



## **Boot Configuration**

# Table of contents

## Boot Config Using Redfish

---

Retrieving Information on Pending Boot Configurations

---

Changing BootOrder Configuration

---

Example of Changing BootOrder Configuration

---

Retrieving Active Boot Configuration Values

---

Applying Pending Boot Configurations

---

## Boot Source Override

---

Setting Persistent PXE Boot

---

Resetting Boot Override to Default

---

Boot Source Override Config Using IPMI

---

Boot Source Override Config Using Redfish

---

Retrieving the Current Boot Override Settings

---

Get Boot Source Override Configuration

---

Configuring One-Time PXE Boot without Timeout

---

Behavior of Boot Source Override on BlueField

---

Examples

---

Configuring One-Time PXE Boot with Timeout

---

Set Boot Source Override Configuration

---

BMC supports boot option selection commands using the Redfish or IPMI interfaces. UEFI on NVIDIA® BlueField® can query for the boot options through an IPMI/Redfish command. The BMC IPMI command supports changing the boot device selector flag only through the following options: PXE boot, or the default boot device as selected in the boot menu on BlueField. In contrast, the Redfish interface supports all available boot options.

## Boot Config Using Redfish

### Retrieving Active Boot Configuration Values

- To retrieve the active boot configuration, run:

```
curl -k -u root:'<password>' -X GET  
https://<bmc_ip>/redfish/v1/Systems/Bluefield
```

#### Info

The relevant configurations would be under `Boot`.

- To retrieve all boot options (active and pending):

```
curl -k -u root:'<password>' -X GET  
https://<bmc_ip>/redfish/v1/Systems/Bluefield/BootOptions/
```

- To retrieve detailed information on a specific boot option:

```
curl -k -u root:'<password>' -X GET
https://<bmc_ip>/redfish/v1/Systems/Bluefield/BootOptions/<boo
option>
```

## Retrieving Information on Pending Boot Configurations

- To retrieve the pending boot settings:

```
curl -k -u root:'<password>' -X GET
https://<bmc_ip>/redfish/v1/Systems/Bluefield/Settings
```

- The following command retrieves only `BootOptions` configurations with a pending value different than the active one.

```
curl -k -u root:'<password>' -X GET
https://<bmc_ip>/redfish/v1/Systems/Bluefield/Settings/BootOpt
```

- To retrieve the details of a specific pending boot option:

```
curl -k -u root:'<password>' -X GET
https://<bmc_ip>/redfish/v1/Systems/Bluefield/Settings/BootOpt
id>
```

## Applying Pending Boot Configurations

## **Note**

Power reset of the BlueField is necessary for these changes to take effect.

- To alter the boot configuration, applying patches to the setting attribute is required :

```
curl -k -u root:'<password>' -X PATCH
https://<bmc_ip>/redfish/v1/Systems/Bluefield/Settings -d
'{"Boot":{ ... }{'
```

- To set the pending boot order. The list must contain all the Boot option, even if the boot option is disabled.

```
curl -k -u root:'<password>' -X PATCH
https://<bmc_ip>/redfish/v1/Systems/Bluefield/Settings/
-d '{"Boot":{ "BootOrder": ["Boot0002", ..., "BootXXX"]
}}{'
```

- To alter the bootOption value, currently supporting only BootOptionEnable

```
curl -k -u root:'<password>' -X PATCH
https://<bmc_ip>/redfish/v1/Systems/Bluefield/Settings/BootOpt
id> -d '{"BootOptionEnabled": false}'
```

## **Changing BootOrder Configuration**

To set boot order using boot order schema, follow this procedure:

1. Check the current boot order by doing GET on the `ComputerSystem` schema over 1GbE OOB to the BlueField BMC. Look for the `BootOrder` attribute under the `Boot`.

```
curl -k -X GET -u root:<password> https://<BF-BMC-IP>/redfish/v1/Systems/<SystemID>/ | python3 -m json.tool
{
  ....
  "Boot": {
    ....
    "BootOrder": [
      "Boot0017",
      "Boot0001",
      "Boot0002",
      "Boot0003",
      "Boot0004",
      "Boot0005",
      "Boot0006",
      "Boot0007",
    ],
    ....
  }
  ....
}
```

2. To get the details of a particular entity in the `BootOrder` array, perform a GET to the respective `BootOption` URL over 1GbE OOB to the BlueField BMC. For example, to get details of `Boot0006`, run:

```

curl -k -X GET -u root:<password> https://<BF-BMC-
IP>/redfish/v1/Systems/<SystemID>/BootOptions/Boot0006 |
python3 -m json.tool

{
  "@odata.type": "#BootOption.v1_0_3.BootOption",
  "@odata.id":
"/redfish/v1/Systems/SystemId/BootOptions/Boot0006",
  "Id": "Boot0006",
  "BootOptionEnabled": true,
  "BootOptionReference": "Boot0006",
  "DisplayName": "UEFI HTTPv6 (MAC:B8CEF6B8A006)",
  "UefiDevicePath":
"PciRoot(0x0)/Pci(0x0,0x0)/Pci(0x0,0x0)/Pci(0x0,0x0)/Pci(0x0,0
}

```

3. To change the boot order, the entire `BootOrder` array must be PATCHed to the pending settings URI. For this example of the `BootOrder` array, if you intend to have `Boot0006` at the beginning of the array, then the PATCH operation is as follows:

**Note**

Updating the `BootOrder` array results in a permanent boot order change (persistent across reboots).

```
curl -k -u root:<password> -X PATCH -d '{ "Boot": {  
  "BootOrder": [ "Boot0006", "Boot0017", "Boot0001",  
  "Boot0002", "Boot0003", "Boot0004", "Boot0005", "Boot0007", ]  
}}' https://<BF-BMC-  
IP>/redfish/v1/Systems/<SystemID>/Settings | python3 -m  
json.tool
```

4. After a successful PATCH, reboot the BlueField and check if the settings have been applied by doing a GET on the `ComputerSystem` schema.
5. If the `BootOrder` array is updated as intended then the settings have been applied and the BlueField should boot as per the order in preceding cycles.
6. If `BootSourceOverrideEnabled` is set to `Once`, boot override is disabled and any related properties are reset to their former values to avoid repetition. If it is set to `Continuous`, then on every reboot, BlueField would keep performing boot override (`HTTPBoot`).

## Example of Changing BootOrder Configuration

To get the supported boot options:

```

curl -k -u root:<password>' -X GET
https://<bmc_ip>/redfish/v1/Systems/Bluefield/BootOptions
{
  "@odata.id": "/redfish/v1/Systems/Bluefield/BootOptions",
  "@odata.type": "#BootOptionCollection.BootOptionCollection",
  "Members": [
    {
      "@odata.id":
"/redfish/v1/Systems/Bluefield/BootOptions/Boot0000"
    },
    {
      "@odata.id":
"/redfish/v1/Systems/Bluefield/BootOptions/Boot000A"
    },
    {
      "@odata.id":
"/redfish/v1/Systems/Bluefield/BootOptions/Boot000B"
    },
    {
      "@odata.id":
"/redfish/v1/Systems/Bluefield/BootOptions/Boot000C"
    },
    {
      "@odata.id":
"/redfish/v1/Systems/Bluefield/BootOptions/Boot000D"
    },
    {
      "@odata.id":
"/redfish/v1/Systems/Bluefield/BootOptions/Boot000E"
    },
    {
      "@odata.id":
"/redfish/v1/Systems/Bluefield/BootOptions/Boot000F"
    },
    {

```

```
    "@odata.id" :  
"/redfish/v1/Systems/Bluefield/BootOptions/Boot0001"  
  },  
  {  
    "@odata.id" :  
"/redfish/v1/Systems/Bluefield/BootOptions/Boot0002"  
  },  
  {  
    "@odata.id" :  
"/redfish/v1/Systems/Bluefield/BootOptions/Boot0003"  
  },  
  {  
    "@odata.id" :  
"/redfish/v1/Systems/Bluefield/BootOptions/Boot0004"  
  },  
  {  
    "@odata.id" :  
"/redfish/v1/Systems/Bluefield/BootOptions/Boot0005"  
  },  
  {  
    "@odata.id" :  
"/redfish/v1/Systems/Bluefield/BootOptions/Boot0006"  
  },  
  {  
    "@odata.id" :  
"/redfish/v1/Systems/Bluefield/BootOptions/Boot0007"  
  },  
  {  
    "@odata.id" :  
"/redfish/v1/Systems/Bluefield/BootOptions/Boot0008"  
  },  
  {  
    "@odata.id" :  
"/redfish/v1/Systems/Bluefield/BootOptions/Boot0009"  
  },  
  {
```

```
    "@odata.id" :
"/redfish/v1/Systems/Bluefield/BootOptions/Boot0010"
  },
  {
    "@odata.id" :
"/redfish/v1/Systems/Bluefield/BootOptions/Boot0011"
  },
  {
    "@odata.id" :
"/redfish/v1/Systems/Bluefield/BootOptions/Boot0012"
  },
  {
    "@odata.id" :
"/redfish/v1/Systems/Bluefield/BootOptions/Boot0013"
  },
  {
    "@odata.id" :
"/redfish/v1/Systems/Bluefield/BootOptions/Boot0014"
  },
  {
    "@odata.id" :
"/redfish/v1/Systems/Bluefield/BootOptions/Boot0015"
  },
  {
    "@odata.id" :
"/redfish/v1/Systems/Bluefield/BootOptions/Boot0016"
  },
  {
    "@odata.id" :
"/redfish/v1/Systems/Bluefield/BootOptions/Boot0017"
  },
  {
    "@odata.id" :
"/redfish/v1/Systems/Bluefield/BootOptions/Boot0040"
  }
],
```

```
"Members@odata.count": 25,  
"Name": "Boot Option Collection"  
}
```

To set the pending boot order settings:

### **i** Info

In this example, 25 boot options are present. Therefore, the command to establish the boot option order must encompass all 25 options in the active `BootOrder` list according to the desired sequence.

```
curl -k -u root:'<password>' -X PATCH  
https://<bmc_ip>/redfish/v1/Systems/Bluefield/Settings -d  
'{"Boot":{ "BootOrder": ["Boot0040", "Boot0017", "Boot0000",  
"Boot0001", "Boot0002", "Boot0003", "Boot0004", "Boot0005",  
"Boot0006", "Boot0007", "Boot0008", "Boot0009", "Boot000A",  
"Boot000B", "Boot000C", "Boot000D", "Boot000E", "Boot000F",  
"Boot0010", "Boot0011", "Boot0012", "Boot0013", "Boot0014",  
"Boot0015", "Boot0016"] }}'
```

## Boot Source Override

Boot Source Override allows administrators to remotely control the system's boot sequence without requiring physical access to configure boot order settings. This capability supports one-time or persistent boot source overrides, making it useful for automated OS deployment, system recovery, remote diagnostics, and enforcing specific security boot policies.

By dynamically setting the boot target (e.g., PXE, UEFI HTTP), administrators gain flexibility for provisioning, firmware updates, and disaster recovery workflows.

# Boot Source Override Config Using Redfish

## Get Boot Source Override Configuration

To retrieve the current boot source override configuration:

```
curl -k -u root:'<password>' -X GET  
https://<bmc_ip>/redfish/v1/Systems/Bluefield
```

Example output:

```
"Boot": {  
  ...  
  "BootSourceOverrideEnabled": "Disabled",  
  "BootSourceOverrideMode": "UEFI",  
  "BootSourceOverrideTarget": "None",  
  "HttpBootUri": "path",  
  "StopBootOnFault": "Never",  
  "UefiTargetBootSourceOverride": "None"  
  ...  
}
```

## Set Boot Source Override Configuration

To set the boot source override, use:

```
curl -k -u root:'<password>' -X PATCH \  
https://<bmc_ip>/redfish/v1/Systems/Bluefield/Settings \  
-d '{  
  "Boot":{  
    "BootSourceOverrideEnabled": "Once",  
    "BootSourceOverrideMode": "UEFI",  
    "BootSourceOverrideTarget": "UefiHttp",  
    "UefiTargetBootSourceOverride": "None",  
    "BootNext": "",  
    "AutomaticRetryConfig": "Disabled"  
  }  
'
```

### Note

The boot override settings take effect on the next boot and are reflected in the `redfish/v1/Systems/Bluefield` boot schema.

The following parameters can be set when configuring the boot source via the Redfish command:

- `BootSourceOverrideEnabled` –
  - `Disabled` – Disable override
  - `Once` – Apply override for next boot only
  - `Continuous` – Always use the boot override setting
- `BootSourceOverrideMode` – Must be set to `UEFI`
- `BootSourceOverrideTarget` – Must be one of the values allowed under `Boot/BootSourceOverrideTarget@Redfish.AllowableValues`

- `UefiTargetBootSourceOverride` – required if `BootSourceOverrideTarget` is set to `UefiTarget`. Set to the `UefiDevicePath` attribute of the desired boot option.
- `BootNext` – Used if `BootSourceOverrideTarget` is set to `UefiBootnext`
- `AutomaticRetryConfig` – Only `Disabled` is supported

## Examples

### Set Next Boot to a Specific UEFI HTTP Target

1. Query the `BootOptions` attributes:

```
curl -k -u root:'<password>' -X GET \
https://<bmc_ip>/redfish/v1/Systems/Bluefield/BootOptions/Boot
```

Example output:

```
{
  "BootOptionReference" : "Boot0002",
  "DisplayName" : "NET-OOB-IPV4-HTTP",
  "UefiDevicePath" : "MAC(B83FD2CA4B27,0x1)/IPv4(0.0.0.0,0x0,DHCP,...)/Uri()"
}
```

2. Use the `UefiDevicePath` value as the `UefiTargetBootSourceOverride`:

```
curl -k -u root:'<password>' -X PATCH \
https://<bmc_ip>/redfish/v1/Systems/Bluefield/Settings \
-d '{
  "Boot":{
    "BootSourceOverrideEnabled": "Once",
    "BootSourceOverrideMode": "UEFI",
    "BootSourceOverrideTarget": "UefiTarget",
    "UefiTargetBootSourceOverride":
"MAC(B83FD2CA4B27,0x1)/IPv4(0.0.0.0,...)/Uri()",
    "AutomaticRetryConfig": "Disabled"
  }
}'
```

### Set Next Boot to PXE (Non-persistent)

```
curl -k -u root:'<password>' -X PATCH \
https://<bmc_ip>/redfish/v1/Systems/Bluefield/Settings \
-d '{
  "Boot":{
    "BootSourceOverrideEnabled": "Once",
    "BootSourceOverrideMode": "UEFI",
    "BootSourceOverrideTarget": "Pxe"
  }
}'
```

## Boot Source Override Config Using IPMI

The `ipmitool` utility allows configuring the Boot Source Override option, enabling the system to boot from a PXE server or another specified device.

## Retrieving the Current Boot Override Settings

To view the current boot override configuration, run:

```
ipmitool chassis bootparam get 5
```

This command returns information about:

- Whether the boot override option is valid.
- Whether it is persistent or applies to the next boot only.
- The configured boot device type.

## Configuring One-Time PXE Boot with Timeout

To configure a one-time PXE boot with a 60-second timeout, use the following command:

```
ipmitool chassis bootparam set bootflag force_pxe  
options=timeout
```

### **Note**

If the DPU is not reset within 60 seconds, the boot parameters will be invalidated.

## Configuring One-Time PXE Boot without Timeout

To configure a one-time PXE boot without the 60-second timeout, run:

```
ipmitool chassis bootparam set bootflag force_pxe options=no-  
timeout
```

### **(i) Note**

The boot override timer is only applicable for BlueField-3.

### **(i) Note**

It is not recommended to use `ipmitool chassis bootparam` without explicitly specifying the `options` parameter, as this may result in unpredictable timer behavior.

## **Resetting Boot Override to Default**

To clear the boot override and return to the default boot device, use:

```
ipmitool chassis bootparam set bootflag none
```

## **Setting Persistent PXE Boot**

To configure the system to always boot from PXE (persistent override), execute:

```
ipmitool chassis bootdev pxe options=persistent
```

### **i Info**

The persistent option prevents the 60-second timeout from being triggered.

### **i Info**

If you modify bootdev or bootparam settings without explicitly specifying `options=persistent`, the persistent configuration will be disabled.

## **Behavior of Boot Source Override on BlueField**

The Boot Source Override configuration set through the BMC remains persistent until one of the following occurs:

- It is explicitly reset to `none`.
- The BFB image is updated, which will clear the override settings.

## 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