Instalar el diseño del acoplador en el shell Bash de NX-OS

Contenido

Introducción

Este documento describe los pasos utilizados para instalar el paquete Docker Compose dentro del shell NX-OS Bash.

Los dispositivos Nexus de Cisco serie 3000 y 9000 admiten la funcionalidad de Docker dentro del shell Bash a partir de NX-OS versión 9.2(1). Tal y como se describe en la documentación Redactar Docker, "Redactar es una herramienta para definir y ejecutar aplicaciones de Docker de varios contenedores". Docker Compose permite a los desarrolladores de aplicaciones definir todos los servicios que constituyen una aplicación dentro de un único archivo YAML denominado "docker-compose.yml". A continuación, con un único comando, se pueden crear, crear e iniciar todos esos servicios. Además, todos los servicios se pueden detener y monitorear desde el conjunto de comandos Docker Compose.

Mientras que la funcionalidad de Docker se admite de forma nativa en el shell Bash de NX-OS, el diseño de Docker debe instalarse por separado.

Prerequisites

Requirements

Este documento requiere que el shell Bash esté habilitado en su dispositivo Cisco Nexus. Refiérase a la sección "Acceso a Bash" del capítulo Bash de la Guía de Programación de Cisco Nexus serie 9000 NX-OS para obtener instrucciones para habilitar el shell Bash.

Este documento requiere que el shell Bash se configure como un cliente DNS capaz de resolver nombres de host de dominio a direcciones IP. Consulte el documento para obtener instrucciones para configurar los servidores DNS dentro del shell Bash.

Componentes Utilizados

La información que contiene este documento se basa en las siguientes versiones de software y hardware.

- Plataforma Nexus 9000 a partir de NX-OS versión 9.2(1)
- Plataforma Nexus 3000 a partir de NX-OS versión 9.2(1)

La información de este documento se originó a partir de dispositivos dentro de un ambiente de laboratorio específico. All of the devices used in this document started with a cleared (default) configuration. If your network is live, make sure that you understand the potential impact of any command.

Configuración de proxy HTTP/HTTPS

Si su entorno requiere el uso de un proxy HTTP o HTTPS, el shell Bash deberá configurarse para utilizar estos proxies antes de que pueda instalarse Docker Compose.

Inicie sesión en el shell Bash como el usuario root a través del comando `run bash sudo su -`.

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

Configuración temporal de proxy HTTP/HTTPS

Para configurar temporalmente los proxies HTTP/HTTPS para esta sesión, utilice el comando `export` para definir las variables de entorno "http_proxy" y "https_proxy". A continuación se muestra un ejemplo de esto, donde "proxy.example-domain.com" es el nombre de host de un servidor proxy hipotético.

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

Confirme que las variables de entorno se configuran como desee mediante los comandos `echo $http_proxy` y `echo $https_proxy`, como se muestra a continuación:

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

Los valores asignados a estas variables de entorno se borrarán cuando finalice la sesión y deberán reconfigurarse cada vez que se introduzca el shell Bash. En el siguiente ejemplo, la sesión Bash donde se encuentra la configuración anterior, devuelve el mensaje a NX-OS. Luego, se crea una nueva sesión al shell Bash, donde se han borrado las variables de entorno.

```
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#
```

Configuración permanente de proxy HTTP/HTTPS

Para configurar permanentemente los proxies HTTP/HTTPS para todas las sesiones de un usuario específico que ingresa en el shell Bash, las variables de entorno "http_proxy" y "https_proxy" se deben exportar automáticamente cada vez que cualquier usuario inicia sesión. Esto se puede lograr agregando comandos `export` al archivo .bash_profile ubicado en el directorio del usuario, que Bash se carga automáticamente cuando ese usuario inicia sesión en el shell Bash. A continuación se muestra un ejemplo de esto, donde "proxy.example-domain.com" es el nombre de host de un servidor proxy hipotético.

```
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/
```

Si desea configurar proxies HTTP/HTTPS específicos para todas las sesiones para todos los usuarios que ingresan al shell Bash, añada estos comandos `export` al archivo /etc/profile. Bash carga automáticamente este archivo primero cuando cualquier usuario inicia sesión en el shell de Bash; como resultado, todos los usuarios que inicien sesión en el shell de Bash tendrán sus proxies HTTP/HTTPS configurados en consecuencia.

A continuación se muestra un ejemplo de esto, donde "proxy.example-domain.com" es el nombre de host de un servidor proxy hipotético. A continuación, la cuenta de usuario "docker-admin" se configura con el tipo de identificación Bash, que permite que la cuenta de usuario inicie sesión directamente en el shell Bash cuando acceda de forma remota al dispositivo. SSH se utiliza entonces para acceder a la dirección IP mgmt0 (192.0.2.1) del dispositivo Nexus a través del VRF de administración usando la cuenta de usuario docker-admin. El ejemplo muestra que se han establecido las variables de entorno "http_proxy" y "https_proxy", incluso cuando se ha registrado una cuenta de usuario nueva en el shell de Bash.

```
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/
```

Instalación de la composición del soporte

Para instalar Docker Compose, se debe utilizar la utilidad `wget` para descargar la última versión binaria de Docker Compose y, a continuación, colocar ese binario en el directorio /usr/bin.

1. Determine la versión estable más reciente de Docker Compose disponible con las últimas versiones disponibles en la página Docker Compose GitHub. Busque el número de versión de la última versión estable en la parte superior de la página web. Al momento de escribir este artículo, la última versión estable es 1.23.2.

2. Cree la URL para el binario Docker Compose reemplazando `{última-versión}` en la URL siguiente con el número de versión de la última versión estable encontrada en el paso anterior:

https://github.com/docker/compose/releases/download/{última versión}/docker-compose-Linux-x86_64

Por ejemplo, la URL de 1.23.2 en el momento de escribir este artículo es la siguiente: https://github.com/docker/compose/releases/download/1.23.2/docker-compose-Linux-x86_64

3. Ingrese el shell Bash como root desde la indicación NX-OS con el comando `run bash sudo su -`, como se muestra a continuación:

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

4. Si es necesario, cambie el contexto de espacio de nombres de red del shell Bash a un espacio de nombres con DNS y conectividad a Internet. Los espacios de nombres de red son lógicamente idénticos a los VRF de NX-OS. El siguiente ejemplo muestra cómo cambiar al contexto del espacio de nombres de la red de administración, que tiene conectividad DNS e Internet en este entorno específico.

```
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. Ingrese el siguiente comando, reemplazando `{docker-url}` por la URL creada en el paso anterior: `wget {docker-url} -O /usr/bin/docker-compose`. A continuación se muestra un ejemplo de ejecución de este comando, utilizando https://github.com/docker/compose/releases/download/1.23.2/docker-compose-Linux-x86_64 como URL de reemplazo para `{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. Modifique los permisos del archivo binario /usr/bin/docker-compose de modo que sea ejecutable mediante el `comando chmod +x /usr/bin/docker-compose`. Esto se demuestra a continuación:

```
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
```

Verificación de la Funcionalidad de Redacción de Docker

Se puede verificar que Docker Compose se ha instalado correctamente y funciona creando y ejecutando un archivo docker-compose.yml pequeño. El siguiente ejemplo describe los pasos de este proceso.

```
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#
```

Precaución: Asegúrese de que cuando se ejecute el comando `docker-compose`, se realice dentro del contexto de un espacio de nombres de red que tenga DNS y conectividad a Internet. De lo contrario, Docker Compose no podrá extraer las imágenes solicitadas del Docker Hub.

Nota: Para desactivar una aplicación Docker de varios contenedores iniciada por Docker Compose mientras se conecta a la sesión Redactar acoplador, pulse la combinación de teclas "Ctrl+C".

Información Relacionada

- Documentación de la instalación del dispositivo Docker
- Descripción general de la documentación de composición de Docker
- Guía de programabilidad de Cisco Nexus serie 9000 NX-OS, versión 9.x
- Guía de programabilidad de Cisco Nexus serie 9000 NX-OS, versión 7.x
- Guía de programabilidad de Cisco Nexus serie 9000 NX-OS, versión 6.x
- Guía de programabilidad de Cisco Nexus serie 3000 NX-OS, versión 9.x
- Guía de programabilidad de Cisco Nexus serie 3000 NX-OS, versión 7.x
- Guía de programabilidad de Cisco Nexus serie 3000 NX-OS, versión 6.x
- Guía de programabilidad de Cisco Nexus serie 3500 NX-OS, versión 9.x
- Guía de programabilidad de Cisco Nexus serie 3500 NX-OS, versión 7.x
- Guía de programabilidad de Cisco Nexus serie 3500 NX-OS, versión 6.x
- Guía de programabilidad de Cisco Nexus serie 3600 NX-OS, versión 9.x
- Guía de programabilidad de Cisco Nexus serie 3600 NX-OS, versión 7.x
- Capacidad de programación y automatización con Cisco Open NX-OS