfigMap erstellen. In ähnlicher Weise muss das vorhandene Geheimnis gelöscht werden und ein neuer Schlüssel muss mit den Anmeldeinformationen für den neuen ADM-Agenten erstellt werden.

## NetScaler CPX License Aggregator

November 23, 2023

Derzeit beziehen NetScaler CPXs Lizenzen vom NetScaler ADM-Server. In einer Kubernetes-Umgebung können NetScaler CPXs dynamisch hoch- oder herunterfahren. Wenn ein NetScaler CPX unerwartet ausfällt, dauert es einige Minuten, bis der NetScaler ADM-Server die Lizenz zurückfordert. Der NetScaler ADM-Server muss in der Lage sein, solche Lizenzen sofort zurückzufordern, wenn NetScaler CPXs ausfallen, damit dieselbe Lizenz einem anderen bevorstehenden NetScaler CPX zugewiesen werden kann. Wenn der NetScaler ADM-Server aus irgendeinem Grund nicht erreichbar ist, können Sie keine neuen NetScaler CPXs im Cluster lizenzieren.

NetScaler CPX License Aggregator ist ein Kubernetes-Dienst, der von NetScaler bereitgestellt wird. Dieser Dienst fungiert als lokaler Anbieter für die NetScaler CPX-Lizenzierung innerhalb eines

Kubernetes-Clusters. Der in einem Kubernetes-Cluster bereitgestellte NetScaler CPX License Aggregator Service kann als Vermittler zwischen NetScaler CPXs und dem ADM-Lizenzserver fungieren und NetScaler CPXs und die zugewiesenen Lizenzen verfolgen. Mit dem NetScaler CPX License Aggregator Service kann der NetScaler ADM-Server Lizenzen sofort zurückfordern, wenn NetScaler CPXs ausfallen.

In einem Kubernetes-Cluster unterstützt der NetScaler CPX License Aggregator Service sowohl NetScaler CPX als Sidecar- als auch eigenständige Bereitstellungen.

> **Hinweis:**
>
> Für die Lizenzierung mit dem NetScaler CPX License Aggregator ist NetScaler CPX 13.1-30.x oder höher erforderlich. NetScaler CPX License Aggregator unterstützt die Lizenzierung älterer Versionen von NetScaler CPX nicht.

## Hauptvorteile von NetScaler CPX License Aggregator

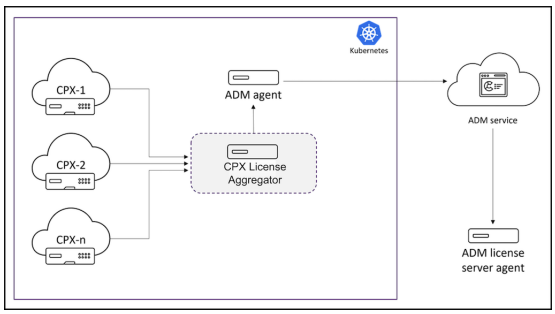
Im Folgenden sind die wichtigsten Vorteile der Verwendung von NetScaler CPX License Aggregator aufgeführt:

- Skalierbarkeit:
  Ein NetScaler ADM-Lizenzserver kann nur bis zu 10000 NetScaler CPX-Bereitstellungen unterstützen. Mit der Einführung des NetScaler CPX License Aggregator Services kann jeder Kubernetes-Cluster als einzelner Client für den NetScaler ADM-Lizenzserver fungieren. Daher können Sie viele NetScaler CPXs mit einem einzigen NetScaler ADM-Lizenzserver skalieren.
- Ressourcenoptimierung:
  Der NetScaler CPX License Aggregator Service unterstützt auch clusterweite Lizenzierungsfunktionen und kann bei Bedarf auch Lizenzen vom NetScaler ADM-Server auschecken. NetScaler CPX License Aggregator kann Lizenzen an den NetScaler ADM-Server zurückgeben.
  NetScaler CPX License Aggregator kann die ungerade Beendigung von NetScaler CPXs verarbeiten und Lizenzen von solchen NetScaler CPXs nach Ablauf der konfigurierten Wartezeit zurückfordern.

## Topologie der Bereitstellung

Das folgende Diagramm zeigt eine NetScaler CPX License Aggregator-Bereitstellung in einem Kubernetes-Cluster.




In diesem Diagramm:

- `CPX` bedeutet NetScaler CPX
- `CPX License Aggregator` bedeutet NetScaler CPX License Aggregator

In dieser Beispielbereitstellung wird ein NetScaler CPX License Aggregator Service zusammen mit NetScaler CPXs und NetScaler ADM Agent im Kubernetes-Cluster bereitgestellt. Der NetScaler CPX License Aggregator Service fungiert als Vermittler zwischen NetScaler CPXs und NetScaler ADM Agent und überwacht alle NetScaler CPXs innerhalb des Clusters und verwaltet Lizenzen für diese.

## Bereitstellen von NetScaler CPX License Aggregator mit Helm-Charts

### Voraussetzungen

Es gelten die folgenden Voraussetzungen:

- Sie benötigen Kubernetes Version 1.16 und höher.
- Sie benötigen die Helm-Version 3.x oder höher.
- Sie müssen die IP-Adresse des Lizenzservers abrufen, der über die Lizenz für NetScaler CPX verfügt.
- Sie müssen ein Kennwort für die Redis-Datenbank in NetScaler CPX License Aggregator angeben. Sie können das Datenbankkennwort über Kubernetes-Geheimnisses angeben und der folgende Befehl kann verwendet werden, um das Geheimnis zu erstellen:

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

### Bereitstellung über Helm Charts

Führen Sie die folgenden Schritte aus, um den NetScaler CPX License Aggregator über Helm Charts je nach Typ der NetScaler CPX-Lizenz bereitzustellen. Weitere Informationen zu den verschiedenen Arten von NetScaler CPX-Lizenzen finden Sie unter NetScaler CPX-Lizenzierung.

**Helm Charts installieren** Fügen Sie das NetScaler CPX License Aggregator Helm Chart-Repository mit dem folgenden Befehl hinzu:

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

**Installieren von NetScaler CPX License Aggregator zur Verwaltung von Bandbreiten-Poollizenzen** Verwenden Sie je nach Typ der gepoolten NetScaler CPX-Lizenz einen der folgenden Befehle. In diesen Befehlen wird `my-release` als Releasename verwendet.

> **Hinweis:**
>
> Standardmäßig installiert das Helm-Diagramm die empfohlenen RBAC-Rollen und Rollenbindungen.

Für Platin-Bandbreiten-Lizenz

```
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
   >
```

Für Enterprise Bandwidth Edition:

```
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>
```

Für die Standard-Bandbreiten-Edition:

```
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
   >
```

Mit diesen Befehlen wird der NetScaler CPX License Aggregator auf dem Kubernetes-Cluster mit der Standardkonfiguration bereitgestellt. Sie können die Parameter bei der Installation konfigurieren.

Weitere Informationen finden Sie im Abschnitt **Konfiguration des NetScaler CPX License Aggregator** im Helm Charts GitHub-Repository, in dem die obligatorischen und optionalen Parameter aufgeführt sind, die Sie während der Installation konfigurieren können.

### Installieren von NetScaler CPX License Aggregator zum Verwalten von vCPU-Lizenzen

Verwenden Sie je nach Typ der NetScaler CPX vCPU-Lizenz einen der folgenden Befehle. In diesen Befehlen wird `my-release` als Releasename verwendet.

**Hinweis:**
Das Helm-Diagramm installiert standardmäßig die empfohlenen RBAC-Rollen und Rollenbindungen.

Für Platin-vCPU-Edition:

```
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>
```

Für die Enterprise vCPU Edition:

```
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
   >
```

Für die Standard-vCPU-Edition:

```
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>
```

### Installieren von NetScaler CPX License Aggregator zur Verwaltung mehrerer Lizenzen

Wenn Sie den NetScaler CPX License Aggregator benötigen, um mehrere Lizenztypen zu verwalten, müssen die relevanten Argumente dieser Lizenzen im Helm-Befehl angegeben werden.

Beispiel:

So stellen Sie NetScaler CPX License Aggregator für `pooled platinum bandwidth edition`- und `vCPU platinum edition`-Lizenzen bereit:

```
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>
```

**Konfigurieren von NetScaler CPX, um eine Lizenz von NetScaler CPX License Aggregator zu erhalten**

Wenn Sie NetScaler CPX License Aggregator für die Lizenzierung von NetScaler CPX verwenden, muss die Umgebungsvariable `CLA` in der NetScaler CPX-Bereitstellungs-YAML bereitgestellt werden.

`ipaddress` oder `domainname`, mit denen auf den NetScaler CPX License Aggregator zugegriffen werden kann, muss in dieser Umgebungsvariablen wie folgt bereitgestellt werden:

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

oder

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

Sie müssen auch die folgenden Umgebungsvariablen in der NetScaler CPX YAML angeben.

- `POD_NAME`: Gibt den Namen des Pods an. Der Name des Pods wird in NetScaler CPX als Umgebungsvariable verfügbar gemacht.
- `POD_NAMESPACE`: Gibt den Namespace des Pods an. Der Namespace des Pods wird NetScaler CPX als Umgebungsvariable zur Verfügung gestellt.
- `Bandwidth`: Gibt die Bandbreite in Mbit/s für die Zuweisung an NetScaler CPX an.
- `Edition`: Gibt die Lizenzversion an. Zu den unterstützten Werten gehören Standard, Platinum und Enterprise.
- `CPX_CORES` : Gibt die Anzahl der Kerne an, die Sie für NetScaler CPX ausführen möchten.

Weitere Informationen zu verschiedenen NetScaler CPX-Lizenzierungsoptionen finden Sie unter NetScaler CPX-Lizenzierung.

Das Folgende zeigt eine Beispielkonfiguration mit diesen Umgebungsvariablen:

```
- 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
```

Sie müssen außerdem das folgende Label zur NetScaler CPX YAML hinzufügen:

```
      labels:
        adc: citrix
```

Ein Beispiel für die Bereitstellung von NetScaler CPX License Aggregator finden Sie unter NetScaler CPX License Aggregator: Beispielbereitstellung.

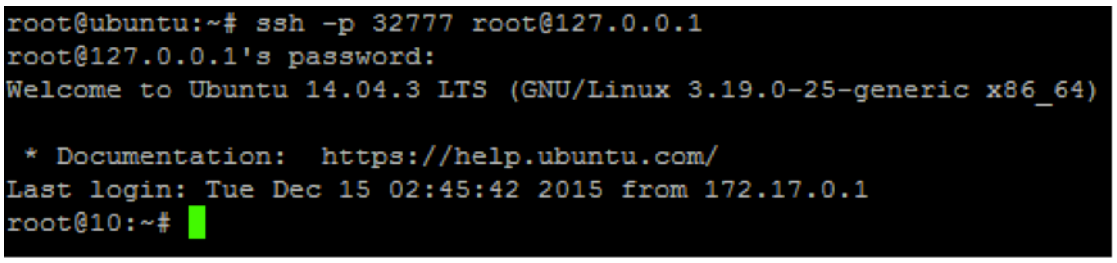
## Konfigurieren von NetScaler CPX

November 23, 2023

Sie können eine NetScaler CPX-Instanz konfigurieren, indem Sie über den Linux-Docker-Host oder mithilfe der NetScaler NITRO APIs auf die CLI-Eingabeaufforderung zugreifen.

### Konfigurieren einer NetScaler CPX-Instanz mithilfe der Befehlszeilenschnittstelle

Greifen Sie auf den Docker-Host zu und melden Sie sich an der SSH-Eingabeaufforderung der Instanz an, wie in der folgenden Abbildung dargestellt. Die standardmäßigen Administratoranmeldeinformationen für die Anmeldung bei einer NetScaler CPX-Instanz sind Root/Linux.




Geben Sie den folgenden Befehl ein, um die Befehlszeilenaufforderung der Instanz zum Ausführen von CLI-Befehlen zu verwenden: **cli_script.sh** "<command>"

**Beispiel**:

```
root@10:~# cli_script.sh "show ip"
exec: show ip
        Ipaddress        Traffic Domain  Type
        ---------        --------------  ----
1)      172.17.0.4       0               NetScaler IP|VIP
2)      192.0.0.1        0               SNIP
```

Um sich von der Instanzaufforderung abzumelden, geben **Sie Abmelden**ein

#### Unterstützung für die Verwendung eines nicht standardmäßigen Kennworts in NetScaler CPX

NetScaler CPX unterstützt die Verwendung eines nicht standardmäßigen Kennworts für das Root-Konto `nsroot`. Ein Standardkennwort wird generiert und dem Benutzer zugewiesen, sobald NetScaler CPX bereitgestellt wurde. Dieses Standardkennwort wird auch für SSH-Benutzer aktualisiert: `root` und `nsroot`. Sie können dieses Standardkennwort manuell ändern. Sie können das standardmäßige SSH-Kennwort für `root` und `nsroot` Benutzerkonten auch manuell zurücksetzen. Citrix empfiehlt, dieses Kennwort manuell zu ändern, um die Sicherheit des Systems zu gewährleisten.

Sobald Sie Ihr Kennwort zurückgesetzt haben, wird das neue Kennwort für die Kommunikation und `cli_script.sh` Ausführung der NITRO API verwendet.
Das Standardkennwort für das Root-Konto wird im Klartext in der Datei */var/deviceinfo/random_id* im NetScaler CPX-Dateisystem gespeichert.

Verwenden Sie die folgende Syntax für die Ausführung `cli_script.sh` mit den Anmeldeinformationen:

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

Um beispielsweise IP-Adressen mit Benutzer `nsroot` und Kennwort anzuzeigen `Citrix123`, verwenden Sie Folgendes: `cli_script.sh`

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

### Konfigurieren einer NetScaler CPX-Instanz mithilfe der NITRO-API

Sie können die NetScaler NITRO API verwenden, um NetScaler CPX-Instanzen zu konfigurieren.

**Um NetScaler CPX-Instanzen mithilfe der Nitro-API zu konfigurieren, geben Sie in einem Webbrowser Folgendes**ein:

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

**Um Statistiken mithilfe der Nitro-API abzurufen, geben Sie in einem Webbrowser Folgendes**ein:

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

Weitere Informationen zur Verwendung der NITRO-API finden Sie unter REST Web Services. Verwenden Sie für NetScaler CPX, `CPX IP address:port` wo erwähnt `netscaler-ip-address` wird.

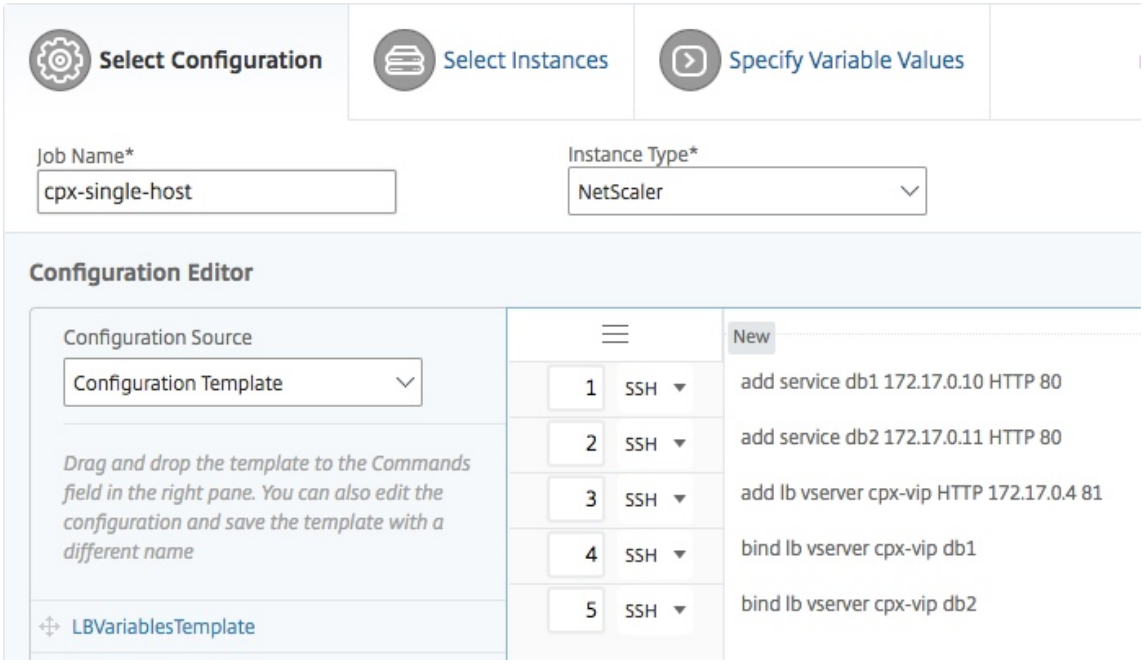
### Konfigurieren einer NetScaler CPX-Instanz mithilfe von Jobs

Sie können NetScaler CPX-Instanzen konfigurieren, indem Sie Jobs in NetScaler ADM erstellen und ausführen. Sie können die Konfigurationen aus Konfigurationsvorlagen verwenden, Konfigurationen extrahieren, die auf anderen Geräten verfügbar sind, und in Textdateien gespeicherte Konfigurationen verwenden. Sie können auch Konfigurationen aufzeichnen, die mit dem Konfigurationsdienstprogramm einer anderen Instanz durchgeführt wurden. NetScaler ADM zeigt dann die entsprechenden CLI-Befehle an, die Sie auf Ihrer NetScaler CPX-Instanz verwenden können. Nachdem Sie die Konfiguration ausgewählt haben, müssen Sie dann **NetScaler CPX-Instanzen** auswählen, auf die Sie die Konfiguration laden möchten, die Variablenwerte angeben und den Job ausführen.

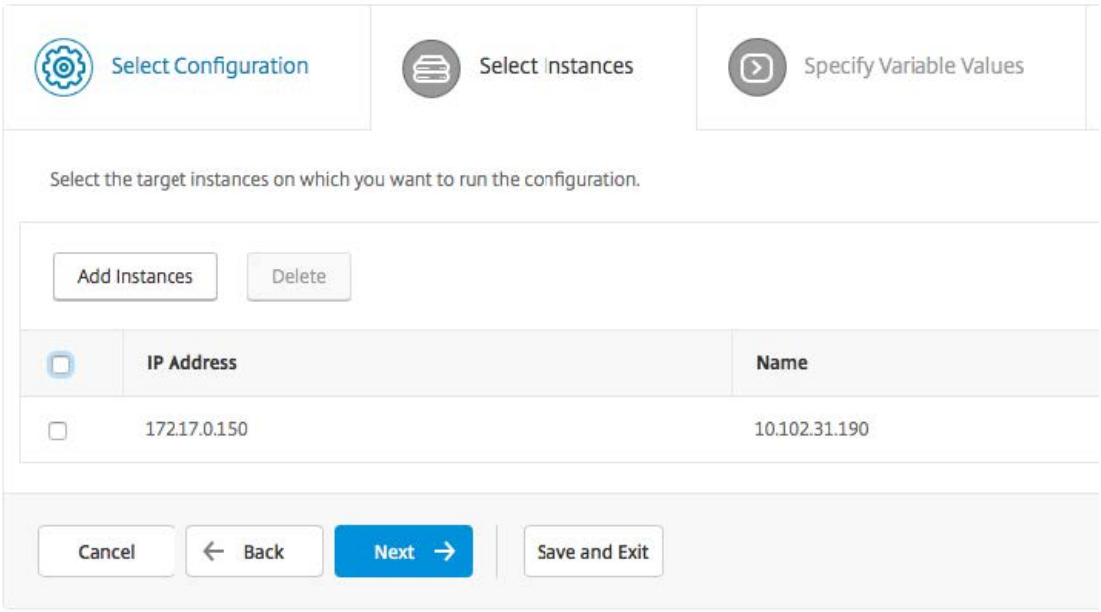
**Konfigurieren von NetScaler CPX-Instanzen mithilfe von Jobs:**

1. Melden Sie sich mit den Administratoranmeldeinformationen bei NetScaler ADM an.
2. Navigieren Sie zu **Networks > Configuration Jobs** und klicken Sie dann auf **Create Job**.
3. Geben Sie die erforderlichen Werte an und wählen Sie die Konfigurationsquelle aus. Sie können auch die Befehle eingeben, die Sie ausführen möchten.

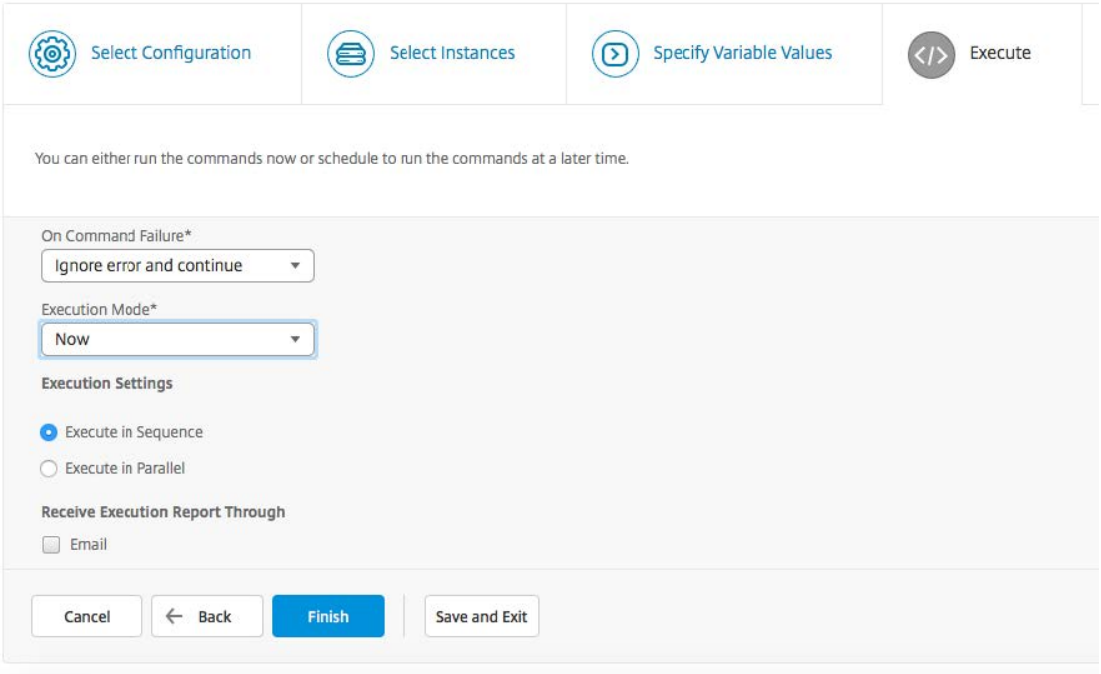



4. Wählen Sie die NetScaler CPX-Instanzen aus, auf denen Sie die Konfiguration ausführen möchten, und klicken Sie auf **Weiter**.




5. Geben Sie die Ausführungseinstellungen an und klicken Sie auf Fertig stellen, um die Befehle auf der NetScaler CPX-Instanz auszuführen. Wenn Sie die Konfiguration speichern und später ausführen möchten, klicken Sie auf **Speichern und Beenden**.




## Konfigurieren von AppFlow auf einer NetScaler CPX-Instanz

November 23, 2023

Sie können die AppFlow-Funktion auf einer NetScaler CPX-Instanz konfigurieren, um Leistungsdaten der Webseite, Informationen auf Fluss- und Benutzersitzungsebene sowie Datenbankinformationen zu erfassen, die für die Überwachung und Analyse der Anwendungsleistung erforderlich sind. Diese Datensätze werden an NetScaler ADM gesendet, wo Sie Echtzeit- und Verlaufsberichte für alle Ihre Anwendungen anzeigen können.

Um AppFlow zu konfigurieren, müssen Sie zunächst die AppFlow-Funktion aktivieren. Anschließend geben Sie die Collectors an, an die die Flow-Datensätze gesendet werden. Danach definieren Sie Aktionen, bei denen es sich um konfigurierte Collectors handelt. Anschließend konfigurieren Sie eine oder mehrere Richtlinien und ordnen jeder Richtlinie eine Aktion zu. Die Richtlinie weist den NetScaler CPX an, Anforderungen auszuwählen, deren Flow-Datensätze an die zugehörige Aktion gesendet werden. Schließlich binden Sie jede Richtlinie entweder global oder an den spezifischen virtuellen Server, um sie in Kraft zu setzen.

Sie können AppFlow-Parameter weiter festlegen, um das Aktualisierungsintervall der Vorlage festzulegen und den Export von `httpURLhttpCookie`, und `httpReferer` Informationen zu ermöglichen. Auf jedem Collector müssen Sie die NetScaler CPX-IP-Adresse als Adresse des Exportprogramms angeben.

Das Konfigurationsdienstprogramm bietet Tools, mit denen Benutzer die Richtlinien und Aktionen definieren können. Es bestimmt genau, wie der NetScaler CPX Datensätze für einen bestimmten Fluss in eine Reihe von Collectors exportiert (Aktion). Die Befehlszeilenschnittstelle bietet einen entsprechenden Satz von CLI-basierten Befehlen für erfahrene Benutzer, die eine Befehlszeile bevorzugen.

Bevor Sie die Datensätze überwachen können, müssen Sie die NetScaler CPX-Instanz zum NetScaler ADM hinzufügen. Weitere Informationen zum Hinzufügen einer NetScaler CPX-Instanz zu NetScaler ADM finden Sie unter Installieren einer NetScaler CPX-Instanz mithilfe von NetScaler ADM.

### AppFlow aktivieren

Um die AppFlow-Funktion verwenden zu können, müssen Sie sie zuerst aktivieren.

**So aktivieren Sie die AppFlow-Funktion mithilfe der Befehlszeilenschnittstelle:**

Führen Sie die folgenden Befehle aus:

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

### Einen Collector angeben

Ein Collector empfängt AppFlow-Datensätze, die vom NetScaler generiert wurden. Um die AppFlow-Datensätze zu senden, müssen Sie mindestens einen Collector angeben. Standardmäßig hört der Collector IPFIX-Nachrichten auf dem UDP-Port 4739 ab. Sie können den Standardanschluss ändern, wenn Sie den Collector konfigurieren.

**So geben Sie einen Collector mit der Befehlszeilenschnittstelle an:**

Verwenden Sie die folgenden Befehle, um einen Collector hinzuzufügen:

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

Um die Konfiguration zu überprüfen, verwenden Sie den folgenden Befehl:

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

**So geben Sie mehrere Collectors mit der Befehlszeilenschnittstelle an:**

Verwenden Sie die folgenden Befehle, um dieselben Daten hinzuzufügen und an mehrere Collectors zu senden:

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

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

```
4
5 add appflow action <action> -collectors <collector1> <collector2> -
     Transport Logstream
6
7 add appflow policy <policy> true <action> -Transport Logstream
8
9 bind lbvserver <lbvserver> -policy <policy> -priority <priority> -
     Transport Logstream
```

### Konfigurieren einer AppFlow-Aktion

Eine AppFlow-Aktion ist ein Set-Collector, an den die Flow-Datensätze gesendet werden, wenn die zugehörige AppFlow-Richtlinie übereinstimmt.

Verwenden Sie die folgenden Befehle, um eine AppFlow-Aktion zu konfigurieren:

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

Um die Konfiguration zu überprüfen, verwenden Sie den folgenden Befehl:

```
1 show appflow action
```

### Konfigurieren einer AppFlow-Richtlinie

Nachdem Sie eine AppFlow-Aktion konfiguriert haben, müssen Sie als Nächstes eine AppFlow-Richtlinie konfigurieren. Eine AppFlow-Richtlinie basiert auf einer Regel, die aus einem oder mehreren Ausdrücken besteht.

**So konfigurieren Sie eine AppFlow-Richtlinie mithilfe der Befehlszeilenschnittstelle:**

Geben Sie an der Eingabeaufforderung den folgenden Befehl ein, um eine AppFlow-Richtlinie hinzuzufügen und die Konfiguration zu überprüfen:

```
1 add appflow policy <name> <rule> <action>
2
3 show appflow policy <name>
```

### Binden einer AppFlow-Richtlinie

Um eine Richtlinie in Kraft zu setzen, müssen Sie sie entweder global binden, sodass sie für den gesamten Datenverkehr gilt, der durch den NetScaler CPX fließt.

**So binden Sie eine AppFlow-Richtlinie global mithilfe der Befehlszeilenschnittstelle:**

Verwenden Sie den folgenden Befehl, um eine AppFlow-Richtlinie global zu binden:

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

Überprüfen Sie die Konfiguration mit dem folgenden Befehl:

```
show appflow global
```

## Konfigurieren von NetScaler CPX mithilfe einer Konfigurationsdatei

November 23, 2023

Anstatt Befehlszeilenschnittstelle (`cli_script.sh`), NITRO-API oder NetScaler ADM-Konfigurationsaufträge zum Konfigurieren des NetScaler CPX zu verwenden, können Sie den NetScaler CPX mithilfe einer statischen Konfigurationsdatei konfigurieren, während Sie die NetScaler CPX-Instanz bereitstellen.

Sie können eine statische Konfigurationsdatei als Eingabedatei bereitstellen, während Sie den NetScaler CPX-Container bereitstellen. Während des Starts des NetScaler CPX-Containers wird der Container basierend auf der in der statischen Konfigurationsdatei angegebenen Konfiguration konfiguriert. Diese Konfiguration umfasst NetScaler-spezifische Konfiguration und Bash-Shell-Befehle, die Sie dynamisch auf dem NetScaler CPX-Container ausführen können.

### Aufbau der statischen Konfigurationsdatei

Wie bereits erwähnt, wird NetScaler CPX bei der Bereitstellung basierend auf den in der statischen Konfigurationsdatei angegebenen Konfigurationen konfiguriert.

Die statische Konfigurationsdatei ist eine `.conf` Datei, die zwei Tags `#NetScaler Commands` und enthält `#Shell Commands`. Unter dem `#NetScaler Commands` Tag müssen Sie alle NetScaler-Befehle hinzufügen, um die NetScaler-spezifische Konfiguration auf NetScaler CPX zu konfigurieren. Unter dem `#Shell Commands` Tag müssen Sie die Shell-Befehle hinzufügen, die Sie auf NetScaler CPX ausführen möchten.

Während der NetScaler CPX-Containerbereitstellung werden die NetScaler-Befehle und Shell-Befehle in der in der Konfigurationsdatei angegebenen Reihenfolge auf dem Container ausgeführt.

> **Wichtig:**
>
> - Die Tags können in der Konfigurationsdatei mehrfach wiederholt werden.
> - Bei den Markierungen wird die Groß-/Kleinschreibung nicht beachtet
> - Die Konfigurationsdatei muss im `/etc` Verzeichnis als `cpx.conf` Datei im Dateisystem des Containers vorhanden sein.

- Die Konfigurationsdatei kann auch Kommentare enthalten. Sie müssen vor Ihren Kommentaren ein "**#"-Zeichen**hinzufügen.
- Wenn beim Bereitstellen des NetScaler CPX-Containers mit der Konfigurationsdatei Ausfallszenarien auftreten, werden die Fehler in der `ns.log` Datei im Container protokolliert.
- Wenn Sie den NetScaler CPX-Container neu starten, wird die Konfigurationsdatei erneut auf den Container angewendet.

```
#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

#This is a test configuration file
<!--NeedCopy-->
```

Um einen NetScaler CPX-Container zu installieren und den NetScaler CPX-Container basierend auf einer Konfigurationsdatei dynamisch zu konfigurieren, mounten Sie die statische Konfigurationsdatei mit der `-v` Option im `docker run` Befehl:

```
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
<!--NeedCopy-->
```

# Unterstützung für dynamisches Routing in NetScaler CPX

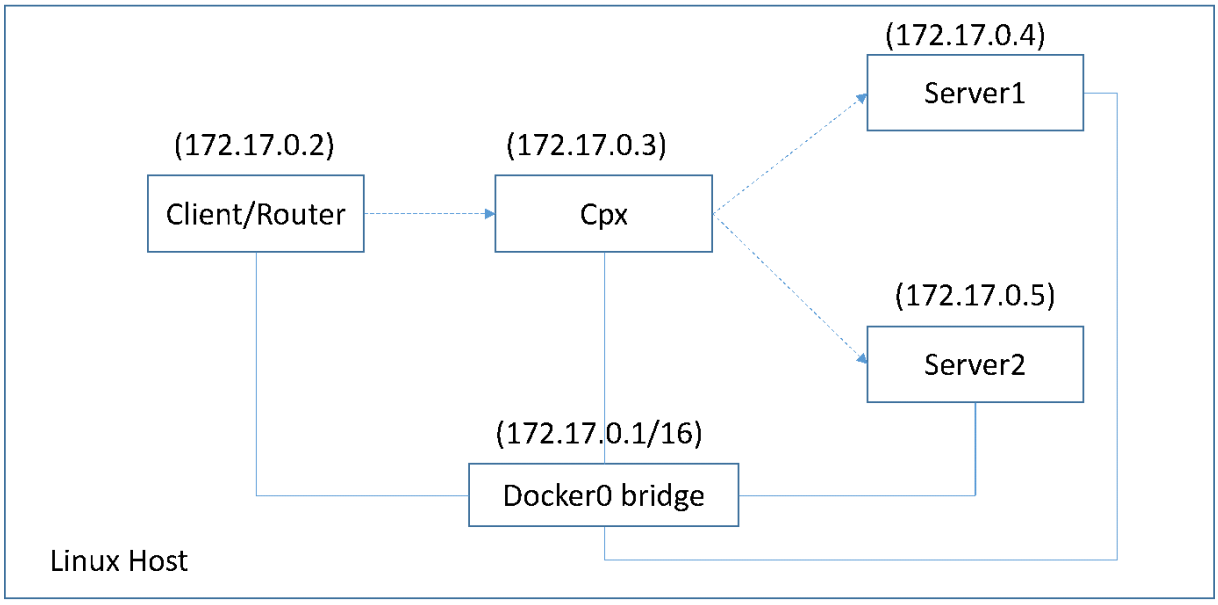
November 23, 2023

NetScaler CPX unterstützt das dynamische Routingprotokoll BGP. Das Hauptziel des dynamischen Routing-Protokolls besteht darin, die IP-Adresse des virtuellen Servers basierend auf der Integrität der an den virtuellen Server gebundenen Dienste bekannt zu geben. Es hilft einem Upstream-Router, die beste aus mehreren Routen zu einem topografisch verteilten virtuellen Server auszuwählen.

Informationen zum nicht standardmäßigen Kennwort in NetScaler CPX finden Sie im `Support for using a non-default password in NetScaler CPX` Abschnitt Konfigurieren von NetScaler CPX .

In einem einzigen Host-Netzwerk werden der Client, die Server und die NetScaler CPX-Instanz als Container auf demselben Docker-Host bereitgestellt. Alle Container sind über die Docker0-Brücke miteinander verbunden. In dieser Umgebung fungiert die NetScaler CPX-Instanz als Proxy für die Anwendungen, die als Container auf demselben Docker-Host bereitgestellt werden. Informationen zur Bereitstellung des NetScaler CPX-Host-Netzwerkmodus finden Sie unter Host-Netzwerkmodus.

Die folgende Abbildung veranschaulicht die Topologie eines einzelnen Hosts.




In dieser Topologie werden virtuelle Server konfiguriert und (basierend auf dem Zustand der Dienste) mithilfe von BGP an das Upstream-Netzwerk oder den Router angekündigt.

Führen Sie die folgenden Schritte aus, um BGP auf NetScaler CPX auf einem einzelnen Docker-Host mit dem Bridge-Netzwerkmodus zu konfigurieren.

### Konfigurieren Sie BGP-basierte Route Health Injection mithilfe der REST-API auf NetScaler CPX

1. Erstellen Sie mit dem folgenden Befehl einen Container aus dem NetScaler CPX-Image:

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

   Beispiel:

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

2. Melden Sie sich mit dem folgenden Befehl beim Container an:

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

3. Aktivieren Sie die BGP-Funktion mit dem folgenden Befehl:

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

4. Besorgen Sie sich das NSIP mit dem `show ns ip` Befehl:

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

5. Fügen Sie den virtuellen Server mit dem folgenden Befehl hinzu:

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

6. Dienste hinzufügen und Dienste an den virtuellen Server binden.

7. Aktivieren Sie `hostroute` für den VIP mit dem folgenden Befehl:

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

   Melden Sie sich vom Container ab und senden Sie BGP NITRO-Befehle vom Host zum NSIP auf dem Port 9080.

8. Konfigurieren Sie den BGP-Router:

   Wenn Sie beispielsweise Folgendes konfigurieren möchten:

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

   Geben Sie den Befehl wie folgt an:

```
 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. Installieren Sie die erlernten BGP-Routen mit dem folgenden NITRO-Befehl in das PE:

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

10. Überprüfen Sie den BGP-Nachbarschaftszustand mit dem folgenden NITRO-Befehl:

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

Beispielausgabe:

```
 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. Stellen Sie mit dem folgenden Befehl sicher, dass die durch BGP erlernten Routen in der Paket-Engine installiert sind:

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

12. Speichern Sie die Konfiguration mit dem folgenden Befehl:

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

Die dynamische Routing-Konfiguration wird in der `/nsconfig/ZebOS.conf` Datei gespeichert.

## Konfigurieren der Hochverfügbarkeit für NetScaler CPX

November 23, 2023

Ein System mit unternehmens- und geschäftskritischen Anwendungen muss kontinuierlich verfügbar sein, ohne dass einzelne Ausfallpunkte auftreten. Systeme mit hoher Verfügbarkeit gewährleisten die kontinuierliche Verfügbarkeit von Anwendungen ohne Unterbrechung der für den Benutzer bereitgestellten Dienste. NetScaler CPX unterstützt die Hochverfügbarkeitsbereitstellung von zwei NetScaler-Instanzen, wodurch die Dienste vor ungeplanten Ausfallzeiten geschützt und die Geschäftskontinuität im Falle eines Ausfalls sichergestellt wird. Sobald Sie die Hochverfügbarkeit konfiguriert haben, können Sie auch die NetScaler CPX-Software aktualisieren, ohne die Dienste für die Benutzer zu unterbrechen.

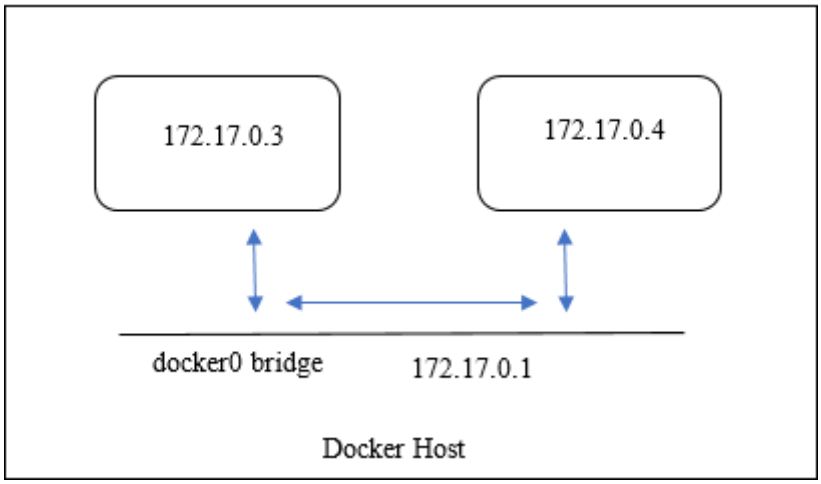
> **Hinweis:**
> Wenn das interne Benutzerkonto deaktiviert ist, wird die Hochverfügbarkeit für NetScaler CPX-Funktion nicht unterstützt.

### Topologie 1: Bereitstellen von NetScaler CPX-Instanzen auf einem einzelnen Docker-Host mit Bridge-Netzwerkmodus

In dieser Topologie werden zwei NetScaler CPX-Knoten auf demselben Docker-Host mit Bridge-Netzwerkmodus erstellt. Beide Knoten befinden sich im selben Brückennetzwerk und die Knoten sind direkt zueinander erreichbar.

Das folgende Diagramm erklärt diese Topologie.




In diesem Beispiel werden zwei NetScaler CPX-Instanzen, CPX-1 (NSIP: 172.17.0.3) und CPX-2 (NSIP: 172.17.0.4), auf demselben Docker-Host erstellt. Für die Unterstützung der Hochverfügbarkeit müssen Sie Hochverfügbarkeitsknoten auf beiden NetScaler CPX-Instanzen mithilfe des NSIP des anderen Knotens konfigurieren.

Führen Sie die folgenden Schritte aus, um die Hochverfügbarkeitsunterstützung auf NetScaler CPX-Instanzen auf einem einzelnen Docker-Host im Bridge-Modus zu konfigurieren.

1. Greifen Sie auf den Docker-Host zu und melden Sie sich an der SSH-Eingabeaufforderung der NetScaler CPX-Instanz an. Weitere Informationen finden Sie unter Konfigurieren einer NetScaler CPX-Instanz mithilfe der Befehlszeilenschnittstelle.
2. Konfigurieren Sie mit dem folgenden Befehl einen Knoten mit hoher Verfügbarkeit auf der CPX-1-Instanz.

```
cli_script.sh ‘add ha node 1 172.17.0.4 [-inc enabled]’
```

3. Konfigurieren Sie mit dem folgenden Befehl einen Knoten mit hoher Verfügbarkeit auf der CPX-2-Instanz.

```
cli_script.sh ‘add ha node 1 172.17.0.3 [-inc enabled]’
```

> **Hinweis:**
> Wenn ein NetScaler CPX-Knoten im Bridge-Netzwerkmodus neu gestartet wird, kann sich die einem NetScaler CPX zugewiesene IP-Adresse je nach Docker-Version auf dem Host ändern. Wenn sich das NSIP eines der Knoten nach dem Neustart eines NetScaler CPX ändert, funktioniert die vorhandene Hochverfügbarkeitskonfiguration nicht, obwohl die Konfiguration gespeichert ist. In diesem Fall müssen Sie die Hochverfügbarkeit auf NetScaler CPX-Knoten erneut konfigurieren.

### Topologie 2: Bereitstellen von NetScaler CPXs auf verschiedenen Docker-Hosts im Bridge-Netzwerkmodus

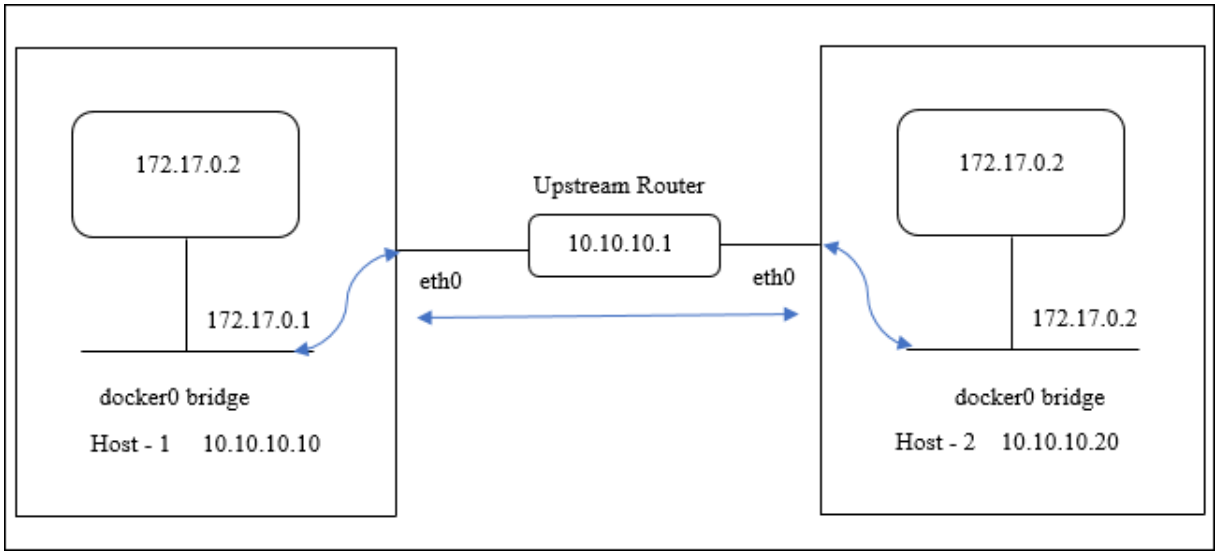
In dieser Topologie werden zwei NetScaler CPX-Instanzen im Bridge-Modus auf zwei verschiedenen Docker-Hosts bereitgestellt, die voneinander erreichbar sind. In dieser Bereitstellung muss NetScaler CPX die IP-Adresse des Hosts kennen. Die **HOST-Umgebungsvariable** kann zum Zeitpunkt der Bereitstellung des NetScaler CPX verwendet werden, um NetScaler CPX auf die IP-Adresse des Hosts aufmerksam zu machen.

Sie müssen die Portzuordnung für NetScaler CPX-Knoten festlegen. Sie können die Option -p des **Docker-Befehls run** verwenden, während Sie den NetScaler CPX-Knoten erstellen, um die Portzuordnung für die erforderlichen Ports zu aktivieren.

Sie müssen die folgenden Ports zuordnen:

- UDP 3003
- TCP 3008
- TCP-8873

Das folgende Diagramm erklärt die Topologie der Bereitstellung von zwei NetScaler CPX-Instanzen im Bridge-Modus auf zwei verschiedenen Docker-Hosts.




In diesem Diagramm stellt die gerade blaue Linie den Fluss des CPX-HA-Datenverkehrs zwischen zwei Hosts dar.

> **Hinweis:** Auf einem Docker-Host kann nur ein NetScaler CPX ein Hochverfügbarkeitspaar bilden. Jeder andere NetScaler CPX auf demselben Host kann kein Hochverfügbarkeitspaar mit einem anderen NetScaler CPX auf einem anderen Host bilden.

Führen Sie die folgenden Schritte aus, um NetScaler-Instanzen im Bridge-Modus auf verschiedenen

Docker-Hosts bereitzustellen und die Hochverfügbarkeitsunterstützung mithilfe der Beispieltopologie zu konfigurieren.

In diesem Beispiel ist die host1-IP-Adresse als `10.10.10.10/24` und die host2-IP-Adresse als konfiguriert `10.10.10.20/24`.

1. Stellen Sie NetScaler CPX mit der erforderlichen Port-Zuordnung auf host1 mit dem folgenden Befehl bereit.

```
Docker run -dt --privileged=true -e EULA=yes --ulimit core=-1 -p
    8873:8873 -p 3003:3003/udp -p 3008:3008 -e Host=10.10.10.10 cpx
    :latest
```

2. Stellen Sie NetScaler CPX auf host2 mit demselben Befehl mit der IP-Adresse von Host 2 bereit.

```
docker run -dt --privileged=true -e EULA=yes --ulimit core=-1 -p
    8873:8873 -p 3003:3003/udp -p 3008:3008 -e HOST=10.10.10.20 cpx
    :latest
```

3. Konfigurieren Sie mit dem folgenden Befehl einen Knoten mit hoher Verfügbarkeit auf der CPX-1-Instanz.

```
cli_script.sh  ' add ha node 1 10.10.10.20 -inc enabled '
```

4. Konfigurieren Sie mit dem folgenden Befehl einen Knoten mit hoher Verfügbarkeit auf der CPX-2-Instanz.

```
cli_script.sh  ' add ha node 1 10.10.10.10 -inc enabled '
```

> **Hinweis:** In dieser Bereitstellung müssen Sie die Host-IP-Adresse des Hochverfügbarkeitsknotens anstelle der NSIP-Adresse des Hochverfügbarkeitsknotens verwenden.

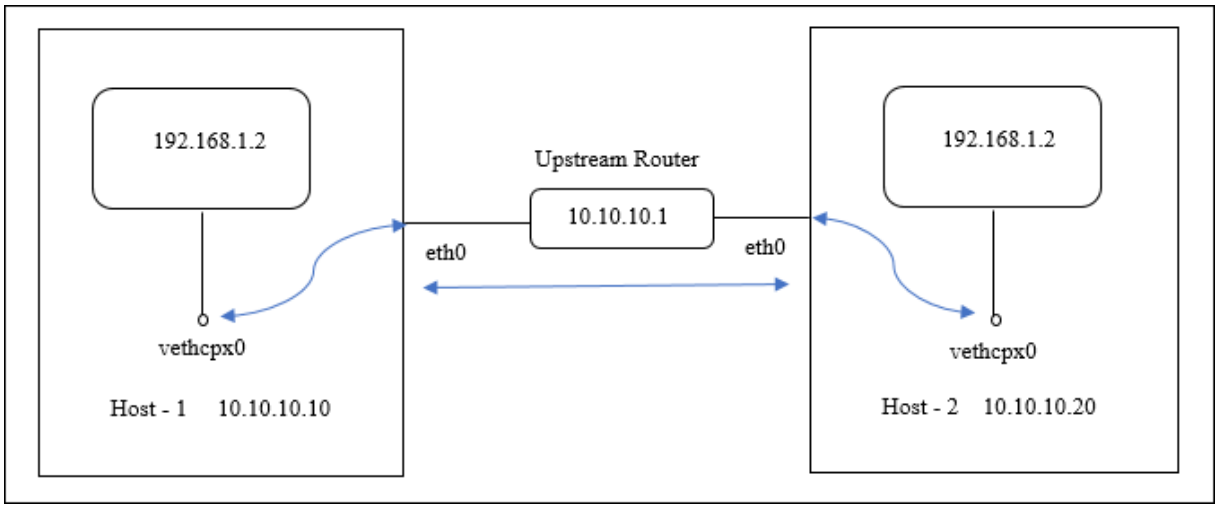
### Topologie 3: Bereitstellen von NetScaler CPXs auf verschiedenen Docker-Hosts im Host-Netzwerkmodus ohne dediziertes Interface

In dieser Topologie werden zwei NetScaler CPX-Instanzen auf zwei verschiedenen Docker-Hosts im Hostmodus ohne dedizierte Schnittstelle bereitgestellt. Die Hosts müssen füreinander erreichbar sein.
In dieser Bereitstellung muss NetScaler CPX die IP-Adresse des Hosts kennen. Sie können die Umgebungsvariable **HOST** während der Bereitstellung von NetScaler CPX verwenden, um sie auf die IP-Adresse des Hosts aufmerksam zu machen.

Sie müssen die Portzuordnung für den NetScaler CPX-Knoten festlegen. Sie können die Option -p des **Docker-Befehls run** verwenden, während Sie den NetScaler CPX-Knoten erstellen, um die Portzuordnung für die erforderlichen Ports zu aktivieren.

Das folgende Diagramm erklärt die Topologie.




In diesem Diagramm stellt die gerade blaue Linie den Fluss des CPX-HA-Datenverkehrs zwischen zwei Hosts dar.

> **Hinweis:** Auf einem Docker-Host können Sie nur einen NetScaler CPX im Host-Modus bereitstellen.

Führen Sie die folgenden Schritte aus, um die NetScaler CPX-Instanzen bereitzustellen und die Hochverfügbarkeitsunterstützung mithilfe der Beispieltopologie zu konfigurieren.

1. Stellen Sie NetScaler CPX mit der erforderlichen Port-Zuordnung und auf host1 mit dem folgenden Befehl bereit.

```
docker run -dt --privileged=true -e EULA=yes --ulimit core=-1 --
    net=host -e NS_NETMODE=HOST -e HOST=10.10.10.10 cpx:latest
```

2. Stellen Sie NetScaler CPX auf host2 mit der IP-Adresse von host2 mit dem folgenden Befehl bereit.

```
docker run -dt --privileged=true -e EULA=yes --ulimit core=-1
--net=host -e NS_NETMODE=HOST -e HOST=10.10.10.20 cpx:latest
```

3. Konfigurieren Sie mit dem folgenden Befehl einen Knoten mit hoher Verfügbarkeit auf der CPX-1-Instanz.

```
cli_script.sh  ‘ add ha node 1 10.10.10.20 -inc enabled
```

4. Konfigurieren Sie mit dem folgenden Befehl einen Knoten mit hoher Verfügbarkeit auf der CPX-2-Instanz.

```
cli_script.sh ‘ add ha node 1 10.10.10.10 -inc enabled ’
```

### Topologie 4: Bereitstellung von CPXs auf verschiedenen Docker-Hosts mit Host-Netzwerkmodus und dedizierten Schnittstellen

In dieser Topologie werden zwei NetScaler CPX-Instanzen im Host-Netzwerkmodus auf verschiedenen Docker-Hosts bereitgestellt. Die Hosts müssen über mehr als eine Schnittstelle verfügen. Sie können die dedizierte Schnittstelle für NetScaler CPX angeben, indem Sie die Umgebungsvariable **CPX_NW_DEV** verwenden.

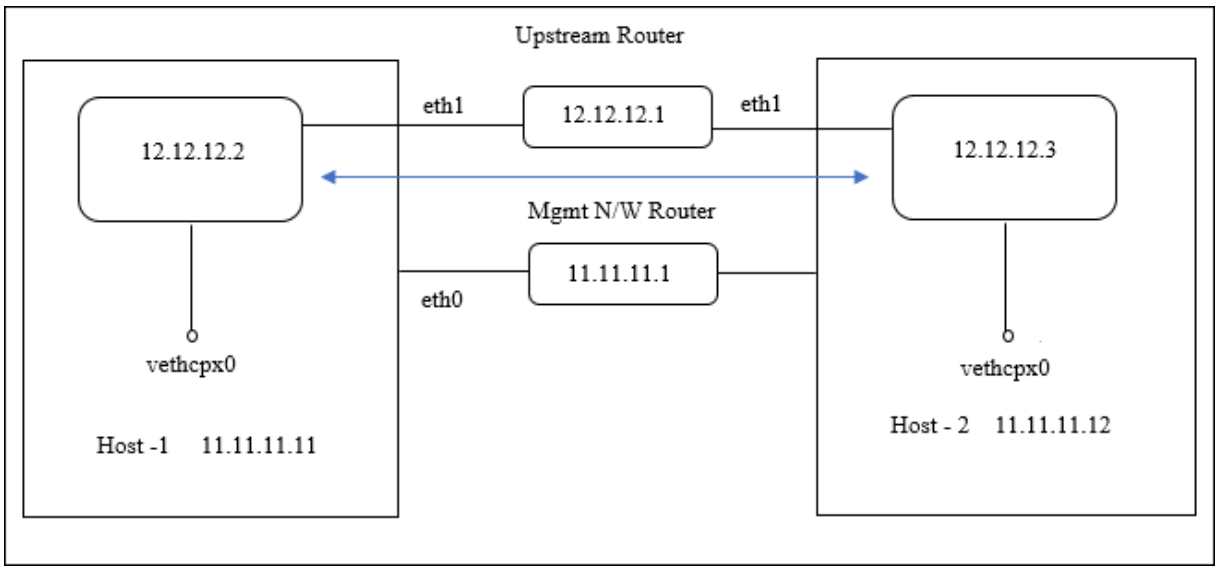
Weitere Informationen zum Zuweisen dedizierter Netzwerkschnittstellen für NetScaler CPX mithilfe der Umgebungsvariablen CPX_NW_DEV finden Sie unter Bereitstellen der NetScaler CPX-Instanz mit dem Docker-Run-Befehl.

NetScaler CPXs, die auf verschiedenen Docker-Hosts bereitgestellt werden, müssen in diesem Datennetzwerk mit der dedizierten Schnittstelle füreinander erreichbar sein.

Diese Konfiguration ermöglicht es Hochverfügbarkeitsknoten, Heartbeat-Nachrichten auszutauschen und Konfigurationsdateien zu synchronisieren, indem sie direkt über die Ports 3003, 3008 und 8873 kommunizieren. Es sind keine NAT-Regeln auf dem Host erforderlich. Das Standard-NSIP von NetScaler CPX, das im Hostmodus erstellt wurde, ist auf beiden Knoten identisch. Daher müssen Sie auch die **NS_IP- und**NS_GATEWAY-Informationen** angeben.

In diesem Beispiel werden zwei NetScaler CPXs im Host-Modus auf zwei verschiedenen Hosts erstellt. NetScaler CPX-Instanzen besitzen die **eth1-Schnittstellen** auf beiden Hosts, und **eth1-Schnittstellen** sind mit demselben Netzwerk verbunden.

Das folgende Diagramm erklärt die Topologie. In diesem Diagramm stellt der blaue Pfeil den Fluss des CPX-HA-Datenverkehrs in dem an die eth1-Schnittstelle angeschlossenen Netzwerk dar.




> **Hinweis:** Auf einem Docker-Host können Sie nur einen NetScaler CPX im Host-Modus bereitstellen.

Führen Sie die folgenden Schritte aus, um die NetScaler CPX-Instanzen bereitzustellen und die Hochverfügbarkeitsunterstützung mithilfe der Beispieltopologie zu konfigurieren.

1. Stellen Sie NetScaler CPX im Hostmodus auf host1 mit dem folgenden Befehl bereit.

```
docker run -dt --privileged=true --net=host -e NS_NETMODE="HOST" -
    e CPX_NW_DEV=eth1 -e NS_IP='12.12.12.2' -e NS_GATEWAY='
    12.12.12.9' -e EULA=yes --ulimit core=-1 cpx:latest
```

2. Stellen Sie NetScaler CPX mit dem folgenden Befehl im Hostmodus auf host2 bereit.

```
docker run -dt --privileged=true --net=host -e NS_NETMODE="HOST" -
    e CPX_NW_DEV=eth1 -e NS_IP='12.12.12.3' -e NS_GATEWAY='
    12.12.12.10' -e EULA=yes --ulimit core=-1 cpx:latest
```

> **Hinweis:** Sie müssen statische Routen für beide NetScaler CPX-Knoten konfigurieren, um den anderen NetScaler CPX-Knoten zu erreichen, um Heartbeat-Nachrichten auszutauschen und Konfigurationsdateien zu synchronisieren.

3. Konfigurieren Sie mit dem folgenden Befehl einen Knoten mit hoher Verfügbarkeit auf der CPX-1-Instanz.

```
cli_script.sh  ' add ha node 1 12.12.12.3 [-inc enabled] '
```

4. Konfigurieren Sie mit dem folgenden Befehl einen Knoten mit hoher Verfügbarkeit auf der CPX-2-Instanz.

```
cli_script.sh  ' add high availability node 1 12.12.12.2 [-inc
    enabled] '
```

## Konfigurieren von Docker-Protokolltreibern

November 23, 2023

Docker enthält Protokollierungsmechanismen, die als "Protokollierungstreiber"bezeichnet werden, damit Sie Informationen aus den laufenden Containern Sie können einen NetScaler CPX-Container so konfigurieren, dass er generierte Protokolle an die Docker-Protokollierungstreiber weiterleitet. Weitere Informationen zu Docker-Protokollierungstreibern finden Sie unter Konfigurieren von Protokolltreibern

Standardmäßig werden alle vom NetScaler CPX-Container generierten Protokolle in einer `/cpx/log/ns.log` Datei auf dem Docker-Host gespeichert. Wenn Sie den NetScaler CPX-Container mit dem Docker-Befehl starten, können Sie ihn so konfigurieren, dass alle generierten Protokolle mithilfe der `--log-driver` Option an einen Docker-Protokollierungstreiber weitergeleitet werden. Wenn der

Protokolltreiber über konfigurierbare Parameter verfügt, können Sie diese mit der `--log-opt <NAME>=<VALUE>` Option festlegen.

Im folgenden Beispiel ist der NetScaler CPX-Container so konfiguriert, dass er alle generierten Protokolle mit Syslog als Protokollierungstreiber weiterleitet.

```
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-->
```

In ähnlicher Weise ist im folgenden Beispiel der NetScaler CPX-Container so konfiguriert, dass er alle generierten Protokolle mit Splunk als Protokollierungstreiber weiterleitet.

```
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-->
```

## Aktualisieren einer NetScaler CPX-Instanz

November 23, 2023

Sie können eine NetScaler CPX-Instanz aktualisieren, indem Sie sie herunterfahren, die neueste Version auf demselben Bereitstellungspunkt installieren und dann die alte Instanz löschen. Ein Einhängepunkt ist ein Verzeichnis, in das Sie das **/cpx-Verzeichnis** auf dem Host einhängen.

Um beispielsweise das Verzeichnis **/cpx** der vorhandenen NetScaler CPX-Instanz im Verzeichnis **/var/cpx** des Hosts zu mounten, ist der Bereitstellungspunkt **/var/cpx** und das NetScaler CPX-Mount-Verzeichnis ist **/cpx**, wie unten gezeigt:

```
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-->
```

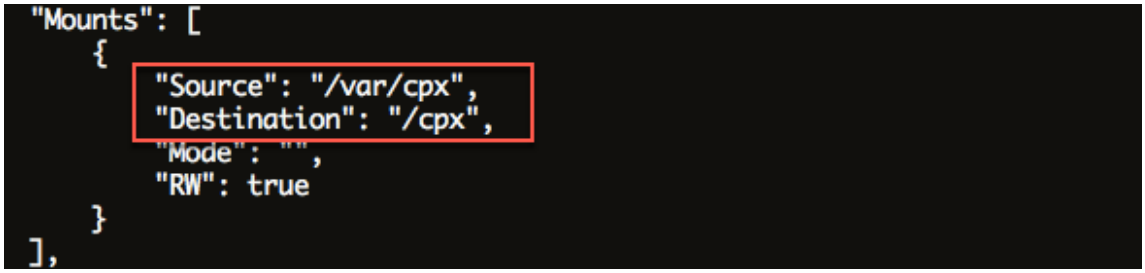
### Voraussetzungen

Folgende Voraussetzungen müssen erfüllt sein:

- Details des Hostverzeichnisses, in dem Sie das Verzeichnis `/cpx` der vorhandenen NetScaler CPX-Instanz gemountet haben. Sie können den Befehl `docker inspect <containerName>` verwenden, wobei `<containerName>` der Name des NetScaler CPX-Containers ist, um Informationen zum Hostverzeichnis anzuzeigen.

Die Ausgabe des Befehls liefert die Details der Containerkonfigurationen, einschließlich der Volumes. Im Eintrag **Mounts** zeigt der Untereintrag **Source** den Speicherort des Hostverzeichnisses auf dem Host an.




- Laden Sie die neueste NetScaler CPX Docker-Imagedatei herunter und laden Sie das NetScaler CPX Docker-Image. Um das Image zu laden, navigieren Sie zu dem Verzeichnis, in dem Sie die Docker-Imagedatei gespeichert haben. Verwenden Sie den Befehl `docker load -i <image_name>`, um das Image zu laden. Nachdem das NetScaler CPX-Image geladen wurde, können Sie den Befehl Docker Images eingeben, um Informationen zum Image anzuzeigen:

```
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-->
```

### So aktualisieren Sie eine NetScaler CPX-Instanz

1. Stoppen Sie die vorhandene NetScaler CPX-Instanz, indem Sie den Befehl `docker stop <containerName>`eingeben, wobei `<containerName>` der Name der NetScaler CPX-Instanz ist.

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

2. Stellen Sie mit dem Befehl `docker run` die neueste NetScaler CPX-Instanz aus dem NetScaler CPX-Image bereit, das Sie auf den Host geladen haben. Stellen Sie sicher, dass Sie die Instanz am selben Bereitstellungspunkt (z. B. `/var/cpx:/cpx`) bereitstellen, den Sie für die vorhandene NetScaler CPX-Instanz verwendet haben.

```
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-->
```

Sie können den Befehl `docker ps` eingeben, um sicherzustellen, dass die bereitgestellte NetScaler CPX-Instanz die neueste Version ist.

```
```
    root@ubuntu:~# docker ps

    CONTAINER ID        IMAGE               COMMAND
       CREATED             STATUS                          PORTS
                          NAMES

    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
<!--NeedCopy--> ```
```

3. Nachdem Sie überprüft haben, dass Sie die richtige NetScaler CPX-Instanz bereitgestellt haben, geben Sie den Docker-Befehl **rm <containerName>** ein, um die ältere Instanz zu löschen.

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

## Verwenden virtueller Platzhalterserver in der NetScaler CPX-Instanz

November 23, 2023

Wenn Sie eine NetScaler-Instanz bereitstellen, wird einer NetScaler CPX-Instanz von der Docker-Engine nur eine private IP-Adresse (einzelne IP-Adresse) zugewiesen. Die drei IP-Funktionen einer NetScaler-Instanz werden auf eine IP-Adresse gemultiplext. Diese einzelne IP-Adresse verwendet unterschiedliche Portnummern, um als NSIP, SNIP und VIPs zu fungieren.

Die einzelne IP-Adresse, die von der Docker-Engine zugewiesen wird, ist dynamisch. Fügen Sie die virtuellen Server Load Balancing (LB) oder Content Switching (CS) mit der einzelnen IP-Adresse oder mit der IP-Adresse 127.0.0.1 hinzu. Die virtuellen Server, die mit 127.0.0.1 erstellt wurden, werden als virtuelle Wildcard-Server bezeichnet. Wenn Sie einen virtuellen Platzhalterserver erstellen, ersetzt der NetScaler CPX standardmäßig die zugewiesene IP-Adresse des virtuellen Platzhalterservers. Die zugewiesene IP-Adresse ist 127.0.0.1 und wird durch das NSIP ersetzt, das der NetScaler CPX-Instanz von der Docker-Engine zugewiesen wurde.

In NetScaler CPX-Bereitstellungen mit hoher Verfügbarkeit können Sie virtuelle Platzhalterserver auf der primären NetScaler CPX-Instanz hinzufügen. Die Konfigurationssynchronisierung zwischen Knoten konfiguriert den virtuellen Platzhalterserver auf der sekundären NetScaler CPX-Instanz. Dadurch entfällt die Konfiguration des virtuellen Servers auf dem NSIP, das von der Docker-Engine den NetScaler CPX-Instanzen zugewiesen wurde.

 
```

**Wichtige Hinweise:**

- Stellen Sie sicher, dass die Portnummer, die Sie dem virtuellen Platzhalterserver zuweisen, von keinem anderen virtuellen Server in der Bereitstellung verwendet wird.
- Das Hinzufügen eines virtuellen Platzhalterservers schlägt fehl, wenn die Portnummer, die Sie dem virtuellen Platzhalterserver zuweisen, bereits von den internen Diensten verwendet wird.
- Der virtuelle Platzhalterserver unterstützt das Zeichen* nicht.

Um einen virtuellen Platzhalterserver für den Lastenausgleich zu erstellen, geben Sie an der Eingabeaufforderung den folgenden Befehl ein:

```
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-->
```

Um einen virtuellen Platzhalterserver für Content Switching zu erstellen, geben Sie an der Eingabeaufforderung den folgenden Befehl ein:

```
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-->
```

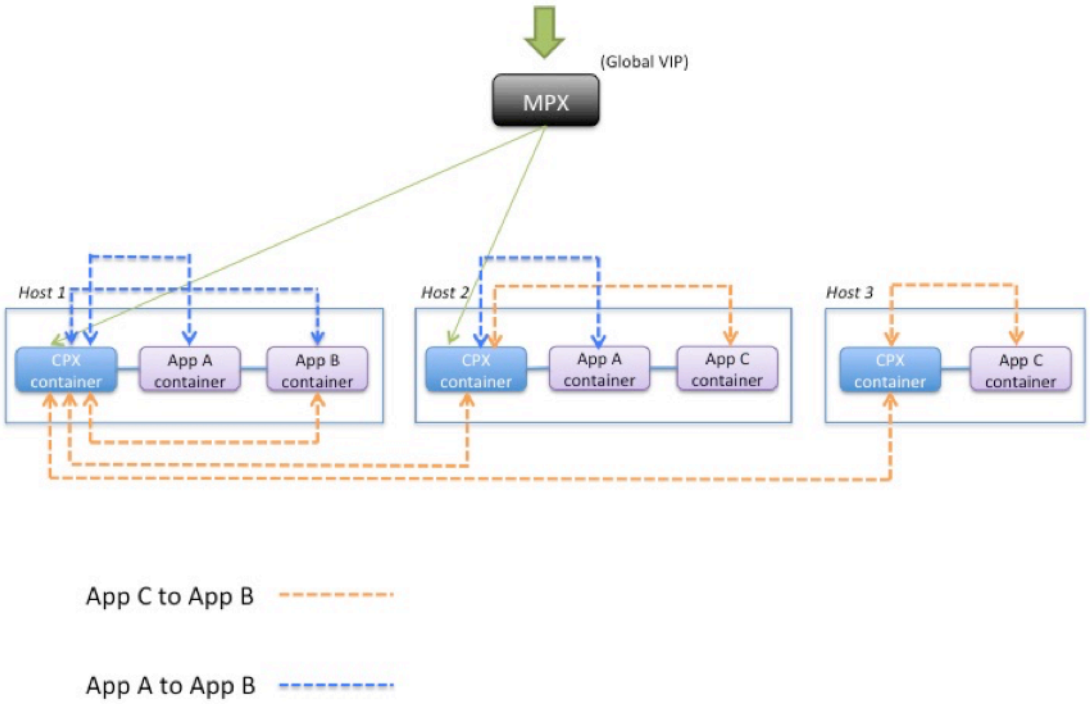
## Stellen Sie NetScaler CPX als Proxy bereit, um den Ost-West-Verkehrsfluss zu ermöglichen

November 23, 2023

In dieser Bereitstellung fungiert die NetScaler CPX-Instanz als Proxy, um die Kommunikation zwischen Anwendungscontainern zu ermöglichen, die sich auf mehreren Hosts befinden. Die NetScaler CPX-Instanz wird zusammen mit den Anwendungen auf mehreren Hosts bereitgestellt und bietet den kürzesten Weg für die Kommunikation.

Das folgende Bild zeigt den Verkehrsfluss zwischen zwei Anwendungen durch die NetScaler CPX-Instanzen.

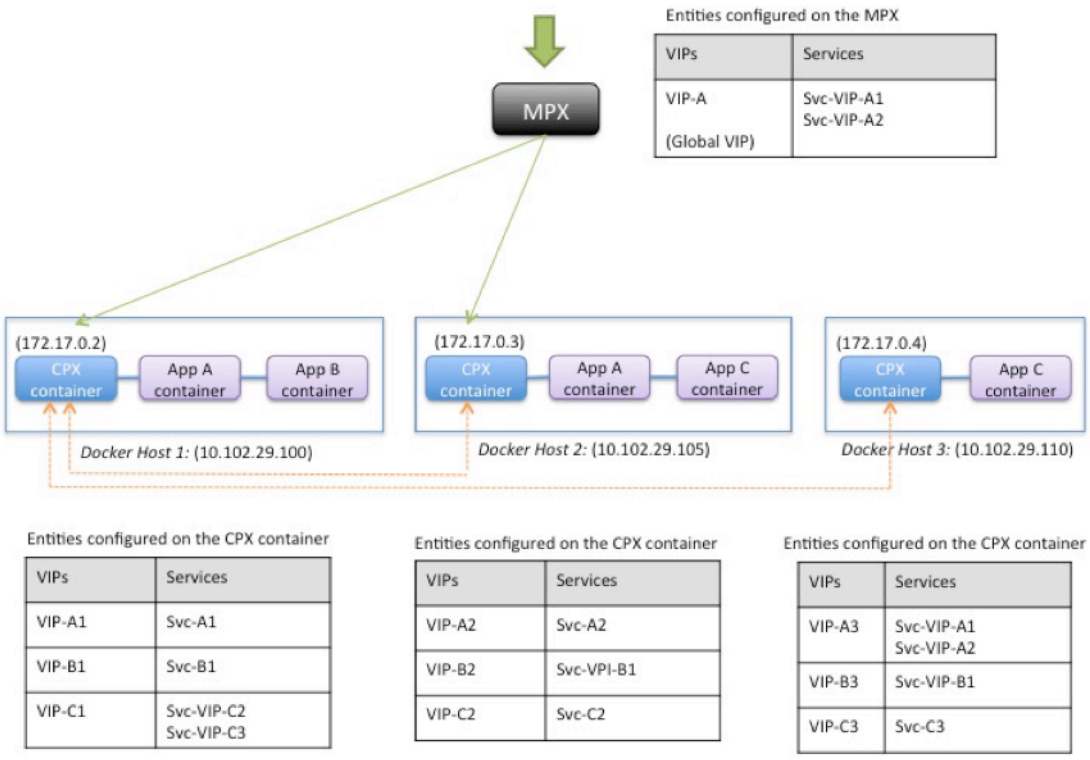



Dieses Bild zeigt den Verkehrsfluss zwischen Anwendung C und Anwendung B sowie zwischen Anwendung A und Anwendung B. Wenn App C (in einem der Hosts) eine Anforderung an B sendet, wird die Anforderung zuerst im NetScaler CPX-Container auf demselben Host wie App C empfangen. Dann übergibt der NetScaler CPX-Container den Datenverkehr. an den NetScaler CPX-Container, der auf demselben Host wie App B gehostet wird, und dann wird der Datenverkehr an App B weitergeleitet. Ein ähnlicher Verkehrspfad wird verfolgt, wenn App A eine Anfrage an App B sendet.

In diesem Beispiel wird auch ein NetScaler MPX bereitgestellt, um den Datenverkehr zu den Anwendungen aus dem Internet über einen globalen VIP zu ermöglichen.Der Datenverkehr vom NetScaler MPX wird auf den NetScaler CPX-Containern empfangen, die dann den Datenverkehr auf die Anwendungscontainer verteilen.

Das folgende Diagramm veranschaulicht diese Topologie mit den Konfigurationen, die für die Kommunikation festgelegt werden müssen.




In der folgenden Tabelle sind die IP-Adressen und Ports aufgeführt, die in dieser Beispielkonfiguration auf den NetScaler CPX-Instanzen konfiguriert sind.

| 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 | | | | |

Um dieses Beispielszenario zu konfigurieren, führen Sie den folgenden Befehl an der Linux-Shell-Eingabeaufforderung aus, während Sie den NetScaler CPX-Container auf allen drei Docker-Hosts

erstellen:

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

Führen Sie die folgenden Befehle aus, indem Sie entweder die Funktion Jobs in NetScaler ADM oder mithilfe von NITRO-APIs verwenden.

Auf der NetScaler CPX-Instanz auf 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-->
```

Auf der NetScaler CPX-Instanz auf Docker-Host 2:

```
    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-->
```

Auf der NetScaler CPX-Instanz auf Docker-Host 3:

```
    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-->
```

# Bereitstellen von NetScaler CPX in einem einzigen Host-Netzwerk

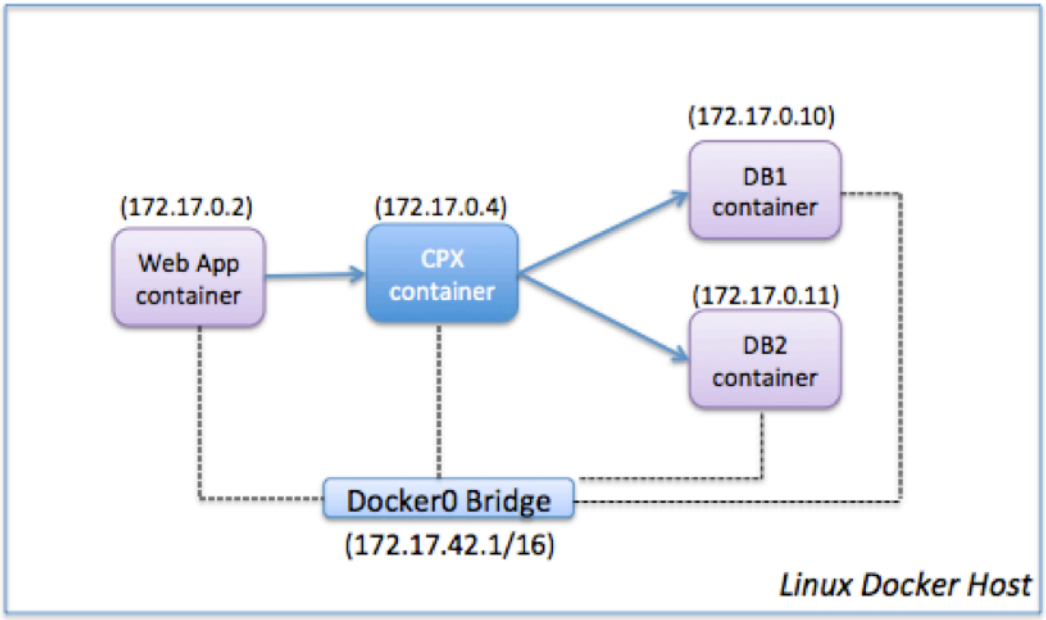
November 23, 2023

In einem einzelnen Host-Netzwerk fungiert die NetScaler CPX-Instanz als Proxy zwischen Anwendungscontainern auf demselben Host. In dieser Eigenschaft bietet die NetScaler CPX-Instanz Skalierbarkeit und Sicherheit für die containerbasierten Anwendungen. Darüber hinaus optimiert es die Leistung und bietet einen Einblick in Telemetriedaten.

In einem einzigen Host-Netzwerk werden der Client, die Server und die NetScaler CPX-Instanz als Container auf demselben Docker-Host bereitgestellt. Alle Container sind über die Docker0-Brücke verbunden.

In dieser Umgebung fungiert die NetScaler CPX-Instanz als Proxy für die Anwendungen, die als Container auf demselben Docker-Host bereitgestellt werden.

Die folgende Abbildung veranschaulicht die Topologie eines einzelnen Hosts.




In diesem Beispiel ist ein Webanwendungscontainer (172.17.0.2) der Client und die beiden Datenbankcontainer DB1 (172.17.0.10) und DB2 (172.17.0.11) sind die Server. Der NetScaler CPX-Container (172.17.0.4) befindet sich zwischen dem Client und den Servern, die als Proxy fungieren.

Damit die Webanwendung über NetScaler CPX mit den Datenbankcontainern kommunizieren kann, müssen Sie zunächst zwei Dienste im NetScaler CPX-Container so konfigurieren, dass sie die beiden Server darstellen. Konfigurieren Sie dann einen virtuellen Server mithilfe der NetScaler CPX-IP-Adresse und eines nicht standardmäßigen HTTP-Ports (z. B. 81), da der NetScaler CPX den

Standard-HTTP-Port 80 für die NITRO-Kommunikation reserviert.

In dieser Topologie müssen Sie keine NAT-Regeln konfigurieren, da sich der Client und der Server im selben Netzwerk befinden.

Um dieses Szenario zu konfigurieren, führen Sie die folgenden Befehle aus, indem Sie entweder die Funktion Jobs in NetScaler ADM oder mithilfe von NITRO-APIs verwenden:

```
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-->
```

## Bereitstellen von NetScaler CPX in einem Multi-Host-Netzwerk

November 23, 2023

Eine NetScaler CPX-Instanz in einem Multi-Host-Netzwerk kann in einer Produktionsbereitstellung im Rechenzentrum konfiguriert werden, wo sie Lastausgleichsfunktionen bereitstellt. Es kann ferner Überwachungsfunktionen und Analysedaten bereitstellen.

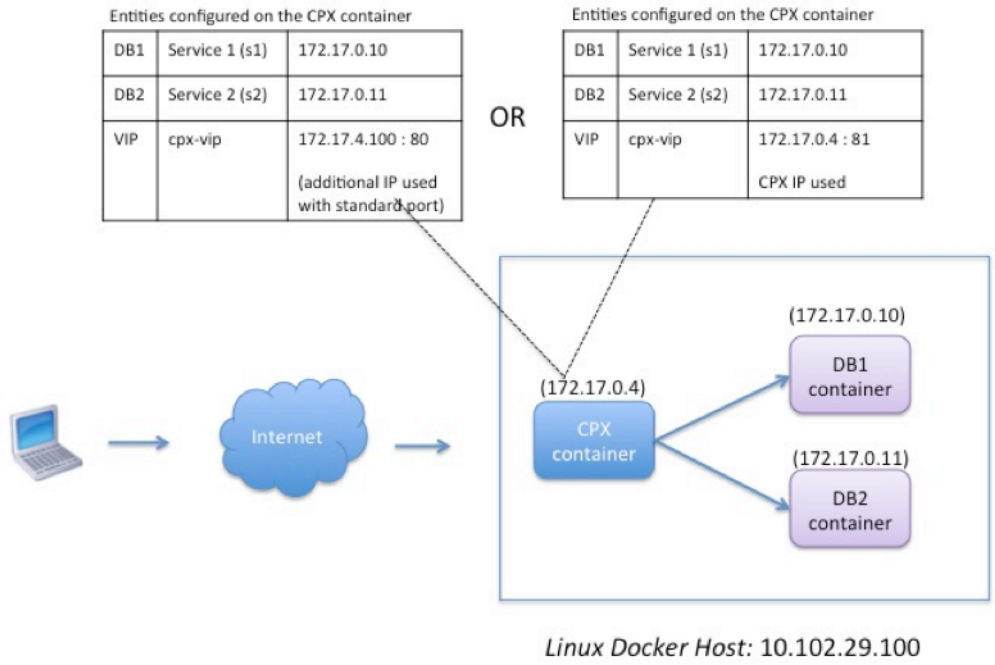
In einem Multi-Host-Netzwerk werden die NetScaler CPX-Instanzen, Backend-Server und die Clients auf verschiedenen Hosts bereitgestellt. Sie können Multi-Host-Topologien in Produktionsbereitstellungen verwenden, bei denen die NetScaler CPX-Instanz eine Reihe von Container-basierten Anwendungen und Servern oder sogar physische Server ausgleicht.

### Topologie 1: NetScaler CPX und Backend-Server auf demselben Host; Client in einem anderen Netzwerk

In dieser Topologie werden die NetScaler CPX-Instanz und die Datenbankserver auf demselben Docker-Host bereitgestellt, der Clientdatenverkehr stammt jedoch von einer anderen Stelle im Netzwerk. Diese Topologie kann in einer Produktionsbereitstellung verwendet werden, in der die NetScaler CPX-Instanz eine Reihe von Container-basierten Anwendungen oder Servern ausgleicht.

Das folgende Diagramm veranschaulicht diese Topologie.




In diesem Beispiel werden die NetScaler CPX-Instanz (172.17.0.4) und die beiden Server DB1 (172.17.0.10) und DB2 (172.17.0.11) auf demselben Docker-Host mit der IP-Adresse 10.102.29.100 bereitgestellt. Der Client befindet sich an einer anderen Stelle im Netzwerk.

Die aus dem Internet stammenden Clientanforderungen werden auf dem VIP empfangen, der auf der NetScaler CPX-Instanz konfiguriert ist und die Anforderungen dann auf die beiden Server verteilt.

Es gibt zwei Methoden, mit denen Sie diese Topologie konfigurieren können:

**Methode 1:** Verwenden einer zusätzlichen IP-Adresse und eines Standardports für den VIP

1. Konfigurieren Sie den VIP auf dem NetScaler CPX-Container mithilfe einer zusätzlichen IP-Adresse.
2. Konfigurieren Sie eine zusätzliche IP-Adresse für den Docker-Host.
3. Konfigurieren Sie NAT-Regeln, um den gesamten auf der zusätzlichen IP-Adresse des Docker-Hosts empfangenen Datenverkehr an die zusätzliche IP-Adresse des VIP weiterzuleiten.
4. Konfigurieren Sie die beiden Server als Dienste auf der NetScaler CPX-Instanz.
5. Binden Sie die Dienste abschließend an den VIP.

Beachten Sie, dass in dieser Beispielkonfiguration das 10.x.x.x-Netzwerk ein öffentliches Netzwerk bezeichnet.

Um dieses Beispielszenario zu konfigurieren, führen Sie die folgenden Befehle aus, indem Sie entweder die Funktion Jobs in NetScaler ADM oder mithilfe von NITRO-APIs verwenden:

```
1      add service s1 172.17.0.10 HTTP 80
2      add service s2 172.17.0.11 HTTP 80
3      add lb vserver cpx-vip HTTP 172.17.4.100 80
4      bind lb vserver cpx-vip s1
5      bind lb vserver cpx-vip s2
6  <!--NeedCopy-->
```

Konfigurieren Sie eine zusätzliche öffentliche IP-Adresse für den Docker-Host und eine NAT-Regel, indem Sie die folgenden Befehle an der Linux-Shell-Eingabeaufforderung ausführen:

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

**Methode 2:** Verwenden der NetScaler CPX-IP-Adresse für den VIP und Konfigurieren der Portzuordnung:

1. Konfigurieren Sie den VIP und die beiden Dienste auf der NetScaler CPX-Instanz. Verwenden Sie einen nicht standardmäßigen Port (81) mit dem VIP.
2. Binden Sie die Dienste an den VIP.
3. Konfigurieren Sie eine NAT-Regel, um den gesamten auf Port 50000 des Docker-Hosts empfangenen Datenverkehr an den VIP und Port 81 weiterzuleiten.

Um dieses Beispielszenario zu konfigurieren, führen Sie den folgenden Befehl an der Linux-Shell-Eingabeaufforderung aus, während Sie den NetScaler CPX-Container auf allen drei Docker-Hosts erstellen:

```
1      docker run -dt -p 22 -p 80 -p 161/udp -p 50000:81 --ulimit core=-1
          --privileged=true cpx:6.2
2
3  <!--NeedCopy-->
```

Nachdem die NetScaler CPX-Instanz bereitgestellt wurde, führen Sie die folgenden Befehle aus, indem Sie entweder die Funktion Jobs in NetScaler ADM oder mithilfe von NITRO-APIs verwenden:

```
1      add service s1 172.17.0.10 http 80
2      add service s2 172.17.0.11 http 80
3      add lb vserver cpx-vip HTTP 172.17.0.4 81
4      bind lb vserver cpx-vip s1
5      bind lb vserver cpx-vip s2
6  <!--NeedCopy-->
```

> **Hinweis:**
>
> Wenn Sie während der Bereitstellung der NetScaler CPX-Instanz keine Portzuordnung konfiguriert haben, konfigurieren Sie eine NAT-Regel, indem Sie die folgenden Befehle an der Linux-Shell-Eingabeaufforderung ausführen:

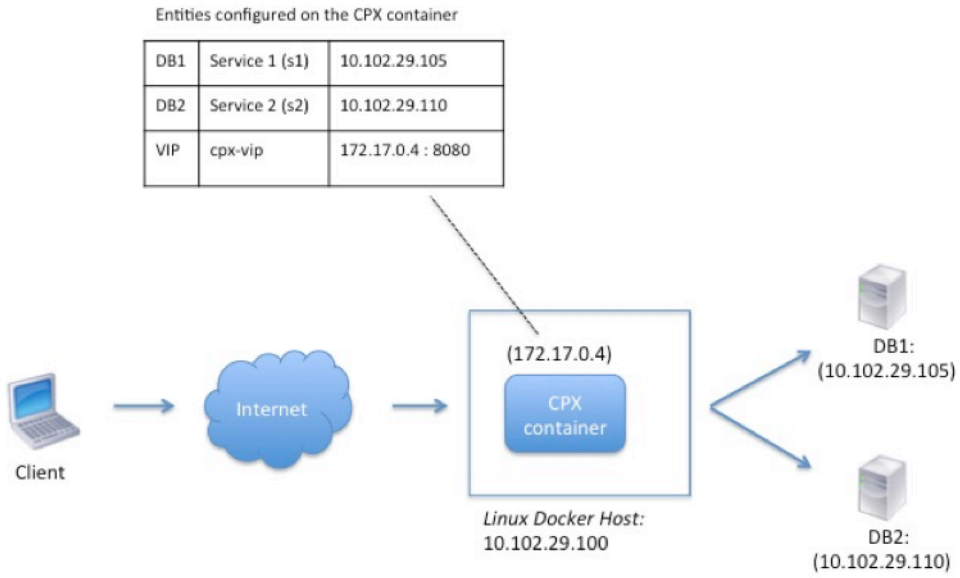
> **iptables -t nat -A PREROUTING -p tcp -m addrtype –dst-type LOCAL -m tcp –dport** 50000 **-j DNAT –to-destination** 172.17.0.4:81

### Topologie 2: NetScaler CPX mit physischen Servern und Client

In dieser Topologie wird nur die NetScaler CPX-Instanz auf einem Docker-Host bereitgestellt. Der Client und die Server sind nicht containerbasiert und befinden sich an anderer Stelle im Netzwerk.

In dieser Umgebung können Sie die NetScaler CPX-Instanz so konfigurieren, dass der Datenverkehr über die physischen Server verteilt wird.

Die folgende Abbildung veranschaulicht diese Topologie.




In diesem Beispiel befindet sich der NetScaler CPX-Container (172.17.0.4) zwischen dem Client und den physischen Servern, die als Proxy fungieren. Die Server DB1 (10.102.29.105) und DB2 (10.102.29.110) befinden sich außerhalb eines Docker-Hosts im Netzwerk. Die Clientanfrage stammt aus dem Internet und wird auf dem NetScaler CPX empfangen, der sie auf die beiden Server verteilt.

Um diese Kommunikation zwischen dem Client und den Servern über NetScaler CPX zu ermöglichen, müssen Sie zuerst die Portzuordnung konfigurieren, während Sie den NetScaler CPX-Container erstellen. Konfigurieren Sie dann die beiden Dienste im NetScaler CPX-Container so, dass sie die beiden Server darstellen. Konfigurieren Sie schließlich einen virtuellen Server mithilfe der NetScaler CPX-IP-Adresse und des nicht standardmäßigen zugeordneten HTTP-Ports 8080.

Beachten Sie, dass in der Beispielkonfiguration das 10.x.x.x-Netzwerk ein öffentliches Netzwerk bezeichnet.

Führen Sie zum Konfigurieren dieses Beispielszenarios den folgenden Befehl an der Linux-Shell-Eingabeaufforderung aus, während Sie den NetScaler CPX-Container erstellen:

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

Führen Sie dann die folgenden Befehle aus, indem Sie entweder die Funktion Jobs in NetScaler ADM oder mithilfe von NITRO-APIs verwenden:

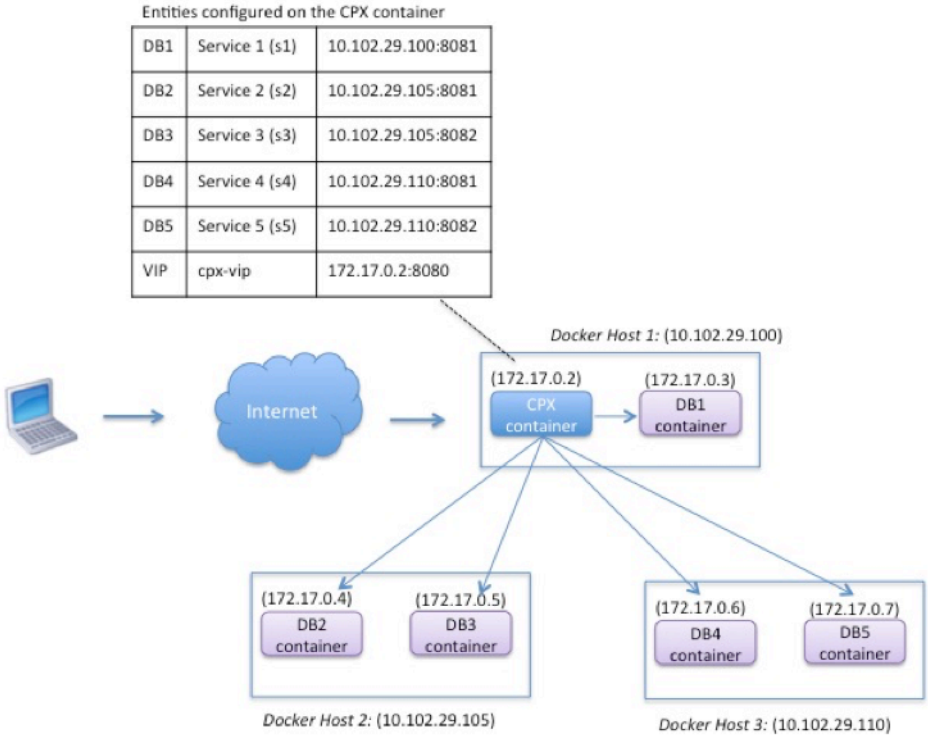
```
1     add service s1 HTTP 10.102.29.105 80
2     add service s2 HTTP 10.102.29.110 80
3     add lb vserver cpx-vip HTTP 172.17.0.4 8080
4     bind lb vserver cpx-vip s1
5     bind lb vserver cpx-vip s2
6 <!--NeedCopy-->
```

### Topologie 3: NetScaler CPX und Server, die auf verschiedenen Hosts bereitgestellt werden

In dieser Topologie werden die NetScaler CPX-Instanz und die Datenbankserver auf verschiedenen Docker-Hosts bereitgestellt, und der Clientverkehr stammt aus dem Internet. Diese Topologie kann in einer Produktionsbereitstellung verwendet werden, in der die NetScaler CPX-Instanz eine Reihe von Container-basierten Anwendungen oder Servern ausgleicht.

Das folgende Diagramm veranschaulicht diese Topologie.




In diesem Beispiel werden die NetScaler CPX-Instanz und ein Server (DB1) auf demselben Docker-Host mit der IP-Adresse 10.102.29.100 bereitgestellt. Vier weitere Server (DB2, DB3, DB4 und DB5) werden auf zwei verschiedenen Docker-Hosts bereitgestellt, 10.102.29.105 und 10.102.29.110.

Die aus dem Internet stammenden Clientanforderungen werden auf der NetScaler CPX-Instanz empfangen, die die Anforderungen dann auf die fünf Server verteilt. Um diese Kommunikation zu ermöglichen, müssen Sie Folgendes konfigurieren:

1. Legen Sie die Portzuordnung fest, während Sie Ihren NetScaler CPX-Container erstellen. In diesem Beispiel bedeutet dies, dass Sie Port 8080 des Containers an Port 8080 auf dem Host weiterleiten müssen. Wenn die Clientanforderung auf Port 8080 des Hosts eintrifft, wird sie Port 8080 des CPX-Containers zugeordnet.
2. Konfigurieren Sie die fünf Server als Dienste auf der NetScaler CPX-Instanz. Sie müssen eine Kombination aus der jeweiligen Docker-Host-IP-Adresse und dem zugeordneten Port verwenden, um diese Dienste festzulegen.
3. Konfigurieren Sie einen VIP auf der NetScaler CPX-Instanz, um die Clientanfrage zu empfangen. Dieser VIP sollte durch die NetScaler CPX-IP-Adresse und den Port 8080 dargestellt werden, der Port 8080 des Hosts zugeordnet wurde.
4. Binden Sie die Dienste abschließend an den VIP.

Beachten Sie, dass in der Beispielkonfiguration das 10.x.x.x-Netzwerk ein öffentliches Netzwerk bezeichnet.

Führen Sie zum Konfigurieren dieses Beispielszenarios den folgenden Befehl an der Linux-Shell-Eingabeaufforderung aus, während Sie den NetScaler CPX-Container erstellen:

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

Führen Sie die folgenden Befehle aus, indem Sie entweder die Funktion Jobs in NetScaler ADM oder mithilfe von NITRO-APIs verwenden:

```
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-->
```

## Stellen Sie NetScaler CPX mit direktem Zugriff auf das Netzwerk bereit

November 23, 2023

Im Bridge-Netzwerkmodus können Sie die NetScaler CPX-Instanz für direkten Zugriff auf das Netzwerk konfigurieren. In diesem Szenario wird der eingehende Datenverkehr direkt auf der NetScaler CPX Virtual Server IP (VIP) empfangen.

Um diese Kommunikation zu ermöglichen, müssen Sie zunächst eine öffentliche IP-Adresse auf der Docker0-Brücke konfigurieren. Dann entfernen Sie die öffentliche IP-Adresse vom Netzwerkport eth0 und binden Sie den Netzwerkport an die Docker0-Brücke.

Konfigurieren Sie den Lastenausgleich, indem Sie die beiden Dienste hinzufügen und dann eine öffentliche Netzwerk-IP-Adresse als VIP auf der NetScaler CPX-Instanz konfigurieren. Die Kundenanfragen gehen direkt im VIP ein.

In der Beispielkonfiguration bezeichnet das 10.x.x.x-Netzwerk ein öffentliches Netzwerk.

Führen Sie zum Konfigurieren dieses Szenarios den folgenden Befehl an der Linux-Shell-Eingabeaufforderung aus:

```
    ip addr add 10.102.29.100/24 dev docker0;
    ip addr del 10.102.29.100/24 dev eth0;
    brctl addif docker0 eth0;
    ip route del default;
    ip route add default via 10.102.29.1 dev docker0
<!--NeedCopy-->
```

Führen Sie entweder mithilfe der Funktion Jobs in NetScaler ADM oder mithilfe von NITRO-APIs die folgenden Befehle aus:

```
    add service s1 172.17.0.8 http 80
    add service s2 172.17.0.9 http 80
    add lb vserver cpx-vip HTTP 10.102.29.102 80
    bind lb vserver cpx-vip s1
    bind lb vserver cpx-vip s2
<!--NeedCopy-->
```

## Konfigurieren von NetScaler CPX in Kubernetes mithilfe von ConfigMaps

November 23, 2023

In Kubernetes können Sie die NetScaler CPX-Instanz mithilfe von ConfigMaps konfigurieren. Mit ConfigMaps können Sie die NetScaler CPX-Instanz während des Starts der Instanz dynamisch konfigurieren.

Erstellen Sie eine `cpx.conf` Konfigurationsdatei, die NetScaler-spezifische Konfiguration und Bash-Shell-Befehle enthält, die Sie dynamisch auf der NetScaler CPX-Instanz ausführen möchten. Für die Struktur der Konfigurationsdatei sind zwei Typen von Tags erforderlich, `#NetScaler Commands` und `#Shell Commands`. Unter dem `#NetScaler Commands` Tag müssen Sie alle NetScaler-Befehle hinzufügen, um die NetScaler-spezifische Konfiguration auf der NetScaler CPX-Instanz zu konfigurieren. Unter dem `#Shell Commands` Tag müssen Sie die Shell-Befehle hinzufügen, die Sie auf der NetScaler CPX-Instanz ausführen möchten.

> **Wichtig:**
>
> - Die Tags können in der Konfigurationsdatei mehrfach wiederholt werden.
> - Die Konfigurationsdatei kann auch Kommentare enthalten. Füge vor Kommentaren ein “**#**” **-Zeichen**hinzu.
> - Bei den Markierungen wird die Groß-/Kleinschreibung nicht beachtet
> - Wenn es bei der Bereitstellung des NetScaler CPX-Containers mit der Konfigurationsdatei Ausfallszenarien gibt, werden die Fehler in der `ns.log` Datei protokolliert.
> - Wenn Sie nach dem Start der NetScaler CPX-Instanz die ConfigMap ändern, wird die aktual-

> isierte Konfiguration nur angewendet, wenn die NetScaler CPX-Instanz neu gestartet wird.

Im Folgenden finden Sie ein Beispiel für eine Konfigurationsdatei:

```
#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-->
```

Nachdem Sie die Konfigurationsdatei erstellt haben, müssen Sie mit dem `kubectl create configmap` Befehl eine ConfigMap aus der Konfigurationsdatei erstellen.

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

Im obigen Beispiel können Sie eine ConfigMap erstellen, die auf der Konfigurationsdatei cpx.conf `cpx-config` basiert. Sie können diese ConfigMap dann in der YAML-Datei verwenden, die zum Bereitstellen der NetScaler CPX-Instanz verwendet wird.

Sie können die erstellte ConfigMap mit dem `kubectl get configmap` Befehl anzeigen.
`root@node1:~/yaml# kubectl get configmap cpx-config -o yaml`

**Musterbeispiel:**

```
    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-->
```

Sie können die erstellte ConfigMap `cpx-config` in der YAML-Datei, die für die Bereitstellung der NetScaler CPX-Instanz verwendet wird, wie folgt angeben:

```
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
```

```
39      configMap:
40        name: cpx-config
41  <!--NeedCopy-->
```

Sobald die NetScaler CPX-Instanz bereitgestellt wurde und die in der ConfigMap angegebene Konfiguration startet, `cpx-config` wird sie auf die NetScaler CPX-Instanz angewendet.

## Bereitstellen von NetScaler CPXs als lokale DNS-Caches für Kubernetes-Knoten

November 23, 2023

Anwendungs-Pods in einem Kubernetes-Cluster sind für die Kommunikation mit anderen Anwendungspods auf DNS angewiesen. DNS-Anfragen von Anwendungen innerhalb eines Kubernetes-Clusters werden von Kubernetes DNS (kube-DNS) bearbeitet. Aufgrund der breiteren Einführung von Microservices-Architekturen steigen die DNS-Anforderungsraten innerhalb eines Kubernetes-Clusters. Infolgedessen ist Kubernetes DNS (Kube-DNS) überlastet. Jetzt können Sie NetScaler CPX als lokalen DNS-Cache auf jedem Kubernetes-Knoten bereitstellen und DNS-Anforderungen von Anwendungspods im Knoten an NetScaler CPX weiterleiten. Daher können Sie DNS-Anfragen schneller lösen und die Belastung von Kubernetes DNS erheblich reduzieren.

Um NetScaler CPXs bereitzustellen, wird eine Kubernetes DaemonSet-Entität verwendet, um NetScaler CPX-Pods auf jedem Knoten im Kubernetes-Cluster zu planen. Ein Kubernetes DaemonSet stellt sicher, dass sich auf jedem Kubernetes-Knoten im Cluster eine Instanz von NetScaler CPX befindet.
Damit Anwendungspods den Datenverkehr zu CPX-DNS-Pods leiten, müssen Sie einen Kubernetes-Dienst mit Endpunkten als NetScaler CPX-Pods erstellen. Die Cluster-IP dieses Dienstes wird als DNS-Endpunkt für die Anwendungs-Pods verwendet. Um sicherzustellen, dass die Anwendungspods die IP-Adresse des NetScaler CPX-Dienstclusters für die DNS-Auflösung verwenden, müssen Sie die Kubelet-Konfigurationsdatei auf jedem Knoten mit der NetScaler CPX-Dienstcluster-IP aktualisieren.

Die folgenden Umgebungsvariablen werden eingeführt, um die Bereitstellung von NetScaler CPX als NodeLocal DNS-Cache zu unterstützen:

- `KUBE_DNS_SVC_IP`: Gibt die Cluster-IP-Adresse des `kube-dns` Dienstes an, die ein obligatorisches Argument zum Auslösen der Konfiguration in einem NetScaler CPX-Pod ist. Der NetScaler CPX-Pod leitet DNS-Abfragen an diese IP-Adresse weiter, wenn die DNS-Abfrageantwort nicht im NetScaler CPX-Cache verfügbar ist.
- `CPX_DNS_SVC_IP`: Gibt die Cluster-IP-Adresse des NetScaler CPX-Dienstes an. Die `CPX_DNS_SVC_IP` Umgebungsvariable wird verwendet, um lokales DNS auf Knoten zu

konfigurieren. Wenn Sie diese Variable konfigurieren, wird eine `iptables` Regel hinzugefügt, um die DNS-Anforderungen, die von Anwendungspods stammen, an den lokalen NetScaler CPX-Pod innerhalb des Knotens weiterzuleiten.

- `NS_DNS_FORCE_TCP`: Diese Umgebungsvariable erzwingt die Verwendung von TCP für DNS-Anforderungen, auch wenn die Abfragen über UDP empfangen werden.
- `NS_DNS_EXT_RESLV_IP`: Gibt die IP-Adresse des externen Nameservers an, um die DNS-Anforderungen für eine bestimmte Domäne weiterzuleiten.
- `NS_DNS_MATCH_DOMAIN`: Gibt die Zeichenfolge der externen Domäne an, mit der abgeglichen werden soll, um die Abfragen an den externen Namenserver weiterzuleiten.

## Bereitstellen von NetScaler CPXs als DNS-Caches auf Knoten

Das Bereitstellen von NetScaler CPX als lokaler DNS-Cache für einen Kubernetes-Cluster umfasst die folgenden Aufgaben:

Auf dem Master-Knoten:

- Erstellen Sie einen Kubernetes-Dienst mit Endpunkten als NetScaler CPX-Pods
- Erstellen Sie eine ConfigMap zum Definieren von Umgebungsvariablen für NetScaler CPX-Pods
- Planen Sie NetScaler CPX-Pods auf jedem Knoten im Kubernetes-Cluster mithilfe eines Kubernetes DaemonSet.

Auf Worker-Knoten:

- Ändern Sie die Kubelet-Konfigurationsdatei mit der Cluster-IP-Adresse des NetScaler CPX-Dienstes, um DNS-Anforderungen an NetScaler CPXs weiterzuleiten.

### Konfiguration auf dem Kubernetes-Masterknoten

Führen Sie die folgenden Schritte auf dem Kubernetes-Masterknoten aus, um NetScaler CPX als lokalen DNS-Cache für Knoten bereitzustellen:

1. Erstellen Sie mithilfe der `cpx_dns_svc.yaml` Datei einen Dienst mit NetScaler CPX-Pods als Endpoints.

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

Die `cpx_dns_svc.yaml` Datei wird wie folgt bereitgestellt:

```
        apiVersion: v1
        kind: Service
        metadata:
```

```
4           name: cpx-dns-svc
5           labels:
6             app: cpxd
7         spec:
8           ports:
9           - protocol: UDP
10            port: 53
11            name: dns
12          - protocol: TCP
13            port: 53
14            name: dns-tcp
15          selector:
16            app: cpx-daemon
```

2. Holen Sie sich die IP-Adresse des NetScaler CPX-Dienstes.

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

3. Holen Sie sich die IP-Adresse des Kube-DNS-Dienstes.

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

4. Erstellen Sie eine ConfigMap zum Definieren von Umgebungsvariablen für NetScaler CPX-Pods. Diese Umgebungsvariablen werden verwendet, um IP-Adressen des NetScaler CPX-Dienstes und des Kube-DNS-Dienstes zu übergeben. In diesem Schritt `cpx-dns-cache` wird ein Beispiel für ConfigMap mit den Umgebungsvariablen erstellt, die als Daten (Schlüssel-Wert-Paare) in einer Datei angegeben sind.

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

Im Folgenden finden Sie eine Beispieldatei mit den Umgebungsvariablen als Schlüssel-Wert-Paare.

```
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
```

Im Folgenden finden Sie ein Beispiel für eine 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. Erstellen Sie ein Kubernetes DaemonSet für NetScaler CPX auf dem Masterknoten.

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

Die `cpx_daemonset.yaml` Datei wird wie folgt bereitgestellt:

```
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
        volumeMounts:
         - mountPath: /netns/default/
        name: test-vol
        ports:
        - containerPort: 53
   envFrom:
       - configMapRef:
       name: cpx-dns-cache
   securityContext:
   privileged: true
   allowPrivilegeEscalation: true
   capabilities:
    add: ["NET_ADMIN"]
   volumes:
      -  name: test-vol
      hostPath:
      path: /proc/1/ns
      type: Directory
```

**Konfiguration auf Worker-Knoten im Kubernetes-Cluster**

Führen Sie nach Abschluss der Konfiguration auf dem Master-Knoten den folgenden Schritt auf Worker-Knoten aus:

1. Ändern Sie die Kubelet-Konfigurationsdatei, sodass Anwendungspods die NetScaler CPX-Dienstcluster-IP für die DNS-Auflösung verwenden können, indem Sie einen der folgenden Schritte ausführen:

   - Befolgen Sie die Schritte unter Rekonfigurieren Sie das Kubelet eines Node neu und ändern Sie den `--cluster-dns` Argumentwert im folgenden Format.

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

     oder

   - Bearbeiten Sie die `/etc/systemd/system/kubelet.service.d/10-kubeadm.conf` Datei und ändern Sie das `--cluster-dns` Argument mit den folgenden Schritten.

     a) Bearbeiten Sie die Kubelet-Konfiguration und geben Sie die Cluster-IP-Adresse des NetScaler CPX-Dienstes und die `kube-dns` Dienst-IP-Adresse für das `--cluster-dns` Argument an.

     ```
     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) Laden Sie das Kubelet der Knoten mit den folgenden Befehlen neu:

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

## NetScaler CPX-Proxy auf Google Compute Engine bereitstellen

March 21, 2024

In diesem Bereitstellungshandbuch wird beschrieben, wie Sie NetScaler CPX mit Docker in Google Compute Engine (GCE) von Google Cloud mit NetScaler ADM bereitstellen können, das im Unternehmensnetzwerk ausgeführt wird. In dieser Bereitstellung gleicht NetScaler CPX, das auf GCE

installiert ist, zwei Back-End-Server aus, und NetScaler ADM bietet Lizenzierungs- und Analyselösungen.

NetScaler CPX ist ein Container-basierter Proxy, der die vollständige Layer-7-Funktionalität, SSL-Offload, mehrere Protokolle und NITRO-API unterstützt. NetScaler ADM bietet Verwaltungs-, Lizenzierungs- und Analyselösungen. Als Lizenzserver bietet NetScaler ADM Berechtigungen für NetScaler CPX-Instanzen, die lokal oder in der Cloud ausgeführt werden.

CPX und CPX Express sind dieselben Images. Wenn Sie das CPX-Image mit NetScaler ADM lizenzieren und installieren, wird das CPX-Image im Docker App Store (Version 11 oder 12) zu einer vollständigen CPX-Instanz. Ohne Lizenz wird das CPX-Image zu einer CPX Express-Instanz, die 20 Mbit/s und 250 SSL-Verbindungen unterstützt.

### Voraussetzungen

- 2 GB Arbeitsspeicher und 1 vCPU für NetScaler CPX
- Docker Open Source von GCE erhältlich
- NetScaler ADM läuft lokal mit Internet- oder VPN-Verbindung zu GCE

> **Hinweis**
>
> Informationen zum Bereitstellen von NetScaler ADM finden Sie unter Bereitstellen von NetScaler ADM.

### Konfigurationsschritte

Sie müssen die folgenden Schritte ausführen, um diese Bereitstellung zu konfigurieren.

1. Installieren Sie Docker auf einer GCE VM.
2. Konfigurieren Sie die Remote-API-Kommunikation mit der Docker-Instanz.
3. Installieren Sie das NetScaler CPX-Image.
4. Erstellen Sie eine CPX-Instanz.
5. Lizenzieren Sie NetScaler CPX über NetScaler ADM.
6. Konfigurieren Sie Load Balancing Services auf NetScaler CPX und überprüfen Sie die Konfiguration.
    a) Installieren Sie die NGINX-Webserver.
    b) Konfigurieren Sie NetScaler CPX für den Lastausgleich und überprüfen Sie die Lastverteilung auf beide Webdienste.

**Schritt 1: Docker auf einer GCE VM installieren**

Erstellen Sie in GCE eine Linux-Ubuntu-VM. Installieren Sie dann Docker mithilfe der im folgenden Beispiel gezeigten Befehle auf der VM:

```
$ 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
LyzxR2cxr2u/xvxwXCkSMe7F4KZAphD+1ws61FhnUMi/PERMYfTFuvPrCkq4gyBj
t3fFpZ2NR/fKW87QOeVcn1ivXl9id3MMs9KXJsg7QasT7mCsee2VIFsxrkFQ2jNp
D+JAERRn9Fj4ArHL5TbwkkFbZZvSi6fr5h2GbCAXIGhIXKnjjorPY/YDX6X8AaHO
W1zblWy/CFr6VFl963jrjJgag0G6tNtBZLrclZgWhOQpeZZ5Lbvz2ZA5CqRrfAVc
wPNW1fObFIRtqV6vuVluFOPCMAAnOnqR02w9t17iVQjO3oVN0mbQi9vjuExXh1Yo
ScVetiO6LSmlQfVEVRTqHLMgXyR/EMo7iQIcBBABCgAGBQJXSWBlAAoJEFyzYeVS
+w0QeH0QAI6btAfYwYPuAjfRUy9qlnPhZ+xt1rnwsUzsbmo8K3XTNh+l/R08nu0d
sczw30Q1wju28fh1N8ay223+69f0+yICaXqR18AbGgFGKX7vo0gfEVaxdItUN3eH
NydGFzmeOKbAlrxIMECnSTG/TkFVYO9Ntlv9vSN2BupmTagTRErxLZKnVsWRzp+X

-----END PGP PUBLIC KEY BLOCK-----

OK
+ sudo -E sh -c mkdir -p /etc/apt/sources.list.d
+ dpkg --print-architecture
+ sudo -E sh -c echo deb \\[arch=amd64\\] https://apt.dockerproject.org
   /repo ubuntu-yakkety main > /etc/apt/sources.list.d/docker.list
+ 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 ...
```

```
Unpacking docker-engine (17.05.0~ce-0~ubuntu-yakkety) ...
Setting up aufs-tools (1:3.2+20130722-1.1ubuntu1) ...
Processing triggers for ureadahead (0.100.0-19) ...
Setting up cgroupfs-mount (1.3) ...
Processing triggers for libc-bin (2.24-3ubuntu2) ...
Processing triggers for systemd (231-9ubuntu4) ...
Setting up libltdl7:amd64 (2.4.6-1) ...
Processing triggers for man-db (2.7.5-1) ...
Setting up docker-engine (17.05.0~ce-0~ubuntu-yakkety) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /lib/systemd/system/docker.socket.
Processing triggers for ureadahead (0.100.0-19) ...
Processing triggers for libc-bin (2.24-3ubuntu2) ...
Processing triggers for systemd (231-9ubuntu4) ...
+ sudo -E sh -c docker version
Client:
Version: 17.05.0-ce
API version: 1.29
Go version: go1.7.5
Git commit: 89658be
Built: Thu May 4 22:15:36 2017
OS/Arch: linux/amd64

Server:
Version: 17.05.0-ce
API version: 1.29 (minimum version 1.12)
Go version: go1.7.5
Git commit: 89658be
Built: Thu May 4 22:15:36 2017
OS/Arch: linux/amd64
Experimental: false

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-->
```

**Schritt 2: Konfigurieren der Remote-API-Kommunikation mit der Docker-Instanz**

Öffnen Sie den Port 4243 für die API-Kommunikation mit der Docker-Instanz. Dieser Port ist erforderlich, damit NetScaler ADM mit der Docker-Instanz kommunizieren kann.

```

 \*\*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)
Active: \*\*active (running)\*\* since Wed 2017-05-31 12:52:17 UTC; 2s
   ago
Docs: https://docs.docker.com
Main PID: 4133 (dockerd)
Tasks: 16 (limit: 4915)
Memory: 30.1M
CPU: 184ms
CGroup: /system.slice/docker.service
├─4133 /usr/bin/dockerd -H fd://
└─4137 docker-containerd -l unix:///var/run/docker/libcontainerd/docker
   -containerd.sock --metrics-interval=0 --start-timeout 2m -

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
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
```

```
May 31 12:52:17 docker-7 dockerd[4133]: time="2017-05-31T12
    :52:17.301681794Z" level=info msg="Loading containers: start."
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
May 31 12:52:17 docker-7 dockerd[4133]: time="2017-05-31T12
    :52:17.465011600Z" level=info msg="Loading containers: done."
May 31 12:52:17 docker-7 dockerd[4133]: time="2017-05-31T12
    :52:17.484747909Z" level=info msg="Daemon has completed
    initialization"
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
May 31 12:52:17 docker-7 systemd[1]: Started Docker Application
    Container Engine.
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"
May 31 12:52:17 docker-7 dockerd[4133]: time="2017-05-31T12
    :52:17.504061522Z" level=info msg="API listen on [::]:4243"
$

(external)$ \*\*curl 104.199.209.157:4243/version\*\*
{
 "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" }

(external)$

<!--NeedCopy-->
```

**Schritt 3: Installieren Sie NetScaler CPX Image**

Holen Sie sich das NetScaler CPX-Image aus dem Docker App Store. Der CPX Express und der CPX haben dasselbe Image. Wenn Sie jedoch das CPX-Image mit NetScaler ADM lizenzieren und installieren, wird das Image zu einer vollständigen CPX-Instanz mit einer Leistung von 1 Gbit/s. Ohne Lizenz wird das Image zu einer CPX Express-Instanz, die 20 Mbit/s und 250 SSL-Verbindungen unterstützt.

```
$ \*\*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-->
```

**Schritt 4: Erstellen einer NetScaler CPX-Instanz**

Installieren Sie das NetScaler CPX-Image auf dem Docker-Host. Öffnen Sie Ports für bestimmte Dienste, wie im folgenden Beispiel gezeigt, und geben Sie eine IP-Adresse für NetScaler ADM an:

```
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
$

```

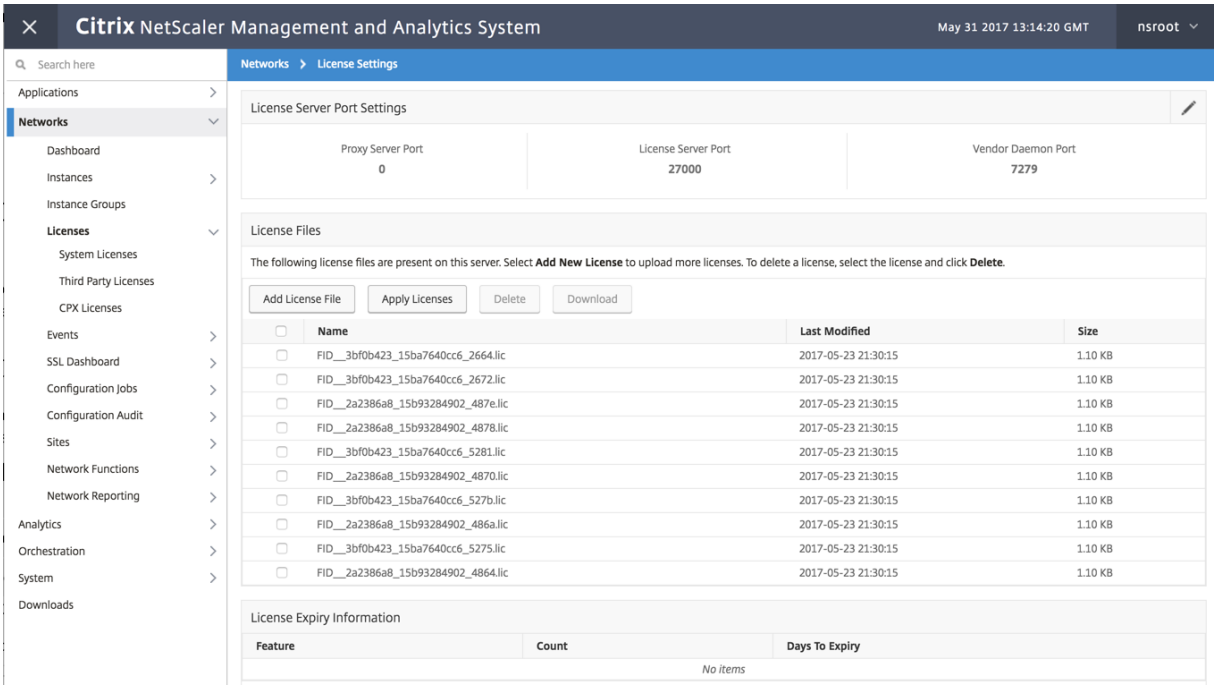
```
$ \*\*sudo docker ps\*\*
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
44ca1c6c0907 store/citrix/citrixadccpx:13.0-36.29 "/bin/sh -c 'bash ...
    " 19 seconds ago Up 17 seconds 0.0.0.0:5022->22/tcp,
    0.0.0.0:5080->80/tcp, 0.0.0.0:50000->88/tcp, 0.0.0.0:5163->161/udp,
    0.0.0.0:5443->443/tcp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

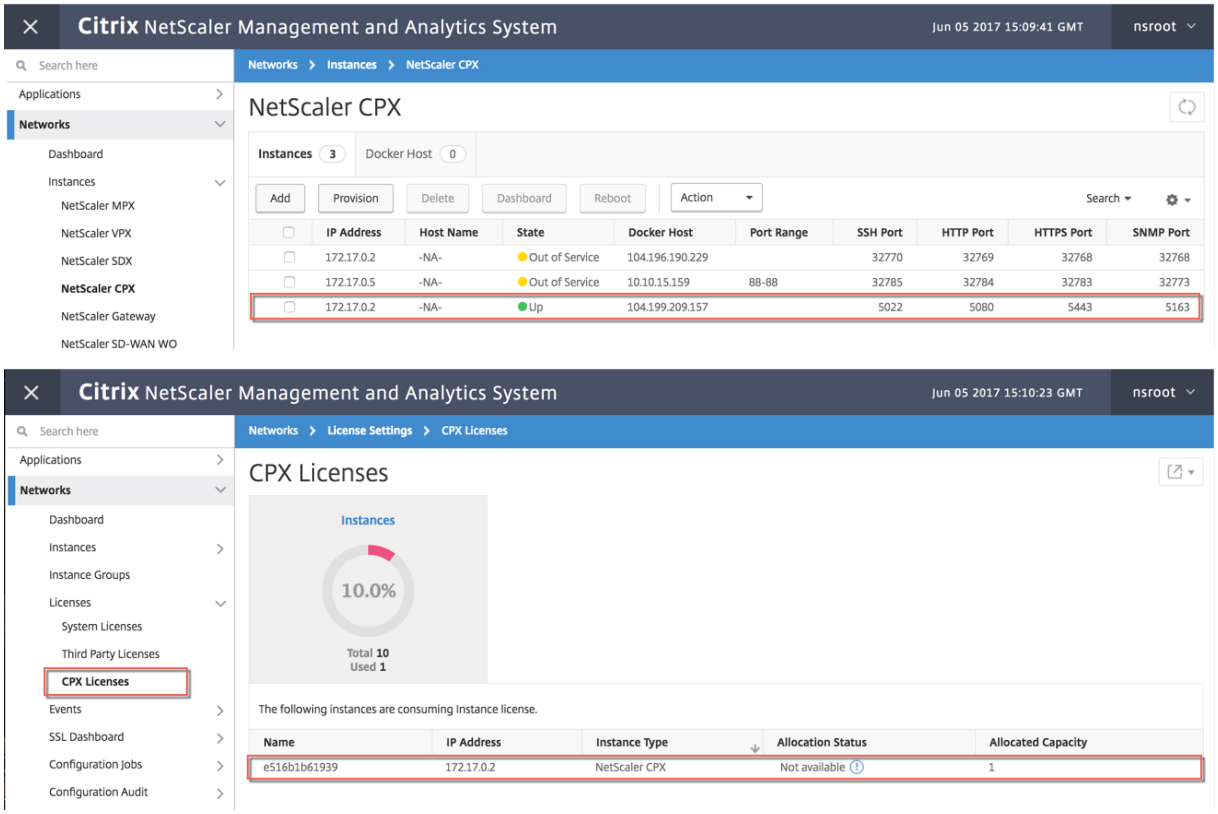
```
MASQUERADE tcp -- 172.17.0.2 172.17.0.2 tcp dpt:22

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-->
```

### Schritt 5: NetScaler CPX über NetScaler ADM lizenzieren

Unter der Annahme, dass NetScaler ADM lokal ausgeführt wird, sollten Sie überprüfen können, ob NetScaler CPX mit NetScaler ADM kommuniziert und Informationen sendet. Die folgenden Bilder zeigen, wie NetScaler CPX eine Lizenz von NetScaler ADM abruft.






**Schritt 6: Konfigurieren Sie Load Balancing Services auf NetScaler CPX und überprüfen Sie die Konfiguration**

Installieren Sie zunächst die NGINX-Webserver auf dem Docker-Host. Konfigurieren Sie anschließend den Lastenausgleich auf NetScaler CPX, um die beiden Webserver zu verteilen, und testen Sie die Konfiguration.

**Installieren Sie die NGINX-Webserver** Verwenden Sie die im folgenden Beispiel gezeigten Befehle, um NGINX-Webserver zu installieren.

```
$ 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-->
```

**Konfigurieren Sie NetScaler CPX für den Lastenausgleich und überprüfen Sie die Lastverteilung an beide Webdienste**

```
$ \*\*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\\>

\<p\>For online documentation and support please refer to
\\<a href="http://nginx.org/"\\>nginx.org\\</a\\>.\\<br/\\>
Commercial support is available at
\\<a href="http://nginx.com/"\\>nginx.com\\</a\\>.\\</p\\>

\<p\>\<em\>Thank you for using nginx.\</em\>\</p\>
\\</body\\>
\\</html\\>
(external)$


(external)$ for i in {
 1..100 }
  ; \*\*do curl http://104.199.209.157:50000 -o /dev/null ; done\*\*

  % Total    % Received % Xferd  Average Speed   Time    Time     Time
   Current

                                 Dload  Upload   Total   Spent    Left
   Speed

100   612  100   612    0     0   1767      0 --:--:-- --:--:--
   --:--:--  1768

```

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time
    Current
                                 Dload  Upload   Total   Spent    Left
    Speed
100   612  100   612    0     0   1893      0 --:--:-- --:--:--
   --:--:--  1894
  % Total    % Received % Xferd  Average Speed   Time    Time     Time
    Current
                                 Dload  Upload   Total   Spent    Left
    Speed
100   612  100   612    0     0   1884      0 --:--:-- --:--:--
   --:--:--  1883
  % Total    % Received % Xferd  Average Speed   Time    Time     Time
    Current
                                 Dload  Upload   Total   Spent    Left
    Speed
100   612  100   612    0     0   1917      0 --:--:-- --:--:--
   --:--:--  1924
  % Total    % Received % Xferd  Average Speed   Time    Time     Time
    Current
                                 Dload  Upload   Total   Spent    Left
    Speed
100   612  100   612    0     0   1877      0 --:--:-- --:--:--
   --:--:--  1883
  % Total    % Received % Xferd  Average Speed   Time    Time     Time
    Current
                                 Dload  Upload   Total   Spent    Left
    Speed
100   612  100   612    0     0   1852      0 --:--:-- --:--:--
   --:--:--  1848
  % Total    % Received % Xferd  Average Speed   Time    Time     Time
    Current
                                 Dload  Upload   Total   Spent    Left
    Speed
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
163   % Total    % Received % Xferd  Average Speed   Time    Time     Time
      Current
164
165                                  Dload  Upload   Total   Spent    Left
      Speed
166
167 100   612  100   612    0     0   1843      0 --:--:-- --:--:--
    --:--:--  1848
168
169
170
171   % Total    % Received % Xferd  Average Speed   Time    Time     Time
      Current
172
173                                  Dload  Upload   Total   Spent    Left
      Speed
174
175 100   612  100   612    0     0   1862      0 --:--:-- --:--:--
    --:--:--  1860
176
177   % Total    % Received % Xferd  Average Speed   Time    Time     Time
      Current
178
179                                  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

cpx-vip           172.17.0.2    88         HTTP           UP      100
           2


           inactSvcs

cpx-vip            0


Virtual Server Statistics

                                          Rate (/s)
   Total

Vserver hits                                    0
   101

Requests                                        0
   101

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-->
```

## NetScaler CPX Problembehandlung

November 23, 2023

In diesem Dokument wird erläutert, wie Sie Probleme beheben können, die bei der Verwendung von NetScaler CPX auftreten können. Mithilfe dieses Dokuments können Sie Protokolle sammeln, um die Ursachen zu ermitteln und Problemumgehungen für einige der häufigsten Probleme im Zusammenhang mit der Installation und Konfiguration von NetScaler CPX anzuwenden.

- Wie kann ich NetScaler CPX-Protokolle anzeigen?

  Sie können NetScaler CPX-Protokolle mit dem `kubectl logs` Befehl anzeigen, wenn NetScaler CPX mit der `tty:true` Option bereitgestellt wird. Sie können den folgenden Befehl ausführen, um die Protokolle anzuzeigen:

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

  Beispiel,

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

Im Folgenden finden Sie ein Beispiel für die NetScaler CPX-Pod-Bereitstellung mit der `tty: true` Option:

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

<!--NeedCopy-->
```

Weitere Startprotokolle finden Sie in der Datei */cpx/log/boot.log* des NetScaler CPX-Dateisystems.

**Hinweis**: Um den Pod-Namen abzurufen, führen Sie den `kubectl get pods -o wide` Befehl aus.

- Wie kann ich das technische Support-Paket von NetScaler CPX abholen?

  Sie können den folgenden Befehl auf der Shell-Schnittstelle des Kubernetes-Masterknotens ausführen, um das technische Supportpaket für NetScaler CPX zu sammeln:

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

  Sie können das technische Support-Paket im Verzeichnis */var/tmp/support* des Dateisystems des NetScaler CPX anzeigen. Verwenden Sie `scp` oder `kubectl cp`, um das technische Support-Paket von NetScaler CPX an das gewünschte Ziel zu kopieren.

  Beispiel:

```
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
```

- Warum steckt der NetScaler CPX-Pod beim Booten fest?

Sie können den Pod-Status mit dem `kubectl describe pods` Befehl überprüfen. Führen Sie den folgenden Befehl aus, um den Pod-Status zu ermitteln:

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

Beispiel:

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

Wenn die Pod-Ereignisse zeigen, dass der Container gestartet wurde, müssen Sie die Pod-Protokolle überprüfen.

- Wie kopiere ich Dateien zwischen dem NetScaler CPX-Pod und dem Kubernetes-Masterknoten?

  Es wird empfohlen, die Volume-Mount-Funktion von Docker zu verwenden, um das `/cpx` Verzeichnis in das Dateisystem des Hosts einzuhängen. Wenn ein NetScaler CPX-Container Core-Dumps beendet, sind Protokolle und andere wichtige Daten auf dem Mount Point verfügbar.

  Sie können einen der folgenden Befehle verwenden, um Dateien zwischen dem NetScaler CPX-Pod und dem Kubernetes-Masterknoten zu kopieren:

  **kubectl cp**: Sie können den folgenden Befehl ausführen, um Dateien vom Pod zum Knoten zu kopieren:

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

  Beispiel:

  ```
  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**: Sie können den Befehl verwenden, um Dateien zwischen dem NetScaler CPX-Pod und dem Kubernetes-Knoten zu kopieren. Führen Sie den folgenden Befehl aus, um Dateien vom Pod zum Knoten zu kopieren. Wenn es nach dem Kennwort fragt, geben Sie das Kennwort für den SSH-Benutzer ein:

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

  Beispiel:

  ```
  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
  ```

```
5   root@localhost:~#
```

- Wie erfasse ich Pakete auf NetScaler CPX?

  Um Pakete auf NetScaler CPX zu erfassen, starten Sie die Shell-Schnittstelle von NetScaler CPX mit dem `kubectl exec` Befehl. Führen Sie den folgenden Befehl aus, um die Shell-Schnittstelle des NetScaler CPX-Pods zu starten:

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

  Beispiel:

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

  Führen Sie außerdem den folgenden Befehl aus, um die Paketerfassung zu starten:

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

  Wenn Sie die laufende Paketerfassung beenden möchten, führen Sie den folgenden Befehl aus:

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

  Sie können die in einer *CAP-Datei* erfassten Pakete im Verzeichnis */cpx/nstrace/timestamp* im NetScaler CPX-Dateisystem anzeigen.

- Warum ist der Lizenzserver nicht konfiguriert, auch wenn NetScaler CPX mit der `LS_IP=<ADM-IP>` Umgebungsvariablen bereitgestellt wird?

  Stellen Sie sicher, dass auf den Lizenzserver von dem Knoten aus zugegriffen werden kann, auf dem NetScaler CPX bereitgestellt wird. Sie können den `ping <ADM-IP>` Befehl verwenden, um die Konnektivität vom NetScaler CPX-Knoten zu NetScaler ADM zu überprüfen.

  Wenn NetScaler ADM vom Knoten aus zugänglich ist, müssen Sie die Lizenzserverkonfigurationsprotokolle in der Datei */cpx/log/boot.log* überprüfen. Sie können auch mit dem folgenden Befehl auf der Shell-Oberfläche des NetScaler CPX-Pods nach der Lizenzserverkonfiguration suchen:

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

  Beispiel:

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

- Warum ist die gepoolte Lizenz auch nach einer erfolgreichen Lizenzserverkonfiguration auf NetScaler CPX nicht auf NetScaler CPX konfiguriert?

  Überprüfen Sie die Lizenzkonfigurationsprotokolle in der Datei */cpx/log/boot.log* . Sie können die konfigurierte gepoolte Lizenz auf NetScaler CPX auch mithilfe des folgenden Befehls auf der Shell-Schnittstelle des NetScaler CPX-Pods überprüfen:

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

  Beispiel,

  ```
  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
  ```

  Stellen Sie außerdem sicher, dass die erforderlichen Lizenzdateien in den Lizenzserver hochgeladen wurden. Sie können auch die verfügbaren Lizenzen auf dem Lizenzserver überprüfen, nachdem er erfolgreich auf NetScaler CPX konfiguriert wurde, indem Sie den folgenden Befehl verwenden. Führen Sie den Befehl auf der Shell-Schnittstelle des NetScaler CPX-Pods aus:

  ```
  cli_script.sh “ sh licenseserverpool ”
  ```

  Beispiel:

  ```
  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-->
  ```

- Warum erhalten NITRO-API-Aufrufe eine Antwort von NetScaler CPX auf *Verbindung verweigert* ?

  Der Standardport für NITRO-APIs ist 9080 (unsicher) und 9443 (sicher) ab NetScaler CPX Release 12.1. Stellen Sie sicher, dass der NITRO-Port von NetScaler CPX, auf den Sie zugreifen möchten, auf dem Pod verfügbar ist. Sie können den `kubectl describe` Befehl ausführen, um den

bereitgestellten und zugeordneten Port des NetScaler CPX-Containers im Abschnitt NetScaler CPX-Container anzuzeigen:

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

Beispiel:

```
      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-->
```

- Warum verbraucht der NSPPE-Prozess in NetScaler CPX den größten Teil der CPU-Auslastung, selbst wenn kein oder wenig Verkehr vorhanden ist?

  Wenn NetScaler CPX mit der `CPX_CONFIG='{ "YIELD":"NO" } '` Umgebungsvariablen bereitgestellt wird, verbraucht der NSPPE-Prozess 100 Prozent CPU-Auslastung, selbst wenn kein oder nur wenig Datenverkehr vorhanden ist. Wenn Sie möchten, dass der NSPPE-Prozess die CPU-Auslastung nicht verbraucht, müssen Sie NetScaler CPX ohne die `CPX_CONFIG='{ "YIELD":"NO" }` Umgebungsvariable bereitstellen. Standardmäßig ist der NSPPE-Prozess in CPX so konfiguriert, dass er die CPU-Auslastung nicht belastet oder verbraucht.

- Warum ist NetScaler CPX nicht in NetScaler ADM aufgeführt, selbst wenn es mit den erforderlichen Umgebungsvariablen für die Registrierung bei NetScaler ADM bereitgestellt wurde?

  Die Protokolle für die NetScaler CPX-Registrierung bei NetScaler ADM finden Sie in der Datei */cpx/log/boot.log* im NetScaler CPX-Dateisystem.

  Sie können die Zugänglichkeit der NetScaler ADM-IP-Adresse über den NetScaler CPX-Pod mit dem Befehl `ping` überprüfen. Stellen Sie außerdem sicher, dass alle erforderlichen Umgebungsvariablen für die NetScaler ADM-Registrierung für den NetScaler CPX-Container konfiguriert sind.

  - `NS_MGMT_SERVER`: Gibt die ADM-IP-Adresse oder den FQDN an.
  - `HOST`: Gibt die IP-Adresse des Knotens an.
  - `NS_HTTP_PORT`: Gibt den zugeordneten HTTP-Port auf dem Knoten an.
  - `NS_HTTPS_PORT`: Gibt den zugeordneten HTTPS-Port auf dem Knoten an.
  - `NS_SSH_PORT`: Gibt den zugeordneten SSH-Port auf dem Knoten an.
  - `NS_SNMP_PORT`: Gibt den zugeordneten SNMP-Port auf dem Knoten an.
  - `NS_ROUTABLE`: Die NetScaler CPX-Pod-IP-Adresse kann nicht von außen weitergeleitet werden.
  - `NS_MGMT_USER`: Gibt den ADM-Benutzernamen an.

- NS_MGMT_PASS: Gibt das ADM-Kennwort an.

- Warum wird die Fehlermeldung *Ungültiger Benutzername oder Kennwort* *cli_script.sh* angezeigt, nachdem das Kennwort für den nsroot-Benutzer geändert wurde?

  Der Befehl cli_script.sh ist ein Wrapper-Dienstprogramm für NSCLI auf NetScaler CPX. Es führt das erste Argument als Befehlszeichenfolge oder Dateipfad aus, und das zweite Argument ist optional, nämlich Anmeldeinformationen. Wenn das Kennwort für den nsroot-Benutzer geändert wird, müssen Sie Anmeldeinformationen als zweites Argument für angeben cli_script.sh. Sie können den folgenden Befehl ausführen, um NSCLI mit Anmeldeinformationen auszuführen:

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

  Beispiel:

  ```
  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
  ```

- Warum schlägt SSH zu NetScaler CPX mit root und nsroot Benutzer fehl?

  Ab Version 13.0-64.35 generiert NetScaler CPX ein Standardkennwort und aktualisiert es für SSH-Benutzer - root und nsroot. Wenn Sie das Kennwort nicht manuell geändert haben, finden Sie das Kennwort für SSH-Benutzer /var/deviceinfo/random_id im Dateisystem von NetScaler CPX.