How to Deploy an IBM OpenPOWER Cluster for Deep Learning

An Automated Multiple-Node Deployment for TensorFlow

Version 1.0

Introduction

This solution consists of this document, a prescribed bill of materials (BoM), and a deployment configuration file. The BOM provides a description and representation of an IBM® PowerAI installation of TensorFlow across multiple OpenPOWER servers. It provides information such as model numbers and feature codes to simplify the ordering process. It also provides the racking and cabling rules for the preferred layout of the servers, switches, and cables.

Figure 1 illustrates a minimal deployment of the solution.

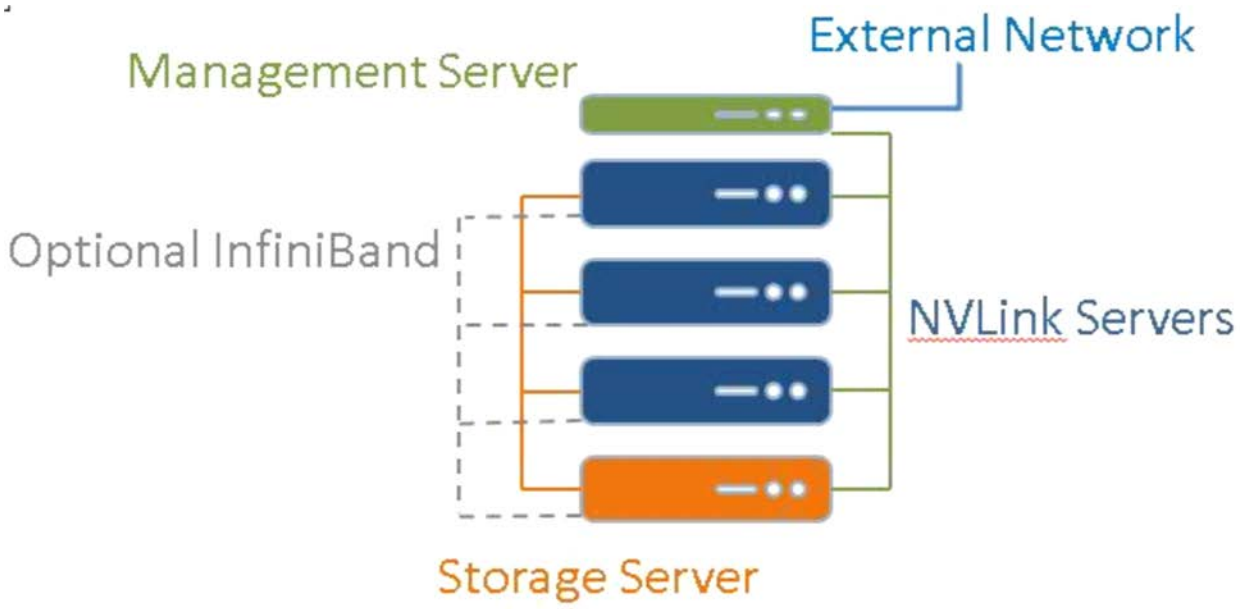

Figure 1. High-level diagram

The hardware consists of IBM POWER8® servers enhanced with NVIDIA® NVLink™ interconnects. There is an additional server for shared storage and an additional server for deployment and management tasks. In the design, there are dedicated networks for storage (**orange**) and management (**green**). The solution includes an option for an additional high-speed, low-latency InfiniBand network (**dashed grey**) for tensor

communication. An existing network (**blue**) with external routing capability is required for the management node. Optionally, it can be extended to the compute nodes.

The management server is used to manage the configuration and, optionally, to automatically deploy the entire solution. This deployment automation is accomplished by using the OpenPOWER Cluster Genesis (OPCG) tool, which uses the configuration file as input. It installs the nodes and subsequently installs and configures the needed software for the solution.

The deployment configuration file, which is included, provides a mapping of servers and switches to software for deployment. Software is mapped by assigning each server a software *role*. A TensorFlow training environment requires an entity to perform a parameter-serving role for updating and sharing parameter data. In addition to this parameter-server role, there are also storage-server and training-compute roles.

The parameter-server role and the storage-server role are assigned to the same physical server: the storage server. The training-compute role is assigned to the NVLink servers.

Compute node architecture

The POWER8 training-compute servers have a unique architecture that includes the high-bandwidth NVLink bus between each of two POWER8 CPUs and the pair of GPUs to which they are attached. The NVLink bus is NVIDIA's high-speed interconnect technology for GPU-accelerated computing. The NVLink bus provides an increased bandwidth to GPUs as well as a reduction in latency and code path lengths. *Figure 2* on page 3 shows the high-level architecture design of the compute-node servers.

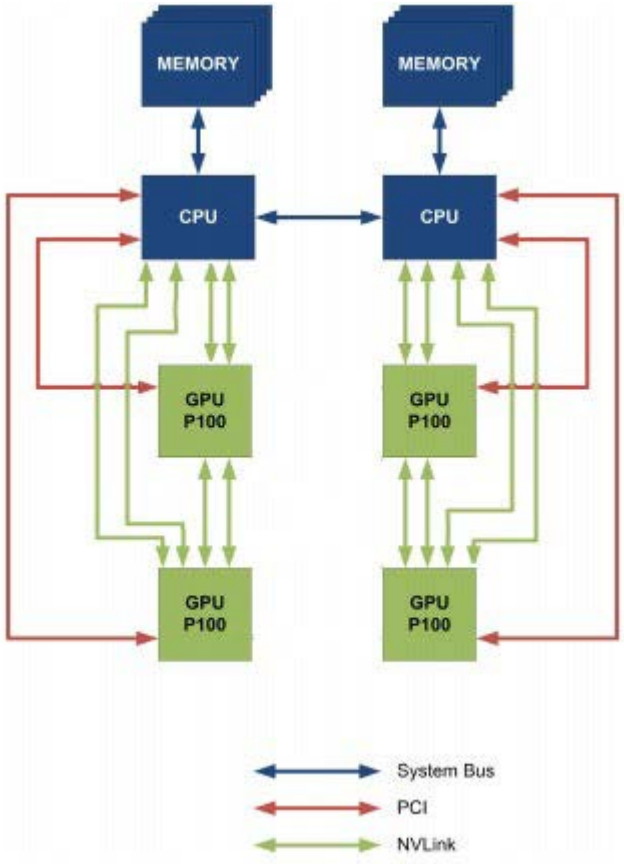

Figure 2. Compute node servers

The initialization of the GPUs is done through the PCIe interfaces shown in *Figure 2*. The PCIe interfaces also contain side-band communication for status, power management, and so on. However, after the GPU is up and running, all data communications is done using the NVLink bus.

High-level deployment steps

Note: Each steps is described in more detail in the sections that follow.

1	Acquire the hardware.
2	Rack and cable the hardware.
3	Choose the basic configuration parameters.
4	Prepare the management node.
5	Configure the cluster using the Cluster Genesis tool.
6	Complete the post-Cluster Genesis configuration.

Step 1: Acquire the hardware

Go to the following link to obtain the bill of materials, which lists the required parts.

https://github.com/open-power-ref-design/deep-learning/tree/master/docs

If you do not already have the needed parts, go to the following link to contact an IBM representative for ordering and purchasing assistance.

```
https://www.ibm.com/marketing/iwm/dre/signup?source=MAIL-power&disableCookie=Yes
```

Step 2: Rack and cable the hardware

This section provides the hardware details, port listings, and cable connection information needed to correctly cable the solution. See *Appendix A: Hardware orientation* on page 17 for specific information about each of the different nodes.

Racking the components

The racking rules specify where to place the servers and switches and where to connect the cables.

The suggested racking rules, shown in *Figure 3* on page *5*, focus on enabling:

- Rack modularity
- Consistency
- Expandability
- Ease of servicing, repurposing, shipping, and cooling

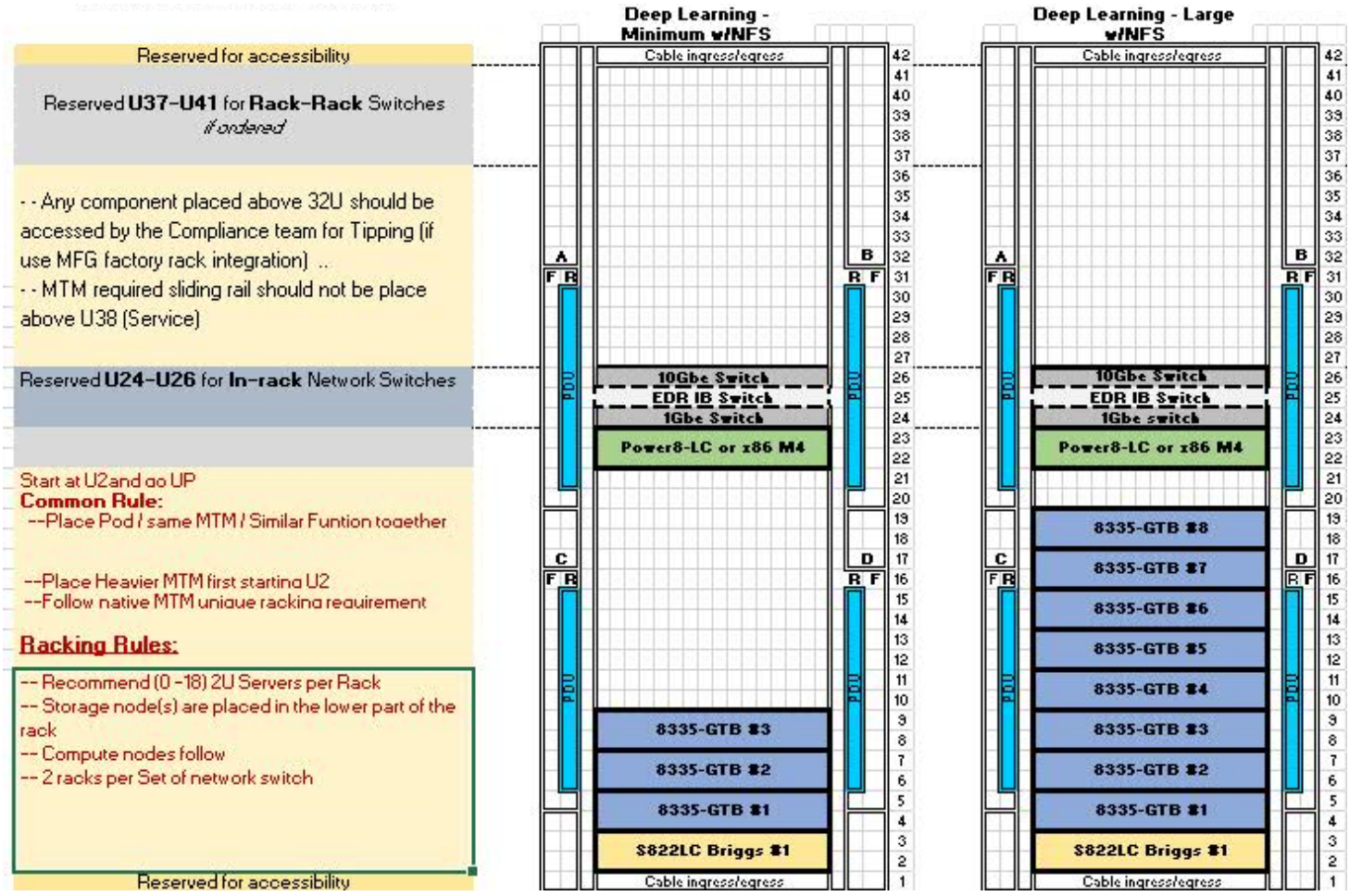

Figure 3. Suggested racking rules

Place the intra-rack (leaf) network switches in slots U24 - U26 as follows:

- Place the 10 GbE storage network switch in slot U26.
- Place the enhanced data rate (EDR) InfiniBand tensor network switch in slot U25.
- Place the 1 GbE management network switch in slot U24.

Place the inter-rack (spine) switches in slots U37 - U41.

If more than four power distribution units (PDUs) are required, horizontal PDUs can be placed in slots 40U and 41U. Spine switches take priority over additional PDUs. Therefore, if slots 40U and 41U are occupied by spine switches, place the horizontal PDUs in the next available slots.

Cable the components together

Figure 4 on page 6 shows a rear server view with labels for three compute nodes named **wkr1**, **wkr2**, and **wkr3**. It also shows the storage/parameter node named **stor**.

The service port enabled for the intelligent platform management interface (IPMI and the onboard 1 GbE network port on each node connect to the management network (green), while the 10 GbE port connects to the storage network (**orange**). The external network (dashed blue) can, optionally, be extended to each of the compute nodes if external connectivity is needed on each node after deployment. The management node, named **mgr** (not shown), requires this external connectivity during deployment.

The optional InfiniBand ports (dashed grey) connect to any available ports on the InfiniBand switch.

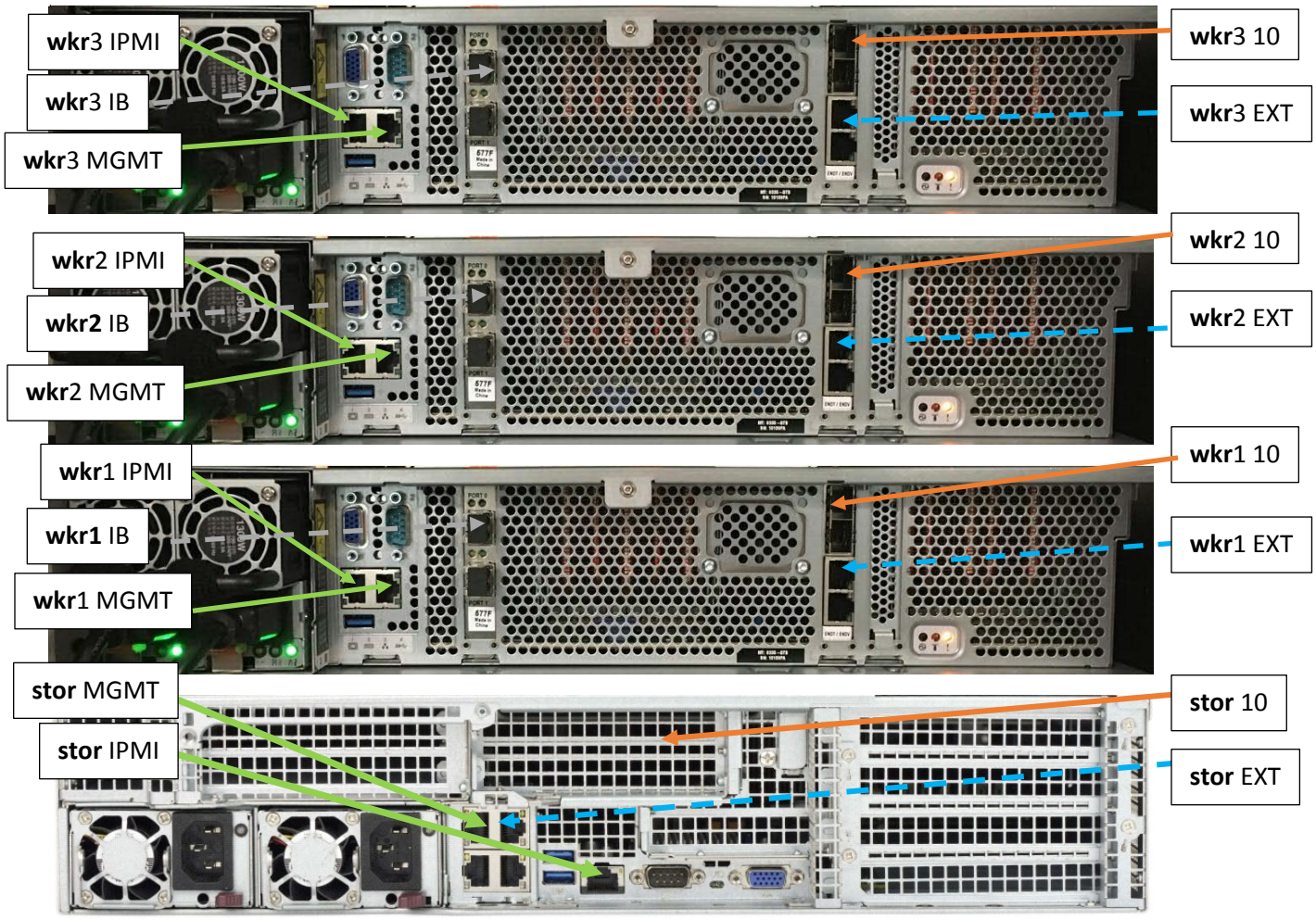

Figure 4. Compute nodes: rear view

Network switches

The Cluster Genesis automation tooling manages the included network switches. It uses the switch access to identify the media access control (MAC) addresses of the connected servers. The configuration file provides a mapping of systems to ports. The

physical cabling must reflect the port mapping that is in the configuration file. Currently, Cluster Genesis supports Lenovo switches in the automation. (For more information, see the BOM.)

Figure 5 shows the network switch view. The 1 GbE switch connects to both the IPMI and management ports on each server. The management node (**mgr**) is also connected to this switch on port 1. Port 1 is also designated as the switch management port and is given an address on a separate subnet. The server-to-port connections in *Figure 5* reflect what is defined in the default configuration file. This can be customized as needed.

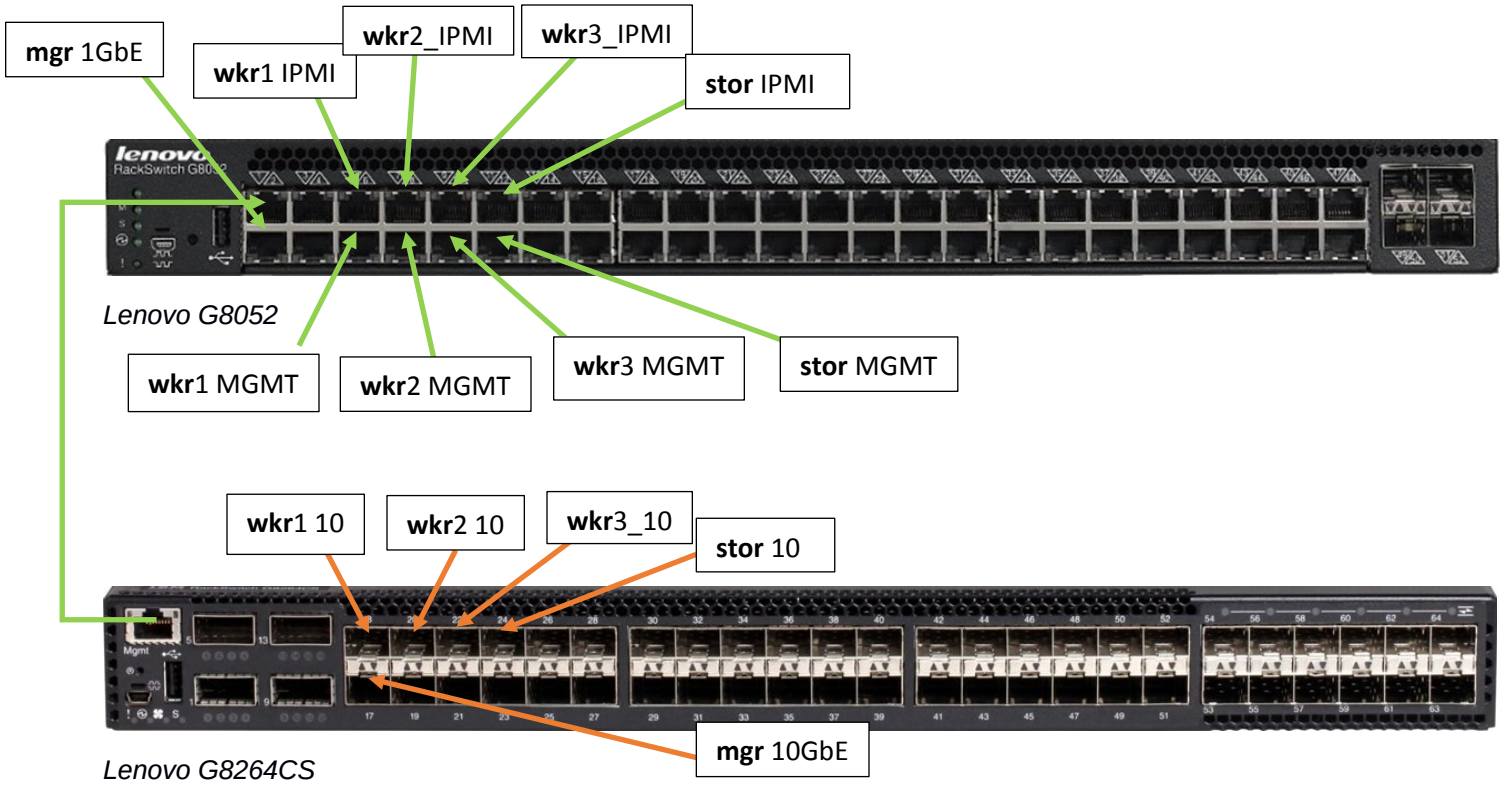

Figure 5. Network switch cabling scheme

The 10 GbE switch creates the storage networks and connects to a 10 GbE port on each server. The management node (**mgr**) is also connected to this switch. The management port of the switch is connected to the management network and is given an address on the same subnet as the management port on the 1 GbE switch.

If the InfiniBand option is used, there is a third switch for InfiniBand. The Cluster Genesis tool does not currently configure InfiniBand switches, so the port connections on the switch do not matter from server to server.

Step 3: Choose the basic configuration parameters

To facilitate the automation process of the solution, collect the parameters in *Table 1*. This data is edited into the configuration file, which is used to automatically configure and deploy the entire solution.

Table 1. Configuration parameters

Parameter	Description	Example
Domain name		Ibm.com
Upstream DNS servers	Although a Domain Name System (DNS) server is configured within the cluster, upstream DNS servers must be defined because the names cannot otherwise be resolved.	*4.4.4.4, 8.8.8.8 as default public upstream DNS servers
Deployment node host name	The name of the deployment node	mgmtnode
Management network IP address	Management for the cluster happens on its own internal network.	192.168.3.0 /24
Data network IP address	Labeled *interconnect* in the configuration file.	10.0.0.1 /24
Management switch IP address	Labeled *ipaddr-mgmt-switch* in the configuration file.	192.168.3.5
Data switch IP addresses	Labeled *ipaddr-data-switch* in the configuration file.	192.168.3.6
Data node hostnames and IPs addresses	Each node in the cluster needs a host name and an IP address for each of the management and data networks.	**Name** / **Management IP** / **Data IP** **wkr**1 / 192.168.3.102 / 10.0.0.2 **wkr**2 / 192.168.3.103 / 10.0.0.3 **wkr**3 / 192.168.3.104 / 10.0.0.4 **stor**1 / 192.168.3.99 / 10.0.0.99

Step 4: Prepare the management node

The management server is the multi-homed head node of the cluster. It has access to any external networks as well as the internal cluster networks. Additionally, when using Cluster Genesis for automated deployment, the management node automatically checks out the latest software and deployment tools to install and populate the cluster. The management node is defined as a 1U OpenPOWER server, but it can also be an x86 server with at least 32 GB RAM.

Ubuntu 16.04 is needed on the management server. You can obtain it from the following locations:

- POWER8 servers: https://www.ubuntu.com/download/server/power8
- X86 servers: http://releases.ubuntu.com/16.04/

Step 5: Configure the cluster using the Cluster Genesis tool

A correct and complete configuration file enables the Cluster Genesis tool to power on, initialize, configure, and install the cluster. This deployment kit provides an automated method to quickly and more predictably go from a physical rack and stack to an operational state.

The Cluster Genesis tool automatically initializes and configures the hardware by accomplishing the following tasks:

- Reading the configuration files with edited, environment-specific changes
- Driving the baseboard management controllers (BMCs) to populate the IP addresses to the nodes
- Detecting and populating relevant configuration data to the management node
- Deploying the required operating system images to the server nodes
- Configuring the network switches (some manual steps are required)
- Installing the required software after the nodes in the cluster are up

Obtain the solution and the default configuration file

Use the **git clone** command to check out the solution to the management server.

```
$ git clone https://www.github.com/open-power-ref-design/deep-learning
```

The Cluster Genesis automation tool uses a configuration file, written in YAML, to specify the target cluster configuration. The deployment uses this YAML text file to specify:

- The IP address locations of the managed switches
- The system nodes attached to the switches
- Other useful details for the deployment process

The generic master copy of the configuration file assumes a four-node cluster that consists of three compute nodes and one storage/parameter node. It is located here:

```
https://www.github.com/open-power-ref-design/deep-learning/config.yml.tfdist.4min
```

Customize the configuration file for the environment

The configuration file contains a lot of information. To enable a cluster tailored to a specific environment, edit the configuration file with the parameters that were collected in *Step 2* on page 4. Refer to *Figure 6* - *Figure 10,* and replace the **red** text with your data.

```
reference-architecture:
  nvidia_playbook:
    description: playbook for installing cuda repository
    cuda_deb: “packages/cuda8.deb”
    cudnn5_deb: “packages/cudnn5.deb”
    cudnn_dev_deb: “packages/cudnn_dev.deb”
    dkms_deb: “packages/dkms.deb
  powerai_playbook:
    description: Playbook for installing Power machine learning
frameworks
      mldl_deb: “packages/mldl.deb”
      powerai_frameworks:
        - tensorflow
#       - ibm-caffe
#       - nv-caffe
#       - bvlc-caffe
#       - chainer
#       - torch
#       - theano
      sameples_src: “static/tensorflow/distributed/samples”
      samples_dest: “/opt/DL/tensorflow/samples”
#     ofed_driver: “packages/ofed.tgz”
      models_tree: https://github.com/tensorflow/models.git
```

Figure 6. Configuration file: Frameworks

The `powerai_frameworks` section in *Figure 6* tells the Cluster Genesis tool to install the frameworks listed on all the nodes. Currently, the frameworks supported in PowerAI include ibm-caffe, nv-caffe, bvlc-caffe, chainer, tensorflow, theano, and torch. Uncomment the lines of the desired frameworks.

The `ofed_driver` parameter is required when using the optional InfiniBand network. Uncomment the line if using InfiniBand.

```
ipaddr-mgmt-network: 192.168.3.0/24  ← MGMT (1GbE) Network
ipaddr-mgmt-switch:
     rack1: 192.168.3.5        ← IP Addr for MGMT (1GbE) switch
ipaddr-data-switch:
     rack1: 192.168.3.6        ← IP Addr for data (10GbE) switch
redundant-network: false
userid-default: Ubuntu          ← node OS user ID
password-default: password      ← node OS password
userid-mgmt-switch: admin       ← MGMT (1 GbE) switch user ID
password-mgmt-switch: admin     ← MGMT (1 GbE) switch password
userid-data-switch: admin       ← data (10 GbE) switch user ID
password-data-switch: admin     ← data (10 GbE) switch password

nfs:
     nfs_network: interconnect
     nfs_server: parameter
     nfs_clients: worker
     nfs_shares:
      - /data                   ← mount point for NFS data share
```

Figure 7. Configuration file: Network space and switch parameters

The `ipaddr-mgmt-network` parameter defines the network space for the management network. The `ipaddr-*-switch` parameters have the actual management port addresses for the management and data switches themselves. The user IDs and passwords for the two switches are also needed.

The next section defines the networks in the solution. Each network gets a network space, method (static versus dynamic host configuration protocol [DHCP]), and a specified network interface name.

External network option

During deployment, the management server provides external access to each of the nodes that are being installed. If external connectivity is required for each compute and parameter node in the cluster after deployment, the external network can also be defined for the compute nodes by uncommenting and customizing the external network definition shown in *Figure 8* on page 13.

```
networks:
#  external:
#     addr: 1.2.3.0/24       ← external network CIDR
#     broadcast: 1.2.3.255   ← external network broadcast
#     gateway: 1.2.3.1       ← external network gateway
#     dns-nameservers: 1.2.3.200  ← external network DNS
#     dns-search: example.com     ← external network DNS search
#     method: static
#     eth-port: eth10        ← external networks should use eth10
  interconnect:
    description: Private 10G Storage Network
    addr: 10.0.0.0/24       ← storage network (10 GbE) CIDR
    broadcast: 10.0.0.255   ← storage network (10 GbE) broadcast
    method: static
    eth-port: eth11     ← interconnect networks should use eth11
  mgmt-pxe:
    description: Management Network
    method: dhcp
    eth-port: eth15
```

Figure 8. Configuration file: External network options

InfiniBand Option

If the InfiniBand option is being used, a few sections need to be in the configuration file, in addition to the `ofed_driver` parameter described in *Figure 6* on page 11. A stanza for the InfiniBand network must be uncommented and customized in the networks section shown in *Figure 9*. The network name must be referenced in each of the template sections for the nodes. See *Figure 9* for the specific additions needed.

```
networks:
#   infiniband:           ← uncomment this section when using IB
#     description: Private IPoIB Network
#     addr: 10.0.1.0/24             ← IB network
#     broadcast: 10.0.1.255         ← IB broadcast IP
#     method: static
#     eth-port: ib0                 ← IB network interface
```

Figure 9. Configuration file: InfiniBand option

Node templates

The node template section includes definitions that are consistent in each class of node. The networking sections of the templates must be reviewed to ensure a correct environment.

Switch ports

The `ports` section of the template defines to which port on each switch the worker nodes are connected. This section must match the physical network created previously. See *Appendix B* on page 19 for information about how to configure this section if adding nodes to the default configuration.

Node network definitions

The `networks` section of the templates, shown in *Figure 10*, defines which networks the worker and parameter/storage nodes are a part of. Uncomment the external or InfiniBand lines in the networks sections to add those networks to each node type.

```
node-templates
  worker:
    networks:
#     - external   ← this define needed if using an ext. network
#     - infiniband ← this define needed if using InfiniBand
      -  interconnect
      -  mgmt-pxe
  parameter:
    networks:
#     - external  ← This define needed if using an ext. network
#     - infiniband  ← This define needed if using InfiniBand
      - interconnect
      - mgmt-pxe
```

Figure 10. Configuration file: Networks

The configuration file is now customized for the environment. In this document, this new file is called *myconfig.yml*.

Additional Manual Steps

A few steps cannot be automated today. These include software dependencies that are currently behind a required login or that require manual click-through to accept a license. Complete the following steps before running the cluster automation to manually download the required software dependencies.

Create the management-network bridge

During deployment, the Cluster Genesis tool creates a container that runs the boot time services. A bridge must be created for this container to have access to the management network. Creation of this bridge is a manual step.

On the management node, edit the `/etc/network/interfaces` file and add the following lines. Customize the red text for the environment. Note that the interface in the `bridge_ports` stanza (enP1p3s0f0) is the interface connected to the management network.

```
auto br0
iface br0 inet static
      address 192.168.3.3
      netmask 255.255.255.0
      bridge_ports enP1p3s0f0
```

A system reboot or **`sudo ifup br0`** command activates the bridge network interface.

NVIDIA CuDNN

The NVIDIA CUDA Deep Neural Network (cuDNN) library is a prerequisite. However, it requires a user registration for download. To download the latest NVIDIA CuDNN packages, visit the NVIDIA website.

```
https://developer.nvidia.com/cudnn
```

Log in to, or register for, the NVIDIA Accelerated Computing Developer Program. Download the following .deb files:

- cuDNN v5.1 Runtime Library for Ubuntu16.04 POWER8 (Deb)
- cuDNN v5.1 Developer Library for Ubuntu16.04 POWER8 (Deb)

Copy the .deb file to the management server and export the location as the CUDNN5 and CUDNN5_DEV environment variables so that the Cluster Genesis tool knows the location.

For example:

```
$ export CUDNN5=/home/Ubuntu/Downloads/libcudnn5_5.1.5-
1+cuda8.0_ppc64el.deb
$ export CUDNN5_DEV=/home/ubuntu/Downloads/libcudnn5-dev_5.1.5-
1+cuda8.0+ppc64el.deb
```

Mellanox OFED

If using the optional InfiniBand network, download Mellanox OpenFabrics (OFFD) from the Mellanox website. Visit the following link to download the correct package. There is a click-through table at the bottom of the page in the *Downloads* tab. Download the .tgz

version of the package.

```
http://www.mellanox.com/page/products_dyn?product_family=26&mtag=linux_sw_drivers
```

After it is downloaded, copy the .tgz file to the management server. Export the location as the `MLX_OFED` environment variable so that the Cluster Genesis tool can find it. For example:

```
export MLNX_OFED=/home/Ubuntu/Downloads/MLNX_OFED_LINUX-4.0-2.0.0.1-ubuntu16.04-ppc64le.tgz
```

Note: If using the InfiniBand option, a session manager is required. Enable the InfiniBand session manager on the InfiniBand switch. InfiniBand switch management is not currently included in the Cluster Genesis tool.

Perform the deployment of Cluster Genesis

To deploy the Cluster Genesis tool, run the installation script.

```
$ ./install.sh
```

The installation script checks out Cluster Genesis from its own GitHub repository. It applies patches and downloads the various dependent packages required for the installation. These dependencies include NVIDIA CUDA, NVIDIA CuDNN, PowerAI, and a few specific Ubuntu packages needed during the automated deployment.

After the *install.sh* is run cleanly, start the automated Genesis deployment:

```
$ ./deploy.sh myconfig.yml
```

The inventory file

During the Cluster Genesis deployment, an entire inventory of the cluster is captured and written into a separate YAML file. This file is the de-facto database for the cluster. It consists of the network switches and server nodes. The data structure for the switches indicates the type of switch (management or data), their IP addresses, and associated login credentials. Similarly, the server node entries also contain access information, descriptive information, and general details. Do not edit this file manually.

After Genesis is complete, this inventory file is located on the deployment node in `/var/oprc/inventory.yml`.

Step 6: Complete the post-Cluster Genesis configuration

TensorFlow run scripts

The Cluster Genesis automation tool fully installs the nodes with an operating system and the necessary software, including TensorFlow. It also places a run script on each node at the following location:

```
/opt/DL/tensorflow/samples/dist_driver.sh
```

The script is personalized for each node, and it can be used to run distributed models on the cluster. Note that not all TensorFlow models implement distributed training. They must be explicitly written and tuned to take advantage of the distributed TensorFlow functions.

Shared storage

Cluster Genesis creates a shared Network File System (NFS) storage disk, and the TensorFlow models library is checked out there. The default mounted NFS share location for these models is:

```
/data/models
```

There is a distributed version of Inception V3 included in the model library. ImageNet is a common academic data set to use with the Inception V3 model, and it can be downloaded from:

```
http://www.image-net.org
```

Appendix A: Hardware orientation

Compute node

Figure 11 on page 18 shows the back of the compute nodes (S822LC for high-performance computing [HPC]) and its base networking.

Note: Although these servers can share the BMC service port (a multi-function port), the automation requires the port to be set up for BMC data only.

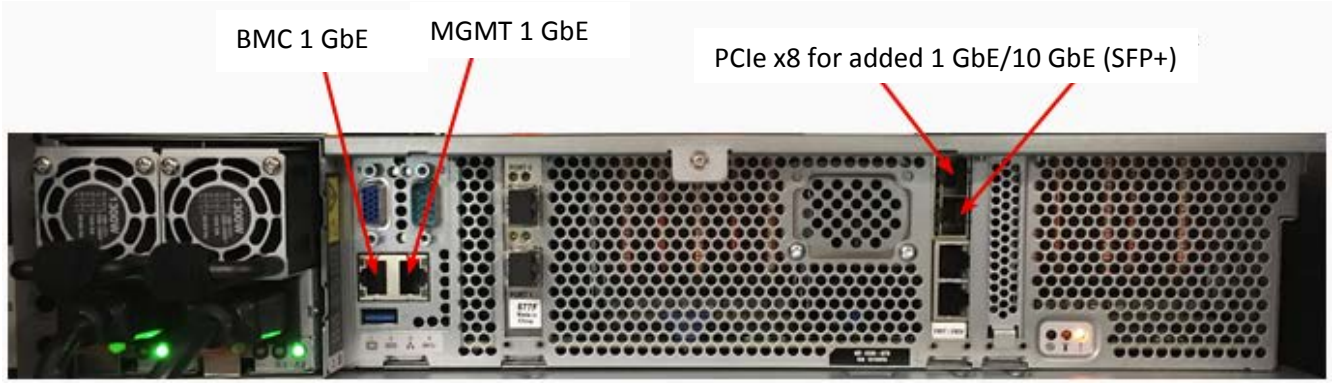

Figure 11. Compute node networking

Management node

Figure 12 shows a rear view of the management node (S812LC). The four native Ethernet (RJ45) ports support 1 GbE, 10 GbE, or 100 MbE, and they can be configured for separate speeds. This solution also includes a 10 GbE card (SFP+) in the PCI3 Slot3.

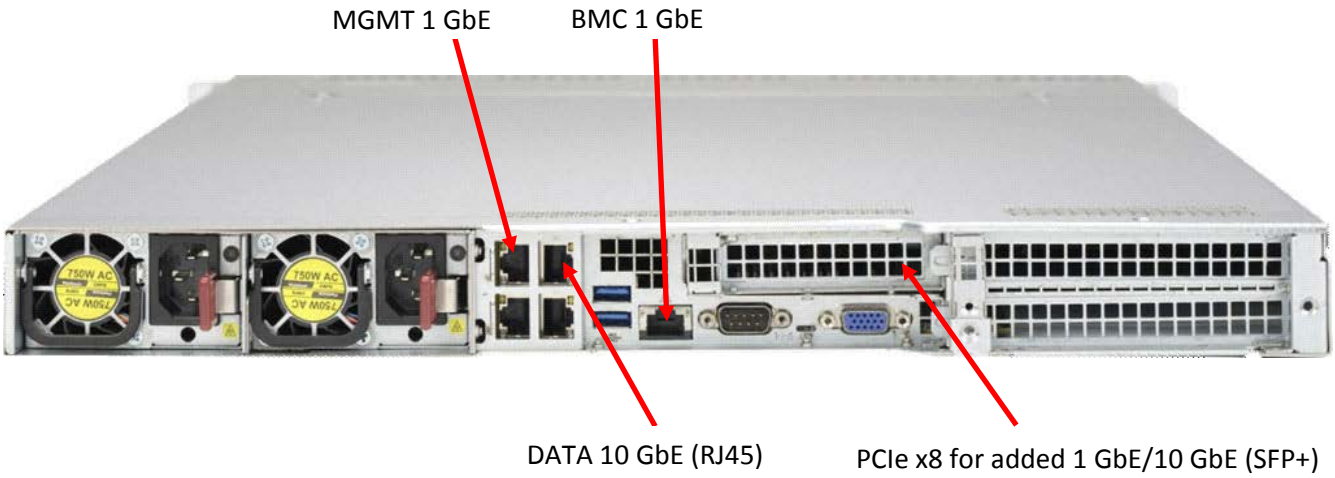

Figure 12. Management node

Storage node

The storage node (S822LC) rear view is shown in *Figure 13* on page 19. The four native Ethernet ports (RJ45) support 1 GbE, 10 GbE, or 100 MbE, and they can be configured for separate speeds. The solution also includes a 10 GbE card (SFP+) in the PCI3 Slot3.

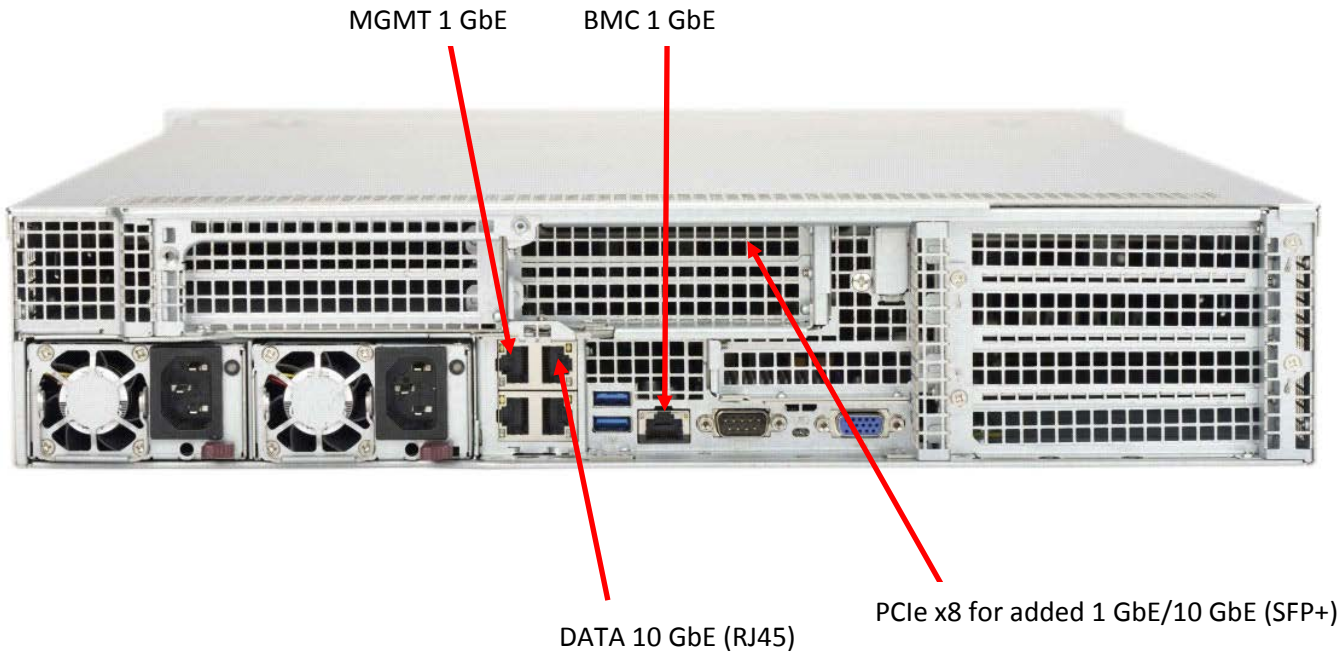

Figure 13. Storage node

Appendix B. How to add a node to the design before deployment

The repository for this solution includes configuration files for a 4-node cluster and an 8-node cluster. To add nodes to these configuration files, the template section for each node class must be edited.

Node templates

The node template section includes definitions that are consistent in each class of node. The networking sections of the templates control how many nodes will be installed and configured as well as which nodes are which.

Switch ports

The ports section of the template defines where on each switch the worker nodes are connected. To add a worker node, a new port number must be added to the list for each network in the ports section. At minimum, a new node must be added to both the prebooot execution environment (PXE) and IPMI networks. The new node must then be physically connected to the switch ports defined in the configuration file. For more information, see the example in *Figure 14* on page 20.

```
node-templates
  worker:
    ports:
      ipmi:
        rack1:
          - 5  ← 1 GbE Switch ports used for worker nodes BMCs
          - 7  ←
          - 9  ←
          - 11 ← Added node
      pxe:
        rack1:
          - 6  ← 1 Gbe Switch ports used for worker nodes mgmt
          - 8  ←
          - 10 ←
          - 12 ← Added node
      eth10:
        rack1:
          - 17 ← 10 GbE Switch ports used for external network
          - 19 ←
          - 21 ←
          - 23 ← Added node
      eth11:
        rack1:
          - 18 ← 10 GbE Switch ports used for storage network
          - 20 ←
          - 22 ←
          - 24 ← Added node
```

Figure 14. Node templates

Appendix C: References

The following links provide more information related to this document:

- Distributed TensorFlow
- How to Train from Scratch in a Distributed Setting
- TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems

Printed in the United States of America June 2017

IBM Systems
294 Route 100, Building SOM4
Somers, NY 10589-3216

The IBM home page can be found at ibm.com®.

Version 1.0
1 June 2017