



適用於**AI**部署的高效能工作範例**ONTAP** NetApp Solutions

NetApp
April 12, 2024

This PDF was generated from https://docs.netapp.com/zh-tw/netapp-solutions/ai/aicp_example_high-performance_jobs_for_ontap_ai_deployments_overview.html on April 12, 2024. Always check docs.netapp.com for the latest.

目錄

- 適用於AI部署的高效能工作範例ONTAP 1
 - 適用於AI部署的高效能工作範例ONTAP 1
 - 執行單節點AI工作負載 1
 - 執行同步分散式AI工作負載 5

適用於AI部署的高效能工作範例ONTAP

本節包括在ONTAP 將Kubernetes部署在AI Pod上時、可執行的各種高效能工作範例。

適用於AI部署的高效能工作範例ONTAP

本節包括在ONTAP 將Kubernetes部署在AI Pod上時、可執行的各種高效能工作範例。

執行單節點AI工作負載

若要在Kubernetes叢集中執行單節點AI和ML工作、請從部署跳接主機執行下列工作。有了Trident、您就能快速輕鬆地建立資料磁碟區、讓Kubernetes工作負載能夠存取可能含有PB資料的資料。若要從Kubernetes Pod中存取此類資料磁碟區、只需在Pod定義中指定一個PVc即可。此步驟為Kubernetes原生作業、不需要NetApp專業人員。



本節假設您已將您嘗試在Kubernetes叢集中執行的特定AI和ML工作負載（採用Docker容器格式）容器化。

1. 下列命令範例顯示使用ImageNet資料集的TensorFlow基準測試工作負載建立Kubernetes工作。如需ImageNet資料集的詳細資訊、請參閱 "[ImageNet網站](#)"。

此範例工作要求八個GPU、因此可在單一GPU工作節點上執行、該工作節點具備八個或更多GPU。此範例工作可在叢集中提交、而具有八個以上GPU的工作節點不存在、或目前正與其他工作負載一起使用。如果是、則工作會維持在擱置狀態、直到該工作者節點可供使用為止。

此外、為了將儲存頻寬最大化、包含所需訓練資料的磁碟區會在本工作所建立的Pod內掛載兩次。另外一個Volume也會掛載在Pod中。第二個磁碟區將用於儲存結果和指標。這些磁碟區會使用PVCS名稱在工作定義中參考。如需Kubernetes工作的詳細資訊、請參閱 "[Kubernetes官方文件](#)"。

此範例所建立的Pod中、會將「medium」值為「mory」的「emptyDir」磁碟區掛載到「開發/ shm」。Docker Container執行時間所自動建立的「/dev/shm」虛擬磁碟區的預設大小、有時可能不足以滿足TensorFlow的需求。如以下範例所示、掛載「emptyDir」磁碟區可提供足夠大的「/dev/shm」虛擬磁碟區。如有關「emptyDir」Volume的詳細資訊、請參閱 "[Kubernetes官方文件](#)"。

在此範例工作定義中所指定的單一容器、其「優先」值為「true」。此值表示容器有效擁有主機的root存取權。在這種情況下會使用此註釋、因為執行的特定工作負載需要root存取權。具體而言、工作負載執行的清除快取作業需要root存取權。是否需要這種「特殊權限：真」註解、取決於您執行的特定工作負載需求。

```
$ cat << EOF > ./netapp-tensorflow-single-imagenet.yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: netapp-tensorflow-single-imagenet
spec:
  backoffLimit: 5
  template:
    spec:
```

```

volumes:
- name: dshm
  emptyDir:
    medium: Memory
- name: testdata-iface1
  persistentVolumeClaim:
    claimName: pb-fg-all-iface1
- name: testdata-iface2
  persistentVolumeClaim:
    claimName: pb-fg-all-iface2
- name: results
  persistentVolumeClaim:
    claimName: tensorflow-results
containers:
- name: netapp-tensorflow-py2
  image: netapp/tensorflow-py2:19.03.0
  command: ["python", "/netapp/scripts/run.py", "--
dataset_dir=/mnt/mount_0/dataset/imagenet", "--dgx_version=dgx1", "--
num_devices=8"]
  resources:
    limits:
      nvidia.com/gpu: 8
  volumeMounts:
    - mountPath: /dev/shm
      name: dshm
    - mountPath: /mnt/mount_0
      name: testdata-iface1
    - mountPath: /mnt/mount_1
      name: testdata-iface2
    - mountPath: /tmp
      name: results
  securityContext:
    privileged: true
  restartPolicy: Never
EOF
$ kubectl create -f ./netapp-tensorflow-single-imagenet.yaml
job.batch/netapp-tensorflow-single-imagenet created
$ kubectl get jobs
NAME                                COMPLETIONS   DURATION   AGE
netapp-tensorflow-single-imagenet   0/1            24s        24s

```

2. 確認您在步驟1中建立的工作正在正確執行。下列範例命令可確認已為工作建立單一Pod（如工作定義所指定）、而且此Pod目前正在其中一個GPU工作節點上執行。

```
$ kubectl get pods -o wide
```

| NAME | READY | STATUS |
|---|-------|---------|
| netapp-tensorflow-single-imagenet-m7x92 | 1/1 | Running |

```

RESTARTS    AGE
IP          NODE          NOMINATED NODE
3m         10.233.68.61   10.61.218.154   <none>

```

3. 確認您在步驟1中建立的工作已成功完成。下列命令範例可確認工作已成功完成。

```

$ kubectl get jobs
NAME                                     COMPLETIONS   DURATION
AGE
netapp-tensorflow-single-imagenet      1/1            5m42s
10m
$ kubectl get pods
NAME                                     READY   STATUS
RESTARTS   AGE
netapp-tensorflow-single-imagenet-m7x92 0/1     Completed
0         11m
$ kubectl logs netapp-tensorflow-single-imagenet-m7x92
[netapp-tensorflow-single-imagenet-m7x92:00008] PMIX ERROR: NO-
PERMISSIONS in file gds_dstore.c at line 702
[netapp-tensorflow-single-imagenet-m7x92:00008] PMIX ERROR: NO-
PERMISSIONS in file gds_dstore.c at line 711
Total images/sec = 6530.59125
===== Clean Cache !!! =====
mpirun -allow-run-as-root -np 1 -H localhost:1 bash -c 'sync; echo 1 >
/proc/sys/vm/drop_caches'
=====
mpirun -allow-run-as-root -np 8 -H localhost:8 -bind-to none -map-by
slot -x NCCL_DEBUG=INFO -x LD_LIBRARY_PATH -x PATH python
/netapp/tensorflow/benchmarks_190205/scripts/tf_cnn_benchmarks/tf_cnn_be
nchmarks.py --model=resnet50 --batch_size=256 --device=gpu
--force_gpu_compatible=True --num_intra_threads=1 --num_inter_threads=48
--variable_update=horovod --batch_group_size=20 --num_batches=500
--nodistortions --num_gpus=1 --data_format=NCHW --use_fp16=True
--use_tf_layers=False --data_name=imagenet --use_datasets=True
--data_dir=/mnt/mount_0/dataset/imagenet
--datasets_parallel_interleave_cycle_length=10
--datasets_sloppy_parallel_interleave=False --num_mounts=2
--mount_prefix=/mnt/mount_%d --datasets_prefetch_buffer_size=2000
--datasets_use_prefetch=True --datasets_num_private_threads=4
--horovod_device=gpu >
/tmp/20190814_105450_tensorflow_horovod_rdma_resnet50_gpu_8_256_b500_ima
genet_nodistort_fp16_r10_m2_nockpt.txt 2>&1

```

4. *選用：*清除工作成品。下列命令範例顯示刪除在步驟1中建立的工作物件。

刪除工作物件時、Kubernetes會自動刪除任何相關的Pod。

```

$ kubectl get jobs
NAME                                     COMPLETIONS   DURATION
AGE
netapp-tensorflow-single-imagenet      1/1            5m42s
10m
$ kubectl get pods
NAME                                     READY   STATUS
RESTARTS   AGE
netapp-tensorflow-single-imagenet-m7x92 0/1     Completed
0         11m
$ kubectl delete job netapp-tensorflow-single-imagenet
job.batch "netapp-tensorflow-single-imagenet" deleted
$ kubectl get jobs
No resources found.
$ kubectl get pods
No resources found.

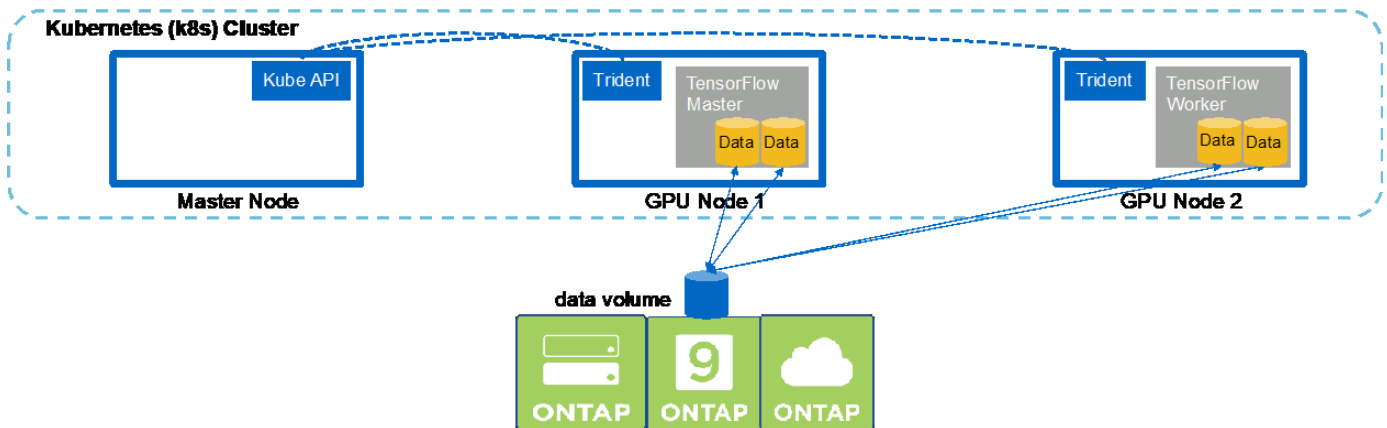
```

執行同步分散式AI工作負載

若要在Kubernetes叢集中執行同步多節點AI和ML工作、請在部署跨接主機上執行下列工作。此程序可讓您利用儲存在NetApp磁碟區上的資料、並使用比單一工作節點更多的GPU。如需同步分散式AI工作的說明、請參閱下圖。



相較於非同步分散式工作、同步分散式工作有助於提升效能和訓練準確度。關於同步工作與非同步工作的優缺點的討論、不在本文的討論範圍之內。



1. 下列命令範例顯示建立一個工作者、以參與本節範例中單一節點上執行的相同TensorFlow基準測試工作之同步分散式執行 "[執行單節點AI工作負載](#)"。在此特定範例中、只會部署一名員工、因為該工作會在兩個工作節點之間執行。

此範例的工作者部署要求八個GPU、因此可在單一GPU工作者節點上執行、該節點具備八個以上的GPU。如果GPU工作節點的GPU功能超過八個GPU、為了發揮最大效能、您可能想要增加此數目、使其等於工作節點所使用的GPU數量。如需Kubernetes部署的詳細資訊、請參閱 "[Kubernetes官方文件](#)"。

在此範例中會建立Kubernetes部署、因為這個特定的容器化工作者永遠不會自行完成。因此、使用Kubernetes工作架構來部署IT並不合理。如果您的員工是自行設計或撰寫完成、則使用工作架構來部署您的員工可能是合理的做法。

本範例部署規格中所指定的Pod、其「hostNetwork」值為「true」。此值表示Pod使用主機工作節點的網路堆疊、而非Kubernetes通常為每個Pod建立的虛擬網路堆疊。此註釋用於此案例、因為特定工作負載仰賴Open MPI、NCCL和Horovod以同步分散的方式執行工作負載。因此、它需要存取主機網路堆疊。關於Open MPI、NCCL和Horovod的討論不在本文的討論範圍之內。是否需要此「hostNetwork: true」註釋、取決於您執行的特定工作負載需求。如需有關「hostNetwork」欄位的詳細資訊、請參閱 ["Kubernetes官方文件"](#)。

```
$ cat << EOF > ./netapp-tensorflow-multi-imagenet-worker.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: netapp-tensorflow-multi-imagenet-worker
spec:
  replicas: 1
  selector:
    matchLabels:
      app: netapp-tensorflow-multi-imagenet-worker
  template:
    metadata:
      labels:
        app: netapp-tensorflow-multi-imagenet-worker
    spec:
      hostNetwork: true
      volumes:
        - name: dshm
          emptyDir:
            medium: Memory
        - name: testdata-iface1
          persistentVolumeClaim:
            claimName: pb-fg-all-iface1
        - name: testdata-iface2
          persistentVolumeClaim:
            claimName: pb-fg-all-iface2
        - name: results
          persistentVolumeClaim:
            claimName: tensorflow-results
      containers:
        - name: netapp-tensorflow-py2
          image: netapp/tensorflow-py2:19.03.0
          command: ["bash", "/netapp/scripts/start-slave-multi.sh",
"22122"]
          resources:
            limits:
```



```

        nvidia.com/gpu: 8
    volumeMounts:
    - mountPath: /dev/shm
      name: dshm
    - mountPath: /mnt/mount_0
      name: testdata-iface1
    - mountPath: /mnt/mount_1
      name: testdata-iface2
    - mountPath: /tmp
      name: results
    securityContext:
      privileged: true
EOF
$ kubectl create -f ./netapp-tensorflow-multi-imagenet-worker.yaml
deployment.apps/netapp-tensorflow-multi-imagenet-worker created
$ kubectl get deployments
NAME                                                    DESIRED   CURRENT   UP-TO-DATE
AVAILABLE   AGE
netapp-tensorflow-multi-imagenet-worker  1           1           1
1           4s

```

2. 確認您在步驟1中建立的工作者部署已成功啟動。下列命令範例可確認已針對部署建立單一工作者Pod、如部署定義所示、而且此Pod目前正在其中一個GPU工作者節點上執行。

```

$ kubectl get pods -o wide
NAME                                                    READY
STATUS    RESTARTS   AGE
IP          NODE          NOMINATED NODE
netapp-tensorflow-multi-imagenet-worker-654fc7f486-v6725  1/1
Running    0           60s    10.61.218.154    10.61.218.154    <none>
$ kubectl logs netapp-tensorflow-multi-imagenet-worker-654fc7f486-v6725
22122

```

3. 為啟動、參與及追蹤同步多節點工作執行的主節點建立Kubernetes工作。下列命令範例可建立一個主磁片、用於啟動、參與及追蹤同一個TensorFlow基準測試工作的同步分散式執行、該工作是在本節範例的單一節點上執行 ["執行單節點AI工作負載"](#)。

此範例主要工作要求八個GPU、因此可在具有八個以上GPU的單一GPU工作節點上執行。如果GPU工作節點的GPU功能超過八個GPU、為了發揮最大效能、您可能想要增加此數目、使其等於工作節點所使用的GPU數量。

本範例工作定義中所指定的主Pod、其「主機網路」值為「真」、就如同在步驟1中給工作群組「主機網路」值「真」一樣。請參閱步驟1、瞭解為何需要此值的詳細資訊。

```

$ cat << EOF > ./netapp-tensorflow-multi-imagenet-master.yaml
apiVersion: batch/v1

```

```

kind: Job
metadata:
  name: netapp-tensorflow-multi-imagenet-master
spec:
  backoffLimit: 5
  template:
    spec:
      hostNetwork: true
      volumes:
      - name: dshm
        emptyDir:
          medium: Memory
      - name: testdata-iface1
        persistentVolumeClaim:
          claimName: pb-fg-all-iface1
      - name: testdata-iface2
        persistentVolumeClaim:
          claimName: pb-fg-all-iface2
      - name: results
        persistentVolumeClaim:
          claimName: tensorflow-results
    containers:
    - name: netapp-tensorflow-py2
      image: netapp/tensorflow-py2:19.03.0
      command: ["python", "/netapp/scripts/run.py", "--
dataset_dir=/mnt/mount_0/dataset/imagenet", "--port=22122", "--
num_devices=16", "--dgx_version=dgx1", "--
nodes=10.61.218.152,10.61.218.154"]
      resources:
        limits:
          nvidia.com/gpu: 8
      volumeMounts:
      - mountPath: /dev/shm
        name: dshm
      - mountPath: /mnt/mount_0
        name: testdata-iface1
      - mountPath: /mnt/mount_1
        name: testdata-iface2
      - mountPath: /tmp
        name: results
      securityContext:
        privileged: true
      restartPolicy: Never
EOF
$ kubectl create -f ./netapp-tensorflow-multi-imagenet-master.yaml
job.batch/netapp-tensorflow-multi-imagenet-master created

```

```
$ kubectl get jobs
```

| NAME | COMPLETIONS | DURATION | AGE |
|---|-------------|----------|-----|
| netapp-tensorflow-multi-imagenet-master | 0/1 | 25s | 25s |

4. 確認您在步驟3中建立的主要工作正在正確執行。下列範例命令可確認已為工作建立單一主Pod、如工作定義所示、而且此Pod目前正在其中一個GPU工作節點上執行。您也應該看到、您在步驟1中看到的工作者Pod仍在執行中、而且主要和工作者Pod正在不同的節點上執行。

```
$ kubectl get pods -o wide
```

| NAME | STATUS | RESTARTS | AGE | IP | NODE | NOMINATED NODE | READY |
|--|---------|----------|-----|---------------|---------------|----------------|-------|
| netapp-tensorflow-multi-imagenet-master-ppwwj | Running | 0 | 45s | 10.61.218.152 | 10.61.218.152 | <none> | 1/1 |
| netapp-tensorflow-multi-imagenet-worker-654fc7f486-v6725 | Running | 0 | 26m | 10.61.218.154 | 10.61.218.154 | <none> | 1/1 |

5. 確認您在步驟3中建立的主要工作已成功完成。下列命令範例可確認工作已成功完成。

```
$ kubectl get jobs
```

| NAME | COMPLETIONS | DURATION | AGE |
|---|-------------|----------|-------|
| netapp-tensorflow-multi-imagenet-master | 1/1 | 5m50s | 9m18s |

```
$ kubectl get pods
```

| NAME | STATUS | RESTARTS | AGE | READY |
|--|-----------|----------|-------|-------|
| netapp-tensorflow-multi-imagenet-master-ppwwj | Completed | 0 | 9m38s | 0/1 |
| netapp-tensorflow-multi-imagenet-worker-654fc7f486-v6725 | Running | 0 | 35m | 1/1 |

```
$ kubectl logs netapp-tensorflow-multi-imagenet-master-ppwwj
```

```
[10.61.218.152:00008] WARNING: local probe returned unhandled
```

```
shell:unknown assuming bash
```

```
rm: cannot remove '/lib': Is a directory
```

```
[10.61.218.154:00033] PMIX ERROR: NO-PERMISSIONS in file gds_dstore.c at line 702
```

```
[10.61.218.154:00033] PMIX ERROR: NO-PERMISSIONS in file gds_dstore.c at line 711
```

```
[10.61.218.152:00008] PMIX ERROR: NO-PERMISSIONS in file gds_dstore.c at line 702
```

```
[10.61.218.152:00008] PMIX ERROR: NO-PERMISSIONS in file gds_dstore.c at line 711
```

```
Total images/sec = 12881.33875
```

```
===== Clean Cache !!! =====
```

```
mpirun -allow-run-as-root -np 2 -H 10.61.218.152:1,10.61.218.154:1 -mca pml obl -mca btl ^openib -mca btl_tcp_if_include enp1s0f0 -mca
```

```

plm_rsh_agent ssh -mca plm_rsh_args "-p 22122" bash -c 'sync; echo 1 >
/proc/sys/vm/drop_caches'
=====
mpirun -allow-run-as-root -np 16 -H 10.61.218.152:8,10.61.218.154:8
-bind-to none -map-by slot -x NCCL_DEBUG=INFO -x LD_LIBRARY_PATH -x PATH
-mca pml ob1 -mca btl ^openib -mca btl_tcp_if_include enpls0f0 -x
NCCL_IB_HCA=mlx5 -x NCCL_NET_GDR_READ=1 -x NCCL_IB_SL=3 -x
NCCL_IB_GID_INDEX=3 -x
NCCL_SOCKET_IFNAME=enp5s0.3091,enp12s0.3092,enp132s0.3093,enp139s0.3094
-x NCCL_IB_CUDA_SUPPORT=1 -mca orte_base_help_aggregate 0 -mca
plm_rsh_agent ssh -mca plm_rsh_args "-p 22122" python
/netapp/tensorflow/benchmarks_190205/scripts/tf_cnn_benchmarks/tf_cnn_ben
chmarks.py --model=resnet50 --batch_size=256 --device=gpu
--force_gpu_compatible=True --num_intra_threads=1 --num_inter_threads=48
--variable_update=horovod --batch_group_size=20 --num_batches=500
--nodistortions --num_gpus=1 --data_format=NCHW --use_fp16=True
--use_tf_layers=False --data_name=imagenet --use_datasets=True
--data_dir=/mnt/mount_0/dataset/imagenet
--datasets_parallel_interleave_cycle_length=10
--datasets_sloppy_parallel_interleave=False --num_mounts=2
--mount_prefix=/mnt/mount_%d --datasets_prefetch_buffer_size=2000 --
datasets_use_prefetch=True --datasets_num_private_threads=4
--horovod_device=gpu >
/tmp/20190814_161609_tensorflow_horovod_rdma_resnet50_gpu_16_256_b500_im
agenet_nodistort_fp16_r10_m2_nockpt.txt 2>&1

```

6. 當您不再需要部署時、請刪除該員工部署。下列命令範例顯示刪除在步驟1中建立的工作者部署物件。

當您刪除工作者部署物件時、Kubernetes會自動刪除任何關聯的工作者Pod。

```

$ kubectl get deployments
NAME                                                    DESIRED   CURRENT   UP-TO-DATE
AVAILABLE      AGE
netapp-tensorflow-multi-imagenet-worker  1         1         1
1         43m
$ kubectl get pods
NAME                                                    READY
STATUS      RESTARTS   AGE
netapp-tensorflow-multi-imagenet-master-ppwwj        0/1
Completed    0         17m
netapp-tensorflow-multi-imagenet-worker-654fc7f486-v6725  1/1
Running      0         43m
$ kubectl delete deployment netapp-tensorflow-multi-imagenet-worker
deployment.extensions "netapp-tensorflow-multi-imagenet-worker" deleted
$ kubectl get deployments
No resources found.
$ kubectl get pods
NAME                                                    READY   STATUS
RESTARTS   AGE
netapp-tensorflow-multi-imagenet-master-ppwwj        0/1     Completed    0
18m

```

7. *選用：*清除主要工作成品。下列命令範例顯示刪除在步驟3中建立的主要工作物件。

刪除主工作物件時、Kubernetes會自動刪除任何相關的主Pod。

```

$ kubectl get jobs
NAME                                                    COMPLETIONS   DURATION   AGE
netapp-tensorflow-multi-imagenet-master  1/1            5m50s     19m
$ kubectl get pods
NAME                                                    READY   STATUS
RESTARTS   AGE
netapp-tensorflow-multi-imagenet-master-ppwwj        0/1     Completed    0
19m
$ kubectl delete job netapp-tensorflow-multi-imagenet-master
job.batch "netapp-tensorflow-multi-imagenet-master" deleted
$ kubectl get jobs
No resources found.
$ kubectl get pods
No resources found.

```

版權資訊

Copyright © 2024 NetApp, Inc. 版權所有。台灣印製。非經版權所有人事先書面同意，不得將本受版權保護文件的任何部分以任何形式或任何方法（圖形、電子或機械）重製，包括影印、錄影、錄音或儲存至電子檢索系統中。

由 NetApp 版權資料衍伸之軟體必須遵守下列授權和免責聲明：

此軟體以 NETAPP「原樣」提供，不含任何明示或暗示的擔保，包括但不限於有關適售性或特定目的適用性之擔保，特此聲明。於任何情況下，就任何已造成或基於任何理論上責任之直接性、間接性、附隨性、特殊性、懲罰性或衍生性損害（包括但不限於替代商品或服務之採購；使用、資料或利潤上的損失；或企業營運中斷），無論是在使用此軟體時以任何方式所產生的契約、嚴格責任或侵權行為（包括疏忽或其他）等方面，NetApp 概不負責，即使已被告知有前述損害存在之可能性亦然。

NetApp 保留隨時變本文所述之任何產品的權利，恕不另行通知。NetApp 不承擔因使用本文所述之產品而產生的責任或義務，除非明確經過 NetApp 書面同意。使用或購買此產品並不會在依據任何專利權、商標權或任何其他 NetApp 智慧財產權的情況下轉讓授權。

本手冊所述之產品受到一項（含）以上的美國專利、國外專利或申請中專利所保障。

有限權利說明：政府機關的使用、複製或公開揭露須受 DFARS 252.227-7013（2014 年 2 月）和 FAR 52.227-19（2007 年 12 月）中的「技術資料權利 - 非商業項目」條款 (b)(3) 小段所述之限制。

此處所含屬於商業產品和 / 或商業服務（如 FAR 2.101 所定義）的資料均為 NetApp, Inc. 所有。根據本協議提供的所有 NetApp 技術資料和電腦軟體皆屬於商業性質，並且完全由私人出資開發。美國政府對於該資料具有非專屬、非轉讓、非轉授權、全球性、有限且不可撤銷的使用權限，僅限於美國政府為傳輸此資料所訂合約所允許之範圍，並基於履行該合約之目的方可使用。除非本文另有規定，否則未經 NetApp Inc. 事前書面許可，不得逕行使用、揭露、重製、修改、履行或展示該資料。美國政府授予國防部之許可權利，僅適用於 DFARS 條款 252.227-7015(b)（2014 年 2 月）所述權利。

商標資訊

NETAPP、NETAPP 標誌及 <http://www.netapp.com/TM> 所列之標章均為 NetApp, Inc. 的商標。文中所涉及的所有其他公司或產品名稱，均為其各自所有者的商標，不得侵犯。