



BIOS Secure Boot Configuration

Table of contents

Reading Secure Boot Status

Setting Secure Boot State

Secure Boot Database Support

Secure Boot Flow Example

The NVIDIA® BlueField® BMC supports the DMTF Secure Boot schema which enables managing the state of the UEFI Secure Boot through the Redfish interface. This allows clients to set whether UEFI should authenticate the OS image during the boot process.

Reading Secure Boot Status

```
curl -k -u root:'<password>' -H 'Content-Type: application/json'
-X GET https://<bmc_ip>/redfish/v1/Systems/Bluefield/SecureBoot
```

Output example:

```
{
  "@odata.id": "/redfish/v1/Systems/Bluefield/SecureBoot",
  "@odata.type": "#SecureBoot.v1_1_0.SecureBoot",
  "Description": "The UEFI Secure Boot associated with this
system.",
  "Id": "SecureBoot",
  "Name": "UEFI Secure Boot",
  "SecureBootCurrentBoot": "Disabled",
  "SecureBootDatabases": {
    "@odata.id":
"/redfish/v1/Systems/Bluefield/SecureBoot/SecureBootDatabases"
  },
  "SecureBootEnable": false,
  "SecureBootMode": "SetupMode"
}
```

Setting Secure Boot State

The following command enables UEFI Secure Boot through the Redfish interface:

```
curl -k -u root:'<password>' -X PATCH -H "Content-Type: application/json" https://<bmc_ip>/redfish/v1/Systems/Bluefield' {"SecureBootEnable":true}'
```

The following command disables UEFI Secure Boot through the Redfish interface:

```
curl -k -u root:<password> -H "Content-Type: application/octet-stream" -X GET https://<BF-BMC-IP>/redfish/v1/Systems/Bluefield/SecureBoot {
  "@odata.id": "/redfish/v1/Systems/Bluefield/SecureBoot",
  "@odata.type": "#SecureBoot.v1_1_0.SecureBoot",
  "Description": "The UEFI Secure Boot associated with this system.",
  "Id": "SecureBoot",
  "Name": "UEFI Secure Boot",
  "SecureBootCurrentBoot": "Enabled",
  "SecureBootEnable": true,
  "SecureBootMode": "SetupMode"
}
curl -k -u root:<BF-BMC-PASSWORD> -X PATCH https://<BF-BMC-IP>/redfish/v1/Systems/Bluefield/SecureBoot -H 'Content-Type: application/json' -d '{"SecureBootEnable": false}'
```

After running this command, the BlueField Arm OS must be rebooted twice. The first reboot is for the UEFI redfish client to read the request from the BMC and apply it; the second reboot is for the setting to take effect.

i Note

The "SecureBootEnable" property in secure boot schema takes precedence over UEFI menu BlueField Arm OS settings. This is because currently there is no separate pending setting URL and hence the value of "SecureBootEnable" property will be applied on BlueField Arm OS reboot.

- From the BlueField BMC using Redfish:

```
curl -k -u root:<BF-BMC-PASSWORD> -X POST https://<BF-BMC-IP>/redfish/v1/Systems/Bluefield/Actions/ComputerSystem.Reset -H 'Content-Type: application/json' -d '{"ResetType": "ForceRestart"}'
```

- From RShim:

```
echo 'SW_RESET 1' > /dev/rshim0/misc
```

- From the BlueField Arm OS:

```
reboot
```

Secure Boot Database Support

The following operations may be performed using Redfish commands. For each operation, a corresponding task is generated within the BMC's Redfish Task Service. During the subsequent BlueField reboot, the UEFI checks for any pending secure boot tasks and executes them in the order of their ascending task ID numbers. After completion, the UEFI then updates the task state to reflect the relevant status.

- To read UEFI Secure boot databases:

```
curl -k -u root:'<password>' -H 'Content-Type:
application/json' -X GET
https://<bmc_ip>/redfish/v1/Systems/Bluefield/SecureBoot/Secure
```

Output example:

```

{
  "@odata.id" :
"/redfish/v1/Systems/Bluefield/SecureBoot/SecureBootDatabases"
  "@odata.type" :
"#SecureBootDatabaseCollection.SecureBootDatabaseCollection",
  "Members" : [
    {
      "@odata.id" :
"/redfish/v1/Systems/Bluefield/SecureBoot/SecureBootDatabases/
    },
    ..
    {
      "@odata.id" :
"/redfish/v1/Systems/Bluefield/SecureBoot/SecureBootDatabases/
    },
    ..
    {
      "@odata.id" :
"/redfish/v1/Systems/Bluefield/SecureBoot/SecureBootDatabases/
    },
    ..
    {
      "@odata.id" :
"/redfish/v1/Systems/Bluefield/SecureBoot/SecureBootDatabases/
    },
    ..
  ],
  "Members@odata.count" : 10,
  "Name" : "UEFI SecureBoot Database Collection"
}

```

- To add a certificate to the UEFI `db`:

Note

The following certificate is an example only. The content of the PEM file is copied into the `curl` POST command; `\n` must be used to replace EOL characters. For illustration purposes here's the original content of the PEM certificate file.

```
-----BEGIN CERTIFICATE-----
MIIDbTCCA1WgAwIBAgIU02MdJt2cTCGr0e04PiBV5Uk0b/IwDQYJKoZIh
BQAwVjELMAkGA1UEBhMCMVVMxCzAJBgNVBAGTAk5DMRAwDgYDVQQHEwdSY
MQ8wDQYDVQQKEwZMZW5vdm8xZzAVBgNVBAMTDkxlbm92byBVRUZJIERCM
MDMxNTIxMTYzNFoXDTQxMDMxNTIxMTYzNFowVjELMAkGA1UEBhMCMVVMxC
BAGTAk5DMRAwDgYDVQQHEwdSYWxlaWdoMQ8wDQYDVQQKEwZMZW5vdm8xZ
BAMTDkxlbm92byBVRUZJIERCMIIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AM
AQEA1ezmdoBlk5yDVuXw8F774Cv1lhMz9bT0/TpH3kmRpPAizSRDzbHLL
W4zjdLxTq0lwZt6UUcWxlyzKcoDPe43cE6YH1kM/rscvm3AaVL+4GcyGg
QFHWER25xCTokMsCdKB42Ty7hWW5FBPepgAS+GDfqQfb/4hoonIlen5X+
RM1DIVBUiIbJdgERYeoGjY/Rh4A1VWl6ErzyzokYnf63JjSFR2kVV0apt
7qBd1RNHWqrcARyRADX1XGvRZURzwQdEXf0qZ0kVjNkr1fD761qvPE8TC
mciMocIXqoqWKPakgbMwKmcSFQIDAQABozMwMTAPBgNVHRMBAf8EBTADA
A1UdDwEB/wQEAwICBDA0BgNVHQ8BAf8EBAMCB4AwDQYJKoZIhvcNAQELE
AJ2U0UjB+sxF/HE5sY56vJbdFIT18o0Yf7XJImL0VtgpYjfeqiE768G2u
hD0ps3+4w4p8FUS06StzCz6UuUyx1UjQzpkxZ970uq1sGhjy7dZybTEBy
l1EpJSfBiwxTdm7svJoABKs8Hs7e9f3XX5PK76Sx1lMbDaxAm7UvCppYE
gWt3rGRi03W6pfd07ioCD03kgGzYNOZeU2S+maE1Xt4kUoYs3HxyrhJGf
4w5LfCKr1xi+3KMf+vXxEBfGYBvjwcA7KCW92GnUQGVjZbEGs6EaTBx7i
oWS/500qiwNRp2xqdBxg1d0=
-----END CERTIFICATE-----
```

```
curl -k -u root:'<password>' -H 'Content-Type:
application/json' -X POST
https://<bmc_ip>/redfish/v1/Systems/Bluefield/SecureBoot/SecureBoot
-d\
'{"CertificateString": "-----BEGIN CERTIFICATE-----
\nMIIDbTCCA1WgAwIBAgIU02MdJt2cTCGr0e04PiBV5Uk0b/IwDQYJKoZIhvcN
-----END CERTIFICATE-----", "CertificateType": "PEM":
"5491316d-9694-4639-b72d-b8630ffa7dab" }'
```

Output example:

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

- To add a signature to the UEFI `db`:

```
curl -k -u root:'<password>' -H 'Content-Type:
application/json' -X POST
https://<bmc_ip>/redfish/v1/Systems/Bluefield/SecureBoot/SecureBoot
-d\
'{"SignatureString":
"80B4D96931BF0D02FD91A61E19D14F1DA452E66DB2408CA8604D411F92659
"UEFI", "SignatureType": "EFI_CERT_SHA256_GUID": "28d5e212-
165b-4ca0-909b-c86b9cee0112" }'
```

Output example:

```
{
  "@odata.id": "/redfish/v1/TaskService/Tasks/1",
  "@odata.type": "#Task.v1_4_3.Task",
  "Id": "1",
  "TaskState": "Pending",
  "TaskStatus": "OK"
}
```

- To delete UEFI `db` certificate #1:

```
curl -k -u root:'<password>' -H 'Content-Type:
application/json' -X DELETE
https://<bmc_ip>/redfish/v1/Systems/Bluefield/SecureBoot/Secure
```

Output example:

```
{
  "@odata.id": "/redfish/v1/TaskService/Tasks/2",
  "@odata.type": "#Task.v1_4_3.Task",
  "Id": "2",
  "TaskState": "Pending",
  "TaskStatus": "OK"
}
```

- To delete all UEFI `db` keys:

```
curl -k -u root:'<password>' -H 'Content-Type:
application/json' -X POST
https://<bmc_ip>/redfish/v1/Systems/Bluefield/SecureBoot/SecureBoot/SecureBoot
-d' {"ResetKeyType": "DeleteAllKeys"}
```

Output example:

```
{
  "@odata.id": "/redfish/v1/TaskService/Tasks/3",
  "@odata.type": "#Task.v1_4_3.Task",
  "Id": "3",
  "TaskState": "Pending",
  "TaskStatus": "OK"
}
```

Secure Boot Flow Example

The following is an example flow for deleting **PK** certificate using Redfish commands. This command would disable UEFI Secure Boot and revert the system to **Setup Mode**.

1. To reset all **db** keys:

```
root:~# curl -k -u root:'<password>' -H 'Content-Type:
application/json' -X POST
https://<bmc_ip>/redfish/v1/Systems/Bluefield/SecureBoot/SecureBoot/SecureBoot
-d' {"ResetKeyType": "DeleteAllKeys"}
```

Output example:

```
{
  "@odata.id": "/redfish/v1/TaskService/Tasks/12",
  "@odata.type": "#Task.v1_4_3.Task",
  "Id": "12",
  "TaskState": "Pending",
  "TaskStatus": "OK"
}
```

Tip

Record the returned task ID, in this example the task ID is 12.

2. To read the status of task 12:

```
root:~# curl -k -u root:'<password>' -H 'Content-Type: application/json' -X
GET https://<bmc_ip>/redfish/v1/TaskService/Tasks/12
```

Output example:

```

{
  "@odata.id": "/redfish/v1/TaskService/Tasks/12",
  "@odata.type": "#Task.v1_4_3.Task",
  "Id": "12",
  "Messages": [],
  "Name": "Task 12",
  "Payload": {
    "HttpHeaders": [
      "Host: <IP>",
      "User-Agent: curl/7.81.0",
      "Accept: */*",
      "Content-Length: 34"
    ],
    "HttpOperation": "POST",
    "JsonBody": "{\n  \"ResetKeyType\":\n  \"DeleteAllKeys\"\n}",
    "TargetUri":
"/redfish/v1/Systems/Bluefield/SecureBoot/SecureBootDatabases/
  },
    "PercentComplete": 0,
    "StartTime": "2023-09-05T16:47:05+00:00",
    "TaskMonitor": "/redfish/v1/TaskService/Tasks/12/Monitor",
    "TaskState": "Pending",
    "TaskStatus": "OK"
  }
}

```

You can see that `TaskStatus` is `OK` and the `TaskState` is `Pending`. This indicates that the operation has successfully enqueued in the task service and is pending the next BlueField boot.

3. Issue the following graceful reset command to BlueField :

```
root:~# curl -k -u root:"<password>" -H "Content-Type:
application/json" -X POST
https://<bmc_ip>/redfish/v1/Systems/Bluefield/Actions/Computer
-d '{"ResetType" : "GracefulRestart"}'
```

Output example:

```
{
  "@Message.ExtendedInfo": [
    {
      "@odata.type": "#Message.v1_1_1.Message",
      "Message": "The request completed successfully.",
      "MessageArgs": [],
      "MessageId": "Base.1.15.0.Success",
      "MessageSeverity": "OK",
      "Resolution": "None"
    }
  ]
}
```

UEFI reads the pending secure boot tasks and executes them.

4. Following BlueField reset, the UEFI updates the status of the operation on the `TaskState` and `TaskStatus` fields. [Poll the task](#) and check the values of `TaskState` and `TaskStatus`.

Success	<pre>"TaskState": "Completed", "TaskStatus": "OK"</pre>
---------	---

Failure	<pre>"TaskState": "Exception", "TaskStatus": "OK"</pre>
---------	---

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