



## **Bare-metal Reprovisioning**

# Table of contents

Initial Provisioning of Golden Image to BMC

---

Retrieving Golden Image Version Information

---

Retrieving Golden Image Version Information Using Redfish

---

Updating Golden Images Using Redfish

---

OOB Network Configuration

---

Golden Images Reprovisioning

---

Signaling BlueField BMC After Arm OS Programming Completion

---

Adding Entries to RShim Log from BlueField Arm OS

---

Expected Output

---

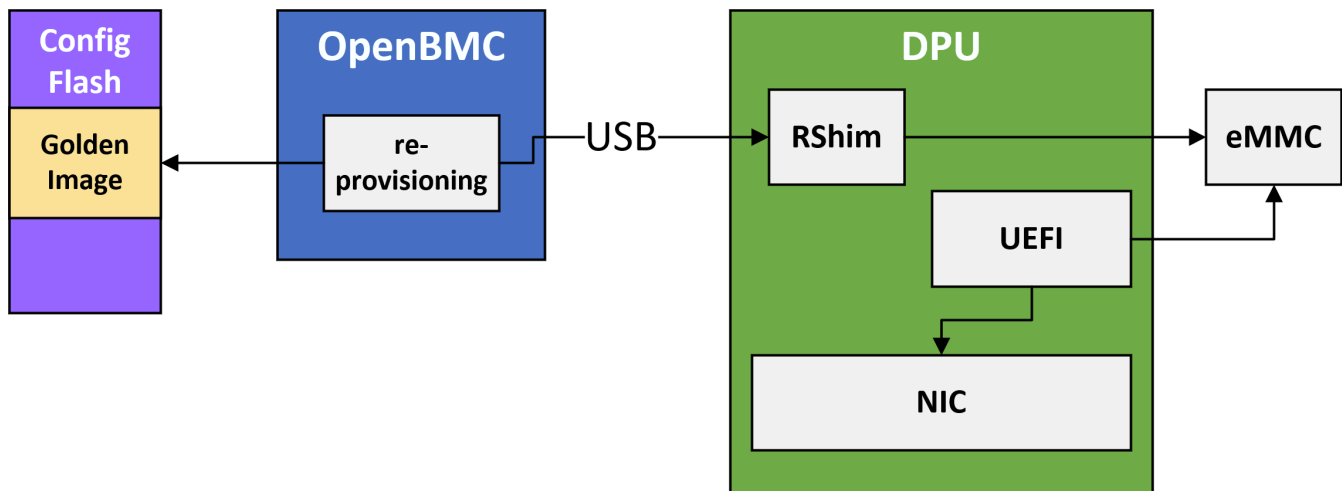
**Note**

Relevant for NVIDIA® BlueField®-3 and later in DPU mode only (not supported in NIC mode).

The re-provisioning flow of the BlueField-3 bare metal system offers a solution for restoring the system to its initial state using built-in resources, eliminating the need for external measures. This approach enables the seamless reloading of the operational image.

To support this functionality, the BMC maintains and manages a golden image for the UEFI and the NIC. This ensures that the UEFI can retrieve the operational image via network protocols such as HTTP or PXE.

The following block diagram provides a high-level overview of the system components and data flow:



The complete flow of network re-provisioning includes the following primary stages:

1. Initial provisioning – Provisioning the golden images to the BMC, typically performed during system manufacturing.
2. In-field updates – Updating the golden images using the in-field update process.
3. OOB network configuration – Configuring network settings through out-of-band (OOB) management.

4. System recovery – Restoring the system by reinstalling the golden images.

## Initial Provisioning of Golden Image to BMC

1. Ensure the BMC is connected to the out-of-band (OOB) network.
2. Use the `scp` command to transfer the golden images from your local storage to the BMC's temporary storage directories (`/tmp/golden-image-nic/` or `/tmp/golden-image-arm/`).

- For `golden_image_nic`:

```
#host> scp <nic-golden-image-directory>/<nic-golden-image-filename> root@<bmc-ip>:/tmp/golden-image-nic/
```

- For `golden_image_arm`:

```
#host> scp <arm-golden-image-directory>/<arm-golden-image-filename> root@<bmc-ip>:/tmp/golden-image-arm/
```

### Info

After the image is copied to the BMC's volatile memory, the version is extracted and stored to enable specific features.

**Note**

The NIC firmware version is extracted from the image filename. Ensure the filename follows the standard format of official releases.

3. Log into the BMC and use the `dpu_golden_image` utility to transfer the golden images from temporary storage to the BMC's non-volatile storage.

- For `golden_image_nic`:

```
#bmc> dpu_golden_image golden_image_nic -w /tmp/golden-image-nic/<nic-golden-image-  
filename>
```

- For `golden_image_arm`:

```
#bmc> dpu_golden_image golden_image_arm -w /tmp/golden-image-arm/<arm-golden-  
image-filename>
```

## **i** Info

If the version of the candidate image matches the one already in non-volatile storage, the update is skipped, and the following message is displayed:

```
Updating image in memory is cancelled as
version will remain unchanged. Force
update by adding -f|--force to the
command line.
```

To force the update, use the `--force` flag:

- For `golden_image_nic`:

```
#bmc> dpu_golden_image golden_image_nic -w /tmp/golden-
image-nic/<nic-golden-image-filename> --force
```

- For `golden_image_arm`:

```
#bmc> dpu_golden_image golden_image_arm -w
/tmp/golden-image-arm/<arm-golden-image-filename> --
force
```

4. After provisioning, verify the correctness of the golden images using the following commands:

- For `golden_image_nic`:

```
#bmc> dpu_golden_image -v golden_image_nic
bmc> echo $?
```

Expected output: 0.

- For `golden_image_arm`:

```
bmc> dpu_golden_image -v golden_image_arm
bmc> echo $?
```

Expected output: 0.

## Retrieving Golden Image Version Information

### Note

This feature is available only for golden images installed following the upgrade of the BMC firmware to version 24.07-14 or later.

To get the human-readable version (`MAJOR.MINOR.PATCH.BUILD` versioning scheme) of the golden images:

- For `golden_image_nic`:

```
bmc> dpu_golden_image golden_image_nic -V -H
```

- For `golden_image_arm`:

```
bmc> dpu_golden_image golden_image_arm -V -H
```

To get the `sha256sum` value:

- For `golden_image_nic`:

```
bmc> dpu_golden_image golden_image_nic -V
```

- For `golden_image_arm`:

```
bmc> dpu_golden_image golden_image_arm -V
```

## Retrieving Golden Image Version Information Using Redfish

### Note

This feature is available only for Golden Images installed following the upgrade of the BMC firmware to version 24.07-14 or later.

To get the human-readable version (`MAJOR.MINOR.PATCH.BUILD` versioning scheme) of the golden images over the Redfish interface, run:

- For Arm golden image:

```
curl -k -u '<username>': '<password>' -H 'Content-type: application/json' -X GET  
'https://<bmc_ip>/redfish/v1/UpdateService/FirmwareInventory/golden_image_arm'
```

- For NIC golden image:

```
curl -k -u '<username>': '<password>' -H 'Content-type: application/json' -X GET  
'https://<bmc_ip>/redfish/v1/UpdateService/FirmwareInventory/golden_image_nic'
```

## Updating Golden Images Using Redfish

The process for updating golden images via Redfish includes the following steps.

1. The process for updating golden images via Redfish includes the following steps:
  - For NIC golden image:

```
curl -k -u root: '<password>' -H "Content-Type: application/json" -X POST  
-d '{"TransferProtocol": "HTTP", "ImageURI": "<remote-server-ip>/<nic-golden-image-path>",  
"Targets": ["redfish/v1/UpdateService/FirmwareInventory/golden_image_nic"]}'  
https://<bmc-  
ip>/redfish/v1/UpdateService/Actions/UpdateService.SimpleU
```

- For Arm golden image:

```
curl -k -u root: '<password>' -H "Content-Type: application/json" -X POST  
-d '{"TransferProtocol": "HTTP", "ImageURI": "<remote-server-ip>/<arm-golden-image-path>",  
"Targets": ["redfish/v1/UpdateService/FirmwareInventory/golden_image_arm"]}'  
https://<bmc-  
ip>/redfish/v1/UpdateService/Actions/UpdateService.SimpleU
```

Parameters:

- `ImageURI` – Specify the image location in the format `<remote-server-ip>/<golden-image-path>`
- `bmc-ip` – Specify the IP address of the BMC

After initiating the update, a new task is created for monitoring progress, with a sample response:

```
{
  "@odata.id": "/redfish/v1/TaskService/Tasks/0",
  "@odata.type": "#Task.v1_4_3.Task",
  "Id": "0",
  "TaskState": "Running",
  "TaskStatus": "OK"
}
```

2. Track the update's progress using the following command:

```
curl -k -u root:'<password>' -X GET https://<bmc-ip>/redfish/v1/TaskService/Tasks/<task-id>
```


Update states:

- `0%` – Update started.
- `10%` – Update in progress.
- `100%` – Update complete.

Expected output after a successful update:

```
"PercentComplete" : 100,  
"TaskState" : "Completed",  
"TaskStatus" : "OK"
```

In case of failure, reboot the BMC and retry the update.

 **Info**

The golden image update process typically takes 1–3 minutes to complete.

If the candidate image version matches the one stored in the BMC's non-volatile memory, the update is skipped by default, and the following response is returned:

```

{
  ...
  {
    "@odata.type": "#MessageRegistry.v1_4_1.MessageRegistry",
    "Message": "The update operation for the component 'BMC' is
skipped because 'Component image is identical'.",
    "MessageArgs": [
      "BMC",
      "Component image is identical"
    ],
    "MessageId": "NvidiaUpdate.1.0.ComponentUpdateSkipped",
    "Resolution": "Retry firmware update operation with the
force flag",
    "Severity": "OK"
  },
],
...
"TaskState": "Completed",
"TaskStatus": "OK"
}

```

To override the default behavior and force the update, include the `"ForceUpdate": true` parameter in the command:

- For NIC golden image:

```

curl -k -u root:'<password>' -H "Content-Type: application/json" -X POST -d
'{"TransferProtocol":"HTTP", "ImageURI":"<remote-server-ip>/<nic-golden-image-path>", "Targets":
[ "redfish/v1/UpdateService/FirmwareInventory/golden_image_nic", "ForceUpdate": true]}'
https://<bmc-
ip>/redfish/v1/UpdateService/Actions/UpdateService.SimpleUpdate

```

- For Arm golden image:

```
curl -k -u root:'<password>' -H "Content-Type: application/json" -X POST -d
'{"TransferProtocol":"HTTP", "ImageURI":"<remote-server-ip>/<arm-golden-image-path>", "Targets":
[{"redfish/v1/UpdateService/FirmwareInventory/golden_image_arm"}, {"ForceUpdate": true}]
https://<bmc-
ip>/redfish/v1/UpdateService/Actions/UpdateService.SimpleUpdate
```

## OoB Network Configuration

To enhance the system's security, a new mechanism has been introduced to control network connectivity over the OoB network. This new feature provides an IPMI command to disable any communication between the BlueField BMC, BlueField, and the OoB management network. A set of IPMI commands are introduced to selectively enable the network on each of the above interfaces. This permits the platform's RoT to have complete control over which network interfaces can be enabled and when.

### Note

This IPMI can only be sent by the platform's ROT. OoB and BlueField are blocked.

By default, the OoB interface is enabled. However, for the host BMC to gain control over this interface, it must disable it during the initial boot. Once disabled, the interface remains in that state regardless of BMC reboots or system cold boots.

For more details, refer to ["OoB Network 3-Port Switch Control"](#).

## Golden Images Reprovisioning

The re-provisioning flow is initiated using the following IPMI command:

```
bmc> ipmitool raw 0x32 0x99 <golden_image_timeout>
<timeout_from_network> <verbosity_level> <halt_hard_reset>
```

This command must be executed from within the BMC, as it can significantly impact the system. Upon execution:

1. The golden images are extracted from the BlueField BMC's non-volatile memory.
2. The recovery process is initiated, pushing the golden images to the RShim.
3. The RShim console output is redirected to the BMC console, allowing the user to monitor the process.

Once the process completes, both the BlueField NIC and ARM execute the designated golden images retrieved from a preconfigured server.

Command parameters:

- `<golden_image_timeout>` – Timeout value (in minutes) for updating the golden images. Default Value: `15` minutes (input `0` to use the default).
- `<timeout_from_network>` – Timeout value (in minutes) for booting the operational image from the network. Default Value: `60` minutes (input `0` to use the default).
- `<verbosity_level>` – Controls the level of detail displayed during the reprovisioning process:
  - `0` – Quiet mode (only error messages are displayed)
  - `1` – Info mode (error messages and reprovisioning process messages are displayed)
  - `2` – Full mode (all messages, including BlueField RShim messages, are displayed)
- `<halt_hard_reset>` (Optional) – Specifies whether to halt the reprovisioning process before performing the final hard reset of the BlueField.

### **(i) Note**

The final hard reset is crucial to activate the NIC firmware installed from the network.

- `0` – Perform the hard reset to complete the reprovisioning process (default)
- `1` – Halt the process before the final hard reset

### **(i) Info**

Reprovisioning messages are prefixed with

```
[<running date> GOLDEN-IMAGE-RECOVERY].
```

## **Signaling BlueField BMC After Arm OS Programming Completion**

After BFB installation is complete, the BlueField BMC waits for a specific sequence of messages over the RShim log:

```
NIC firmware update done           # Indicates that the firmware
update for the NIC subsystem has been successfully completed
Installation finished               # Signals the completion of the
installation process for the BFB from the network
Linux up                            # Indicates that
the BlueField BMC has acknowledged that the Arm OS has booted up
and is ready
```

BlueField BMC expects these messages in the specified order.

## Adding Entries to RShim Log from BlueField Arm OS

Users can add custom entries to the RShim log from the BlueField Arm OS using the `bfrshlog` command. The syntax of the command is: `bfrshlog <output>`.

For example, to add the message "Linux up" to the RShim log, run:

```
bfrshlog "Linux up"
```

### Expected Output

- All output from the BlueField Arm console is redirected to the BlueField BMC console for monitoring purposes.
- The steps of the re-provisioning process are printed with `[<running date> GOLDEN-IMAGE-RECOVERY]` prefix and are outlined in the following:

```
[<running date> GOLDEN-IMAGE-RECOVERY] Checking pcie slot is in
reset
[<running date> GOLDEN-IMAGE-RECOVERY] Read golden images from
flash
[<running date> GOLDEN-IMAGE-RECOVERY] Set FNP to 0
[<running date> GOLDEN-IMAGE-RECOVERY] Checking rshim interface
after SOC hard reset
[<running date> GOLDEN-IMAGE-RECOVERY] Starting ATF/UEFI golden
image update
[<running date> GOLDEN-IMAGE-RECOVERY] Finished updating ATF/UEFI
golden image
[<running date> GOLDEN-IMAGE-RECOVERY] Starting NIC FW golden
image update
[<running date> GOLDEN-IMAGE-RECOVERY] Finished updating NIC FW
golden image
[<running date> GOLDEN-IMAGE-RECOVERY] Stop Redfish server
[<running date> GOLDEN-IMAGE-RECOVERY] Configure Recovery image
to boot from network
[<running date> GOLDEN-IMAGE-RECOVERY] set FNP to 1
[<running date> GOLDEN-IMAGE-RECOVERY] Booting BFB from network
[<running date> GOLDEN-IMAGE-RECOVERY] Start Redfish server
[<running date> GOLDEN-IMAGE-RECOVERY] Set boot option to default
if halt_hard_reset is 0:
[<running date> GOLDEN-IMAGE-RECOVERY] Finished programming image
from network. Start DPU hard reset
if halt_hard_reset is 1:
[<running date> GOLDEN-IMAGE-RECOVERY] Finished programming image
from network
[<running date> GOLDEN-IMAGE-RECOVERY] The Reprovisioning process
was halted at user's request. To complete the process, please
power cycle the device
```

A failed update prints the following:

```
[<running date> GOLDEN-IMAGE-RECOVERY] ERROR: aborting process!  
PCIE is not in reset.  
[<running date> GOLDEN-IMAGE-RECOVERY] ERROR: Reading  
golden_image_nic failed  
[<running date> GOLDEN-IMAGE-RECOVERY] ERROR: Reading  
golden_image_arm failed  
[<running date> GOLDEN-IMAGE-RECOVERY] ERROR: rshim has not  
started successfully  
[<running date> GOLDEN-IMAGE-RECOVERY] ERROR: pushing ATF/UEFI  
golden image over rshim failed  
[<running date> GOLDEN-IMAGE-RECOVERY] ERROR: programming of  
ATF/UEFI golden image failed  
[<running date> GOLDEN-IMAGE-RECOVERY] ERROR: pushing NIC FW  
golden image over rshim failed  
[<running date> GOLDEN-IMAGE-RECOVERY] ERROR: programming of NIC  
FW golden image failed  
[<running date> GOLDEN-IMAGE-RECOVERY] ERROR: failed to configure  
image to boot from network  
[<running date> GOLDEN-IMAGE-RECOVERY] ERROR: programming of  
image from network failed: NIC firmware update failed  
[<running date> GOLDEN-IMAGE-RECOVERY] ERROR: programming of  
image from network failed: Installation failed  
[<running date> GOLDEN-IMAGE-RECOVERY] ERROR: programming of  
image from network failed: Failed to get Linux up
```

Due to line buffering in the BlueField Arm console, buffered output lines receive the same timestamp value in `<running date>` when they are redirected to the BlueField BMC console.

## Notice

This document is provided for information purposes only and shall not be regarded as a warranty of a certain functionality, condition, or quality of a product. NVIDIA Corporation ("NVIDIA") makes no representations or warranties, expressed or implied, as to the accuracy or completeness of the information contained in this document and assumes no responsibility for any errors contained herein. NVIDIA shall have no liability for the consequences or use of such information or for any infringement of patents or other rights of third parties that may result from its use. This document is not a commitment to develop, release, or deliver any Material (defined below), code, or functionality.

NVIDIA reserves the right to make corrections, modifications, enhancements, improvements, and any other changes to this document, at any time without notice.

Customer should obtain the latest relevant information before placing orders and should verify that such information is current and complete.

NVIDIA products are sold subject to the NVIDIA standard terms and conditions of sale supplied at the time of order acknowledgement, unless otherwise agreed in an individual sales agreement signed by authorized representatives of NVIDIA and customer ("Terms of Sale"). NVIDIA hereby expressly objects to applying any customer general terms and conditions with regards to the purchase of the NVIDIA product referenced in this document. No contractual obligations are formed either directly or indirectly by this document.

NVIDIA products are not designed, authorized, or warranted to be suitable for use in medical, military, aircraft, space, or life support equipment, nor in applications where failure or malfunction of the NVIDIA product can reasonably be expected to result in personal injury, death, or property or environmental damage. NVIDIA accepts no liability for inclusion and/or use of NVIDIA products in such equipment or applications and therefore such inclusion and/or use is at customer's own risk.

NVIDIA makes no representation or warranty that products based on this document will be suitable for any specified use. Testing of all parameters of each product is not necessarily performed by NVIDIA. It is customer's sole responsibility to evaluate and determine the applicability of any information contained in this document, ensure the product is suitable and fit for the application planned by customer, and perform the necessary testing for the application in order to avoid a default of the application or the product. Weaknesses in customer's product designs may affect the quality and reliability of the NVIDIA product and may result in additional or different conditions and/or requirements beyond those contained in this document. NVIDIA accepts no liability related to any default, damage, costs, or problem which may be based on or attributable to: (i) the use of the NVIDIA product in any manner that is contrary to this document or (ii) customer product designs.

No license, either expressed or implied, is granted under any NVIDIA patent right, copyright, or other NVIDIA intellectual property right under this document. Information published by NVIDIA regarding third-party products or services does not constitute a license from NVIDIA to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property rights of the third party, or a license from NVIDIA under the patents or other intellectual property rights of NVIDIA.

Reproduction of information in this document is permissible only if approved in advance by NVIDIA in writing, reproduced without alteration and in full compliance with all applicable export laws and regulations, and accompanied by all associated conditions, limitations, and notices.

THIS DOCUMENT AND ALL NVIDIA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, "MATERIALS") ARE BEING PROVIDED "AS IS." NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL NVIDIA BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF

ANY USE OF THIS DOCUMENT, EVEN IF NVIDIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Notwithstanding any damages that customer might incur for any reason whatsoever, NVIDIA's aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms of Sale for the product.

## **Trademarks**

NVIDIA and the NVIDIA logo are trademarks and/or registered trademarks of NVIDIA Corporation in the U.S. and other countries. Other company and product names may be trademarks of the respective companies with which they are associated.

© Copyright 2026, NVIDIA. PDF Generated on 02/28/2026