Docker Compositie installeren in NX-OS Bash Shell

Inhoud

[Inleiding](#)
[Voorwaarden](#)
[Vereisten](#)
[Gebruikte componenten](#)
[HTTP/HTTPS-protocollen configureren](#)
[tijdelijk HTTP/HTTPS-proxy configureren](#)
[Permanent HTTP/HTTPS-proxy's configureren](#)
[Docker-compressie installeren](#)
[Functionaliteit van dokercompressie controleren](#)
[Gerelateerde informatie](#)

Inleiding

Dit document beschrijft de stappen die zijn gebruikt om het pakket Docker Compose te installeren in het schaal NX-OS Bash.

Cisco Nexus 3000 en 9000 Series apparaten ondersteunen Docker-functionaliteit binnen de Bash-schaal vanaf NX-OS release 9.2(1). Zoals beschreven in de documentatie van Docker Compose, is "Compose een middel om multicontainertoepassingen van Docker te definiëren en te gebruiken." Docker Compose stelt toepassingsontwikkelaars in staat alle services te definiëren die een toepassing vormen binnen één enkel YAML bestand dat "docker-compose.yml" wordt genoemd. Vervolgens kunnen met één opdracht al deze services gecreëerd, gebouwd en gestart worden. Bovendien kunnen alle services worden gestopt en gemonitord vanuit de Docker Compose reeks opdrachten.

Terwijl de functionaliteit van de Docker binnen de schaal van het NX-OS Bash als alternatief wordt ondersteund, moet Docker Compose afzonderlijk worden geïnstalleerd.

Voorwaarden

Vereisten

Dit document vereist dat de schelp van de basis op uw Cisco Nexus-apparaat is ingeschakeld. Raadpleeg het gedeelte "Toegang tot basis" van het hoofdstuk in het Cisco Nexus 9000 Series NX-OS programmeerbaarheidsgids voor instructies om het veld Bash in te schakelen.

Dit document vereist dat de schaal van de Bash als een DNS client wordt geconfigureerd die in staat is om domeinhostnamen aan IP-adressen op te lossen. Raadpleeg het document voor instructies om DNS-servers in het veld Bash te configureren.

Gebruikte componenten

De informatie in dit document is gebaseerd op de volgende software- en hardware-versies:

- Nexus 9000 platform vanaf NX-OS release 9.2(1)
- Nexus 3000 platform vanaf NX-OS release 9.2(1)

De informatie in dit document is gemaakt van apparatuur in een specifieke labomgeving. Alle apparaten die in dit document worden beschreven, hadden een opgeschoonde (standaard)configuratie. Als uw netwerk live is, moet u de potentiële impact van elke opdracht begrijpen.

HTTP/HTTPS-protocollen configureren

Als uw omgeving het gebruik van een HTTP- of HTTPS-proxy vereist, moet de Bash-shell zijn geconfigureerd om deze proxy's te gebruiken voordat Docker Compose kan worden geïnstalleerd.

Log in op de schaal van het Bash als basisgebruiker in via de `Start bash sudo -` opdracht.

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

tijdelijk HTTP/HTTPS-proxy configureren

Om tijdelijk HTTP/HTTPS proxy-proxy's voor deze sessie te configureren gebruikt u de opdracht `exporteren` om de "http_proxy" en "https_proxy" omgevingsvariabelen te definiëren. Een voorbeeld hiervan wordt hieronder getoond, waar "proxy.voorbeeld-domain.com" de hostname van een hypothetische proxy-server is.

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

Bevestig dat de omgevingsvariabelen naar wens zijn geconfigureerd met de opdrachten `echo $http_proxy` en `echo $https_proxy`, zoals hieronder wordt getoond:

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

De aan deze omgevingsvariabelen toegewezen waarden worden gewist wanneer de sessie wordt beëindigd en moeten telkens opnieuw worden geconfigureerd als de schaal van de Bash wordt ingevoerd. In het onderstaande voorbeeld: de Bash-sessie waar de bovenstaande configuratie is beëindigd, geeft de melding terug naar NX-OS. Vervolgens wordt een nieuwe sessie gemaakt aan de Bash shell, waar de omgevingsvariabelen zijn gewist.

```
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#
```

Permanent HTTP/HTTPS-proxy's configureren

Om HTTP/HTTPS proxy-proxy-proxy permanent te configureren voor alle sessies voor een specifieke gebruiker die de Bash shell ingaat, moeten de "http_proxy" en "https_proxy" omgevingsvariabelen automatisch worden geëxporteerd telkens wanneer een gebruiker inlogt. Dit kan worden bereikt door opdrachten voor `export toe` te voegen naar het .bash_profile bestand in de map van de gebruiker, dat Bash automatisch laadt wanneer de gebruiker zich inlogt in het veld Bash. Een voorbeeld hiervan wordt hieronder getoond, waar "proxy.voorbeeld-domain.com" de hostname van een hypothetische proxy-server is.

```
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/
```

Als je specifieke HTTP/HTTPS proxy-proxy's wilt configureren voor alle sessies voor alle gebruikers die de Bash shell binnendringen, voeg deze `export-`opdrachten toe aan het /etc/profile-bestand. Bash laadt dit bestand automatisch als een gebruiker zich in de schaal van het Bash-bestand inlogt - als resultaat hiervan worden alle gebruikers die zich in de schaal van het Bash aanmelden hun HTTP/HTTPS-proxy's dienovereenkomstig ingesteld.

Een voorbeeld hiervan wordt hieronder getoond, waar "proxy.voorbeeld-domain.com" de hostname van een hypothetische proxy-server is. De gebruikersaccount "docker-admin" wordt vervolgens ingesteld met de Bash-schelp, waarmee de gebruikersaccount direct in de Bash-schaal kan loggen bij toegang tot het apparaat op afstand. SSH wordt dan gebruikt om het GMA0 IP-adres (192.0.2.1) van het Nexus-apparaat te bereiken via het beheerVRF met behulp van de gebruikersaccount docker-beheerder. Het voorbeeld laat zien dat de "http_proxy" en "https_proxy" omgevingsvariabelen zijn ingesteld, zelfs wanneer er een gloednieuwe gebruikersaccount is ingelogd op de Bash shell.

```
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-compressie installeren

Om Docker Compose te installeren, moet men het watervoorziening gebruiken om de meest recente binaire release van Docker Compose te downloaden, dan plaats die binaire in de /usr/bin folder.

1. Bepaal de nieuwste stabiele versie van Docker Compose beschikbaar met de nieuwste beschikbare releases op de Docker Compose Hub-pagina. Zoek het versienummer voor de laatste stabiele release boven op de webpagina. Op het moment van schrijven is de laatste stabiele release 1.2.3.2.

2. Creëer de URL voor de Docker samenstelling binair door `{laatste-versie}` in de URL hieronder met het versienummer van de nieuwste stabiele release in de vorige stap te vervangen:

https://github.com/docker/compose/releases/download/ {laatste versie}/docker-compose-Linux-x86_64

Ten tijde van dit schrijven is de URL voor 1.23.2 bijvoorbeeld als volgt:
https://github.com/docker/compose/releases/download/1.23.2/docker-compose-Linux-x86_64

3. Voer de schelp van Bash in als root van de NX-OS-prompt met de `run bash sudo -` opdracht, zoals hieronder wordt aangetoond:

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

4. Indien nodig, verander de netwerk-naamruimte van de schaal van de Bash in een naamruimte met DNS en internetconnectiviteit. De netwerknaamruimtes zijn logisch identiek aan NX-OS VRF's. Het voorbeeld hieronder toont hoe te aan het netwerk van het beheer te switches naamruimte, dat DNS en Internet connectiviteit in deze specifieke omgeving heeft.

```
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. Voer het volgende bevel in, dat `{docker-url}` vervangt met de URL die in de vorige stap wordt gecreëerd: `Ophalen {docker-url} -O/usr/bin/docker-compose`. Een voorbeeld van deze opdracht het uitvoeren wordt hieronder getoond, https://github.com/docker/compose/releases/download/1.23.2/docker-compose-Linux-x86_64 als vervanging URL voor `{docker-url}` gebruikt:

```
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. Wijzig de machtigingen van het binaire bestand /usr/bin/docker-compose zodanig dat het uitvoerbaar is met behulp van de opdracht `chmod +x/usr/bin/docker-compose`. Dit wordt hieronder aangetoond:

```
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 deploykeysinfilestoorafromthefileandthefilecreateandandtimefromacommandinarunningcontaineronacommandkillfromtheforaaonecommandnumberofforastartstoptheandstartversiontheversion
```

Functionaliteit van dokercompressie controleren

U kunt controleren dat Docker Compose met succes is geïnstalleerd en functioneert door een klein docker-compose.yml-bestand te maken en uit te voeren. Het onderstaande voorbeeld gaat door dit proces.

```
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#
```

Voorzichtig: Zorg ervoor dat wanneer het bevel `docker-compose` wordt uitgevoerd, dit binnen de context van een netwerk naamruimte wordt gedaan dat DNS en Internet connectiviteit heeft. Anders kan Docker Compose niet gevraagde afbeeldingen van Docker Hub ophalen.

Opmerking: Druk op de toetsencombinatie "Ctrl+C" om een Docker-toepassing uit meerdere containers af te sluiten die is gestart door Docker Compose terwijl deze is aangesloten op de Docker Compose-sessie.

Gerelateerde informatie

- Installatie-documentatie voor documentsamenstelling
- Documentatie samenstellen document - Overzicht
- Cisco Nexus 9000 Series NX-OS programmeerbaarheidsgids, release 9.x
- Cisco Nexus 9000 Series NX-OS programmeerbaarheidsgids, release 7.x
- Cisco Nexus 9000 Series NX-OS programmeerbaarheidsgids, release 6.x
- Cisco Nexus 3000 Series netwerkmodule voor NX-OS programmeerbaarheid, release 9.x
- Cisco Nexus 3000 Series netwerkmodule voor NX-OS programmeerbaarheid, release 7.x
- Cisco Nexus 3000 Series netwerkmodule voor NX-OS programmeerbaarheidsgids, release 6.x
- Cisco Nexus 3500 Series netwerkmodule voor NX-OS programmeerbaarheid, release 9.x
- Cisco Nexus 3500 Series netwerkmodule voor NX-OS programmeerbaarheid, release 7.x
- Cisco Nexus 3500 Series netwerkmodule voor NX-OS programmeerbaarheid, release 6.x
- Cisco Nexus 3600 Series netwerkmodule voor NX-OS programmeerbaarheid, release 9.x
- Cisco Nexus 3600 Series NX-OS programmeerbaarheidsgids, release 7.x
- Programmeerbaarheid en automatisering met Cisco Open NX-OS