Installation von Docker Combo in NX-OS Bash Shell

Inhalt

Einführung

Dieses Dokument beschreibt die Schritte zur Installation des Docker Compose-Pakets in der NX-OS Bash-Shell.

Die Geräte der Cisco Nexus Serien 3000 und 9000 unterstützen Docker-Funktionen in der Bash-Shell, beginnend mit NX-OS Version 9.2(1). Wie in der Docker Compose-Dokumentation beschrieben, "Compose ist ein Tool zum Definieren und Ausführen von Docker-Anwendungen für mehrere Container." Mit Docker Compose können Anwendungsentwickler alle Dienste definieren, die eine Anwendung in einer einzigen YAML-Datei mit dem Namen "docker-comse.yml" darstellen. Mit einem einzigen Befehl können alle diese Dienste erstellt, erstellt und gestartet werden. Darüber hinaus können alle Dienste über die Befehlssuite von Docker Compose beendet und überwacht werden.

Docker-Funktionen werden zwar nativ in der NX-OS Bash-Shell unterstützt, Docker Compose muss jedoch separat installiert werden.

Voraussetzungen

Anforderungen

In diesem Dokument muss die Bash-Shell auf Ihrem Cisco Nexus-Gerät aktiviert sein. Anweisungen zum Aktivieren der Bash-Shell finden Sie im Abschnitt "Accessing Bash" (Zugriff auf Bash) im Bash-Kapitel im Nexus NX-OS-Programmierbarkeitsleitfaden der Cisco Nexus Serie 9000.

Für dieses Dokument muss die Bash-Shell als DNS-Client konfiguriert werden, der Domänen-Hostnamen in IP-Adressen auflösen kann. Anweisungen zum Konfigurieren von DNS-Servern in der Bash-Shell finden Sie im Dokument.

Verwendete Komponenten

Die Informationen in diesem Dokument basieren auf den folgenden Software- und Hardwareversionen:

- Nexus 9000-Plattform ab NX-OS Version 9.2(1)
- Nexus 3000-Plattform ab NX-OS Version 9.2(1)

Die Informationen in diesem Dokument wurden von Geräten in einer bestimmten Laborumgebung erstellt. Alle in diesem Dokument verwendeten Geräte haben mit einer leeren (Standard-)Konfiguration begonnen. Wenn Ihr Netzwerk in Betrieb ist, stellen Sie sicher, dass Sie die potenziellen Auswirkungen eines Befehls verstehen.

Konfigurieren von HTTP-/HTTPS-Proxys

Wenn in Ihrer Umgebung ein HTTP- oder HTTPS-Proxy erforderlich ist, muss die Bash-Shell so konfiguriert werden, dass diese Proxys verwendet werden, bevor Docker Compose installiert werden kann.

Melden Sie sich über den Befehl `run bash sudo su -` als root-Benutzer bei der Bash-Shell an.

```
Nexus# run bash sudo su -
root@Nexus#whoami
root
```

Vorübergehende Konfiguration von HTTP-/HTTPS-Proxys

Um HTTP/HTTPS-Proxys für diese Sitzung vorübergehend zu konfigurieren, verwenden Sie den Befehl `export`, um die Umgebungsvariablen "http_proxy" und "https_proxy" zu definieren. Ein Beispiel hierfür ist unten dargestellt, wobei "proxy.example-domain.com" der Hostname eines hypothetischen Proxyservers ist.

```
root@Nexus#export http_proxy=http://proxy.example-domain.com:80/
root@Nexus#export https_proxy=https://proxy.example-domain.com:80/
```

Vergewissern Sie sich, dass die Umgebungsvariablen wie gewünscht mithilfe der Befehle `echo $http_proxy` und `echo $https_proxy` konfiguriert werden, wie unten gezeigt:

```
root@Nexus#echo $http_proxy
http://proxy.example-domain.com:80/ root@Nexus#echo $https_proxy
https://proxy.example-domain.com:80/
```

Die diesen Umgebungsvariablen zugewiesenen Werte werden gelöscht, wenn die Sitzung beendet wird, und müssen bei jeder Eingabe der Bash-Shell neu konfiguriert werden. Im folgenden Beispiel wird die Bash-Sitzung, in der die obige Konfiguration beendet ist, zurückgegeben und die Eingabeaufforderung an NX-OS zurückgegeben. Dann wird eine neue Sitzung zur Bash-Shell erstellt, in der die Umgebungsvariablen gelöscht wurden.

```
root@Nexus#export http_proxy=http://proxy.example-domain.com:80/
root@Nexus#export https_proxy=https://proxy.example-domain.com:80/
root@Nexus#echo $http_proxy
```

```
http://proxy.example-domain.com:80/ root@Nexus#echo $https_proxy
https://proxy.example-domain.com:80/ root@Nexus#exit
Nexus# run bash sudo su -
root@Nexus#echo $http_proxy

root@Nexus#echo $https_proxy

root@Nexus#
```

Permanente Konfiguration von HTTP-/HTTPS-Proxys

Um HTTP/HTTPS-Proxys für alle Sitzungen eines bestimmten Benutzers, der die Bash-Shell betritt, dauerhaft zu konfigurieren, müssen die Umgebungsvariablen "http_proxy" und "https_proxy" automatisch exportiert werden, wenn sich ein Benutzer anmeldet. Dies kann erreicht werden, indem `Exportbefehle` an die .bash_profile-Datei im Benutzerverzeichnis angefügt werden, die automatisch geladen wird, wenn sich der Benutzer bei der Bash-Shell anmeldet. Ein Beispiel hierfür ist unten dargestellt, wobei "proxy.example-domain.com" der Hostname eines hypothetischen Proxyservers ist.

```
root@Nexus#pwd
/root
root@Nexus#ls -al
total 28
drwxr-xr-x 5 root floppy 200 Dec 6 13:22 . drwxrwxr-t 62 root network-admin 1540 Nov 26 18:10 ..
-rw------- 1 root root 9486 Dec 6 13:22 .bash_history -rw-r--r-- 1 root floppy 703 Dec 6 13:22
.bash_profile drwx------ 3 root root 60 Nov 26 18:10 .config drwxr-xr-x 2 root root 60 Nov 26
18:11 .ncftp -rw------- 1 root root 0 Dec 5 14:37 .python-history -rw------- 1 root floppy 12
Nov 5 05:38 .rhosts drwxr-xr-x 2 root floppy 60 Nov 5 06:17 .ssh -rw------- 1 root root 5499 Dec
6 13:20 .viminfo root@Nexus#echo "export http_proxy=http://proxy.example-domain.com:80/" >>
.bash_profile
root@Nexus#echo "export https_proxy=https://proxy.example-domain.com:80/" >> .bash_profile
root@Nexus#cat .bash_profile | grep proxy
export http_proxy=http://proxy.example-domain.com:80/
export https_proxy=https://proxy.example-domain.com:80/
root@Nexus#exit
Nexus# run bash sudo su - root@Nexus#echo $http_proxy
http://proxy.example-domain.com:80/
root@Nexus#echo $https_proxy
https://proxy.example-domain.com:80/
```

Wenn Sie bestimmte HTTP/HTTPS-Proxys für alle Sitzungen für alle Benutzer konfigurieren möchten, die die Bash-Shell eingeben, fügen Sie diese `Exportbefehle` an die /etc/profile-Datei an. Bash lädt diese Datei automatisch zuerst, wenn sich ein Benutzer bei der Bash-Shell anmeldet. Daher werden alle Benutzer, die sich in der Bash-Shell anmelden, ihre HTTP-/HTTPS-Proxys entsprechend konfiguriert.

Ein Beispiel hierfür ist unten dargestellt, wobei "proxy.example-domain.com" der Hostname eines hypothetischen Proxyservers ist. Das Benutzerkonto "docker-admin" wird dann mit dem Bash Shelltype konfiguriert, der es dem Benutzerkonto ermöglicht, sich beim Remote-Zugriff auf das Gerät direkt in der Bash Shell anzumelden. Anschließend wird mithilfe von SSH der Zugriff auf die mgmt0-IP-Adresse (192.0.2.1) des Nexus-Geräts über die Management-VRF mit dem Benutzerkonto "docker-admin" ermöglicht. Das Beispiel zeigt, dass die Umgebungsvariablen "http_proxy" und "https_proxy" festgelegt wurden, auch wenn ein brandneues Benutzerkonto in der Bash-Shell angemeldet wurde.

```
root@Nexus#echo "export http_proxy=http://proxy.example-domain.com:80/" >> /etc/profile
```

```
root@Nexus#echo "export https_proxy=https://proxy.example-domain.com:80/" >> /etc/profile
root@Nexus#cat /etc/profile | grep proxy
export http_proxy=http://proxy.example-domain.com:80/ export https_proxy=https://proxy.example-
domain.com:80/ root@Nexus#exit
Nexus# run bash sudo su -
root@Nexus#echo $http_proxy
http://proxy.example-domain.com:80/ root@Nexus#echo $https_proxy
https://proxy.example-domain.com:80/ root@Nexus#exit
Nexus# configure terminal
Nexus(config)# username docker-admin role dev-ops password example_password
Nexus(config)# username docker-admin shelltype bash
Nexus(config)# exit
Nexus# ssh docker-admin@192.0.2.1 vrf management
Password: -bash-4.3$ whoami
docker-admin
-bash-4.3$ echo $http_proxy
http://proxy.example-domain.com:80/ -bash-4.3$ echo $https_proxy
https://proxy.example-domain.com:80/
```

Docker installieren

Um Docker Compose zu installieren, muss man das `wget`-Dienstprogramm verwenden, um die neueste Binärversion von Docker Compose herunterzuladen und diese Binärdatei dann im Verzeichnis /usr/bin abzulegen.

1. Bestimmen Sie die neueste stabile Version von Docker Compose, die mit den neuesten verfügbaren Versionen auf der Seite Docker Compose GitHub verfügbar ist. Suchen Sie die Versionsnummer für die neueste stabile Version am Anfang der Webseite. Zum Zeitpunkt dieser Veröffentlichung ist die letzte stabile Version 1.23.2.

2. Erstellen Sie die URL für die Docker Compose-Binärdatei, indem Sie `{latest-version}` im unten stehenden URL durch die Versionsnummer der neuesten stabilen Version ersetzen, die im vorherigen Schritt gefunden wurde:

https://github.com/docker/compose/releases/download/{latest-version}/docker-kompostieren-Linux-x86_64

Zum Beispiel lautet die URL für 1.23.2 zum Zeitpunkt dieser Veröffentlichung wie folgt: https://github.com/docker/compose/releases/download/1.23.2/docker-compose-Linux-x86_64

3. Geben Sie die Bash-Shell als Root von der NX-OS-Eingabeaufforderung mit dem Befehl `run bash sudo su -` ein, wie unten gezeigt:

```
Nexus# run bash sudo su -
root@Nexus#whoami
root
```

4. Ändern Sie ggf. den Netzwerknamespace-Kontext der Bash-Shell in einen Namespace mit DNS- und Internetverbindungen. Netzwerknamespaces sind logisch mit NX-OS-VRFs identisch. Im folgenden Beispiel wird veranschaulicht, wie auf den Management-Netzwerknamespace-Kontext umgeschaltet wird, der in dieser spezifischen Umgebung über DNS- und Internetverbindungen verfügt.

```
root@Nexus#ip netns exec management bash
```

```
root@Nexus#ping cisco.com -c 5
PING cisco.com (72.163.4.161) 56(84) bytes of data. 64 bytes from www1.cisco.com (72.163.4.161):
icmp_seq=1 ttl=239 time=29.2 ms 64 bytes from www1.cisco.com (72.163.4.161): icmp_seq=2 ttl=239
time=29.3 ms 64 bytes from www1.cisco.com (72.163.4.161): icmp_seq=3 ttl=239 time=29.3 ms 64
bytes from www1.cisco.com (72.163.4.161): icmp_seq=4 ttl=239 time=29.2 ms 64 bytes from
www1.cisco.com (72.163.4.161): icmp_seq=5 ttl=239 time=29.2 ms --- cisco.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4000ms rtt min/avg/max/mdev =
29.272/29.299/29.347/0.218 ms
```

5. Geben Sie den folgenden Befehl ein, und ersetzen Sie `{docker-url}` durch den im vorherigen Schritt erstellten URL: `wget {docker-url} -O /usr/bin/docker-comse`. Ein Beispiel für die Ausführung dieses Befehls ist unten dargestellt. Sie verwenden https://github.com/docker/compose/releases/download/1.23.2/docker-compose-Linux-x86_64 als Ersatz-URL für `{docker-url}`:

```
root@Nexus#wget https://github.com/docker/compose/releases/download/1.23.2/docker-compose-Linux-
x86_64 -O /usr/bin/docker-compose
--2018-12-06 15:24:36-- https://github.com/docker/compose/releases/download/1.23.2/docker-
compose-Linux-x86_64 Resolving proxy.example-domain.com... 2001:DB8::1, 192.0.2.100 Connecting
to proxy.example-domain.com|2001:DB8::1|:80... failed: Cannot assign requested address.
Connecting to proxy.example-domain.com|192.0.2.100|:80... connected. Proxy request sent,
awaiting response... 302 Found Location: https://github-production-release-asset-
2e65be.s3.amazonaws.com/15045751/67742200-f31f-11e8-947e-bd56efcd8886?X-Amz-Algorithm=AWS4-HMAC-
SHA256&X-Amz-Credential=AKIAIWNJYAX4CSVEH53A%2F20181206%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-
Date=20181206T152526Z&X-Amz-Expires=300&X-Amz-
Signature=dfccfd5a32a908040fd8c18694d6d912616f644e7ab3564c6b4ce314a0adbbc7&X-Amz-
SignedHeaders=host&actor_id=0&response-content-disposition=attachment%3B%20filename%3Ddocker-
compose-Linux-x86_64&response-content-type=application%2Foctet-stream [following] --2018-12-06
15:24:36-- https://github-production-release-asset-2e65be.s3.amazonaws.com/15045751/67742200-
f31f-11e8-947e-bd56efcd8886?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
Credential=AKIAIWNJYAX4CSVEH53A%2F20181206%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-
Date=20181206T152526Z&X-Amz-Expires=300&X-Amz-
Signature=dfccfd5a32a908040fd8c18694d6d912616f644e7ab3564c6b4ce314a0adbbc7&X-Amz-
SignedHeaders=host&actor_id=0&response-content-disposition=attachment%3B%20filename%3Ddocker-
compose-Linux-x86_64&response-content-type=application%2Foctet-stream Connecting to
proxy.example-domain.com|192.0.2.100|:80... connected. Proxy request sent, awaiting response...
200 OK Length: 11748168 (11M) [application/octet-stream] Saving to: ‚Äò/usr/bin/docker-
compose‚Äô /usr/bin/docker-compose
100%[=========================================================================================
=========================================>] 11.20M 6.44MB/s in 1.7s 2018-12-06 15:24:38 (6.44
MB/s) - ‚Äò/usr/bin/docker-compose‚Äô saved [11748168/11748168] root@Nexus#
```

6. Ändern Sie die Berechtigungen der Binärdatei /usr/bin/docker-compse so, dass sie mit dem Befehl `chmod +x /usr/bin/docker-compse` ausführbar ist. Dies ist im Folgenden dargestellt:

```
root@Nexus#docker-compose
bash: /usr/bin/docker-compose: Permission denied
root@Nexus#chmod +x /usr/bin/docker-compose
root@Nexus#docker-compose
Define and run multi-container applications with Docker. Usage: docker-compose [-f --help--file
FILE Specify an alternate compose file--project-name NAME Specify an alternate project
namedirectory--verbose Show more output--log-level LEVEL Set log level (DEBUG, INFO, WARNING,
ERROR, CRITICAL)--no-ansi Do not print ANSI control characters--version Print version and exit--
host HOST Daemon socket to connect to--tls Use TLS; implied by --tlsverify--tlscacert CA_PATH
Trust certs signed only by this CA--tlscert CLIENT_CERT_PATH Path to TLS certificate file--
tlskey TLS_KEY_PATH Path to TLS key file--tlsverify Use TLS and verify the remote--skip-
hostname-check Don't check the daemon's hostname against theinthe--project-directory PATH
Specify an alternate working directorytheofthefile--compatibility If set, Compose will attempt
to convert
deploykeysinfilestoorafromthefileandthefilecreateandandtimefromacommandinarunningcontaineronacom
mandkillfromtheforaaonecommandnumberofforastartstoptheandstartversiontheversion
```

Überprüfen der Dockervervollständigungsfunktion

Sie können überprüfen, ob Docker Compose erfolgreich installiert und funktionsfähig ist, indem Sie eine kleine docker-compse.yml-Datei erstellen und ausführen. Im folgenden Beispiel wird dieser Prozess schrittweise erläutert.

```
root@Nexus#mkdir docker-compose-example
root@Nexus#cd docker-compose-example/
root@Nexus#ls -al
total 0
drwxr-xr-x 2 root root    40 Dec  6 15:31 .
drwxr-xr-x 6 root floppy 260 Dec  6 15:31 ..
root@Nexus#vi docker-compose.yml
root@Nexus#cat docker-compose.yml
version: "3"
services:
  example_mongo:
    image: mongo:latest
    container_name: "example_mongo"
  example_alpine:
    image: alpine:latest
    container_name: "example_alpine"
root@Nexus#docker-compose up
Creating network "docker-compose-example_default" with the default driver
Pulling example_mongo (mongo:latest)...
latest: Pulling from library/mongo
7b8b6451c85f: Pull complete
ab4d1096d9ba: Pull complete
e6797d1788ac: Pull complete
e25c5c290bde: Pull complete
45aa1a4d5e06: Pull complete
b7e29f184242: Pull complete
ad78e42605af: Pull complete
1f4ac0b92a65: Pull complete
55880275f9fb: Pull complete
bd0396c9dcef: Pull complete
28bf9db38c03: Pull complete
3e954d14ae9b: Pull complete
cd245aa9c426: Pull complete
Creating example_mongo  ... done
Creating example_alpine ... done
Attaching to example_alpine, example_mongo
example_mongo     | 2018-12-06T15:36:18.710+0000 I CONTROL [main] Automatically disabling TLS
1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none' example_mongo | 2018-12-
06T15:36:18.717+0000 I CONTROL [initandlisten] MongoDB starting : pid=1 port=27017
dbpath=/data/db 64-bit host=c4f095f9adb0 example_mongo | 2018-12-06T15:36:18.717+0000 I CONTROL
[initandlisten] db version v4.0.4 example_mongo | 2018-12-06T15:36:18.717+0000 I CONTROL
[initandlisten] git version: f288a3bdf201007f3693c58e140056adf8b04839 example_mongo | 2018-12-
06T15:36:18.717+0000 I CONTROL [initandlisten] OpenSSL version: OpenSSL 1.0.2g 1 Mar 2016
example_mongo | 2018-12-06T15:36:18.717+0000 I CONTROL [initandlisten] allocator: tcmalloc
example_mongo | 2018-12-06T15:36:18.717+0000 I CONTROL [initandlisten] modules: none
example_mongo | 2018-12-06T15:36:18.717+0000 I CONTROL [initandlisten] build environment:
example_mongo | 2018-12-06T15:36:18.717+0000 I CONTROL [initandlisten] distmod: ubuntu1604
example_mongo | 2018-12-06T15:36:18.717+0000 I CONTROL [initandlisten] distarch: x86_64
example_mongo | 2018-12-06T15:36:18.717+0000 I CONTROL [initandlisten] target_arch: x86_64
example_mongo | 2018-12-06T15:36:18.717+0000 I CONTROL [initandlisten] options: { net: {
bindIpAll: true } } example_mongo | 2018-12-06T15:36:18.717+0000 I STORAGE [initandlisten]
example_mongo | 2018-12-06T15:36:18.717+0000 I STORAGE [initandlisten] ** WARNING: Using the XFS
filesystem is strongly recommended with the WiredTiger storage engine example_mongo | 2018-12-
06T15:36:18.717+0000 I STORAGE [initandlisten] ** See http://dochub.mongodb.org/core/prodnotes-
```

```
filesystem example_mongo | 2018-12-06T15:36:18.717+0000 I STORAGE [initandlisten]
wiredtiger_open config:
create,cache_size=31621M,session_max=20000,eviction=(threads_min=4,threads_max=4),config_base=fa
lse,statistics=(fast),log=(enabled=true,archive=true,path=journal,compressor=snappy),file_manage
r=(close_idle_time=100000),statistics_log=(wait=0),verbose=(recovery_progress), example_alpine
exited with code 0 example_mongo | 2018-12-06T15:36:19.722+0000 I STORAGE [initandlisten]
WiredTiger message [1544110579:722686][1:0x7f9d5de45a40], txn-recover: Set global recovery
timestamp: 0 example_mongo | 2018-12-06T15:36:19.745+0000 I RECOVERY [initandlisten] WiredTiger
recoveryTimestamp. Ts: Timestamp(0, 0) example_mongo | 2018-12-06T15:36:19.782+0000 I CONTROL
[initandlisten] example_mongo | 2018-12-06T15:36:19.782+0000 I CONTROL [initandlisten] **
WARNING: Access control is not enabled for the database. example_mongo | 2018-12-
06T15:36:19.782+0000 I CONTROL [initandlisten] ** Read and write access to data and
configuration is unrestricted. example_mongo | 2018-12-06T15:36:19.782+0000 I CONTROL
[initandlisten] example_mongo | 2018-12-06T15:36:19.783+0000 I STORAGE [initandlisten]
createCollection: admin.system.version with provided UUID: dc0b3249-576e-4546-9d97-de841f5c45c4
example_mongo | 2018-12-06T15:36:19.810+0000 I COMMAND [initandlisten] setting
featureCompatibilityVersion to 4.0 example_mongo | 2018-12-06T15:36:19.814+0000 I STORAGE
[initandlisten] createCollection: local.startup_log with generated UUID: 2f9820f5-11ad-480d-
a46c-c58222beb0ad example_mongo | 2018-12-06T15:36:19.841+0000 I FTDC [initandlisten]
Initializing full-time diagnostic data capture with directory '/data/db/diagnostic.data'
example_mongo | 2018-12-06T15:36:19.842+0000 I NETWORK [initandlisten] waiting for connections
on port 27017 example_mongo | 2018-12-06T15:36:19.842+0000 I STORAGE
[LogicalSessionCacheRefresh] createCollection: config.system.sessions with generated UUID:
d4aeac07-29fd-4208-9f83-394b4af648a2 example_mongo | 2018-12-06T15:36:19.885+0000 I INDEX
[LogicalSessionCacheRefresh] build index on: config.system.sessions properties: { v: 2, key: {
lastUse: 1 }, name: "lsidTTLIndex", ns: "config.system.sessions", expireAfterSeconds: 1800 }
example_mongo | 2018-12-06T15:36:19.885+0000 I INDEX [LogicalSessionCacheRefresh] building index
using bulk method; build may temporarily use up to 500 megabytes of RAM example_mongo | 2018-12-
06T15:36:19.886+0000 I INDEX [LogicalSessionCacheRefresh] build index done. scanned 0 total
records. 0 secs ^C
Gracefully stopping... (press Ctrl+C again to force)
Stopping example_mongo  ... done
root@Nexus#
```

Vorsicht: Stellen Sie sicher, dass der Befehl `docker-compse` ausgeführt wird, und zwar im Kontext eines Netzwerknamespaces, der über DNS- und Internetverbindungen verfügt. Andernfalls kann Docker Compose die angeforderten Bilder nicht aus dem Docker Hub abrufen.

Hinweis: Um eine Multi-Container-Docker-Anwendung herunterzufahren, die von Docker Compose gestartet und an die Docker Compose-Sitzung angeschlossen wurde, drücken Sie die Tastenkombination "Strg+C".

Zugehörige Informationen

- Docker Verfassen der Installationsdokumentation
- Docker - Dokumentationsübersicht verfassen
- NX-OS-Programmierhandbuch für die Cisco Nexus Serie 9000, Version 9.x
- Cisco Nexus NX-OS-Programmierhandbuch der Serie 9000, Version 7.x
- NX-OS-Programmierhandbuch für die Cisco Nexus Serie 9000, Version 6.x
- Cisco Nexus NX-OS-Programmierhandbuch der Serie 3000, Version 9.x
- Cisco Nexus 3000 NX-OS-Programmierhandbuch, Version 7.x
- Cisco Nexus NX-OS-Programmierhandbuch der Serie 3000, Version 6.x
- Cisco Nexus NX-OS-Programmierhandbuch der Serie 3500, Version 9.x

- Cisco Nexus 3500 NX-OS-Programmierhandbuch, Version 7.x
- Cisco Nexus NX-OS-Programmierhandbuch der Serie 3500, Version 6.x
- Cisco Nexus NX-OS-Programmierhandbuch der Serie 3600, Version 9.x
- Cisco Nexus 3600 NX-OS-Programmierhandbuch, Version 7.x
- Programmierbarkeit und Automatisierung mit Cisco Open NX-OS