



## **DOCA UROM Service Guide**

# Table of contents

Introduction

---

Requirements

---

Service Deployment

---

Description

---

Plugin Discovery and Reporting

---

Loading Plugin in Worker

---

Yaml File

---

Troubleshooting

---

Pod is Marked as "Ready" and No Container is Listed

---

Error

---

Solution

---

Pod is Not Listed

---

Error

---

Solution

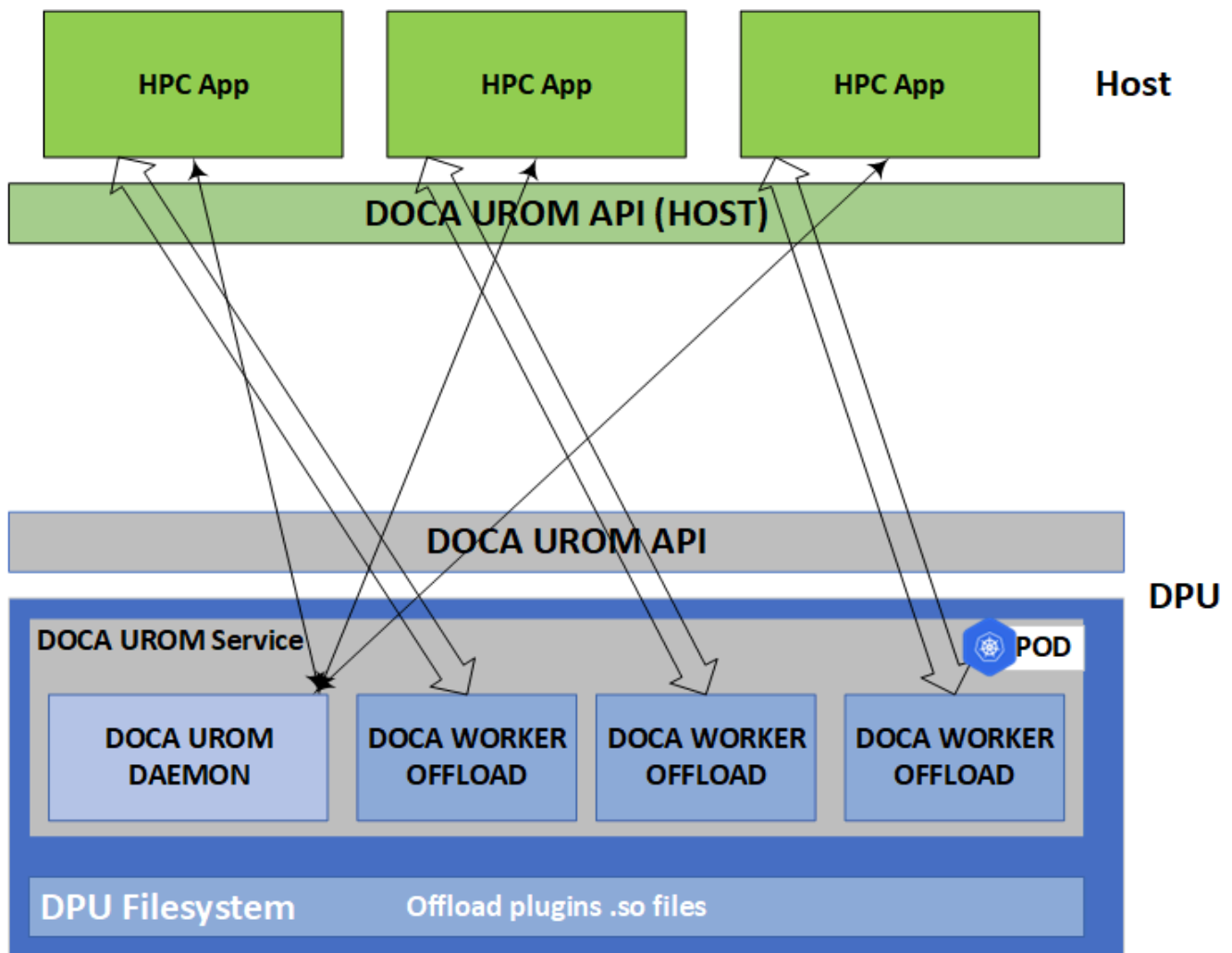
---

This guide provides instructions on how to use the DOCA UROM Service on top of the NVIDIA® BlueField® networking platform.

## Introduction

The DOCA UROM service provides a framework for offloading significant portions of HPC software stack directly from the host and to the BlueField device.

Using a daemon, the service handles the discovery of resources, the coordination between the host and BlueField, and the spawning, management, and teardown of the BlueField workers themselves.



The first step in initiating an offload request involves the UROM host application establishing a connection with the UROM service. Upon receiving the plugin discovery command, the UROM service responds by providing the application with a list of plugins available on the BlueField. The application then attaches the plugin IDs that correspond to the desired workers to their network identifiers. Finally, the service triggers UROM worker

plugin instances on the BlueField to execute the parallel computing tasks. Within the service's Kubernetes pod, workers are spawned by the daemon in response to these offload requests. Each computation can utilize either a single library or multiple computational libraries.

## Requirements

Before deploying the UROM service container, ensure that the following prerequisites are satisfied:

- Allocate huge pages as needed by DOCA:

```
$ echo '2048' | sudo tee -a /sys/kernel/mm/hugepages/hugepages-2048kB/nr_hugepages
$ sudo mkdir /mnt/huge
$ sudo mount -t hugetlbfs -o pagesize=2M nodev /mnt/huge
```

## Service Deployment

For information about the deployment of DOCA containers on top of the BlueField, refer to the [NVIDIA BlueField Container Deployment Guide](#).

Service-specific configuration steps and deployment instructions can be found under the service's [container page](#).

## Description

### Plugin Discovery and Reporting

When the application initiates a connection request to the DOCA UROM Service, the daemon reads the `UROM_PLUGIN_PATH` environment variable. This variable stores directory paths to `.so` files for the plugins with multiple paths separated by semicolons. The daemon scans these paths sequentially and tries loading each `.so` file. Once the daemon finishes the scan, it reports the available BlueField plugins to the host application.

The host application gets the list of available plugins as a list of `doca_urom_service_plugin_info` structures:

```

struct doca_urom_service_plugin_info {
    uint64_t id;           // Unique ID to send commands to the plugin
    uint64_t version;     // Plugin version
    char plugin_name[DOCA_UROM_PLUGIN_NAME_MAX_LEN]; // .so filename
};

```

The UROM daemon is responsible for generating unique identifiers for the plugins, which are necessary to enable the worker to distinguish between different plugin tasks.

## Loading Plugin in Worker

During the spawning of UROM workers by the UROM daemon, the daemon attaches a list of desired plugins in the worker command line. Each plugin is passed in a format of `so_path:id`.

As part of worker bootstrapping, the flow iterates all `.so` files and tries to load them by using `dlopen` system call and look for `urom_plugin_get_iface()` symbol to get the plugin operations interface.

## Yaml File

The `.yaml` file downloaded from NGC can be easily edited according to users' needs:

```

env:
  # Service-Specific command line arguments
  - name: SERVICE_ARGS
    value: "-l 60 -m 4096"
  - name: UROM_PLUGIN_PATH
    value:
"/opt/mellanox/doca/samples/doca_urom/plugins/worker_sandbox;/opt/mellanox/doca/samples/doca_urom/

```

- The `SERVICE_ARGS` are the runtime arguments received by the service:
  - `-l`, `--log-level <value>` – sets the (numeric) log level for the program  
<10=DISABLE, 20=CRITICAL, 30=ERROR, 40=WARNING, 50=INFO, 60=DEBUG, 70=TRACE>
  - `--sdk-log-level` – sets the SDK (numeric) log level for the program  
<10=DISABLE, 20=CRITICAL, 30=ERROR, 40=WARNING, 50=INFO, 60=DEBUG, 70=TRACE>
  - `-m`, `--max-msg-size` – specify UROM communication channel maximum message size
- The `UROM_PLUGIN_PATH` is an env variable that stores directory paths to `.so` files for the plugins

For each plugin on the BlueField, it is necessary to add a volume mount inside the service container. For example:

```
volumes:
  - name: urom-sandbox-plugin
    hostPath:
      path:
        /opt/mellanox/doca/samples/doca_urom/plugins/worker_sandbox
      type: DirectoryOrCreate
  ...
volumeMounts:
  - mountPath:
    /opt/mellanox/doca/samples/doca_urom/plugins/worker_sandbox
    name: urom-sandbox-plugin
```

## Troubleshooting

When troubleshooting a container deployment issues, it is highly recommended to follow the deployment steps and tips found in the "[Review Container Deployment](#)" section of the [DOCA Container Deployment Guide](#) .

One could also check the `/var/log/doca/urom` log files for more details about the running cycles of service components (daemon and workers).

The log file name for workers is `urom_worker_<pid>_dev.log` and for the daemon it is `urom_daemon_dev.log`.

## Pod is Marked as "Ready" and No Container is Listed

### Error

When deploying the container, the pod's STATE is marked as `Ready` and an image is listed, however, no container can be seen running:

```
$ sudo crictl pods
POD ID          CREATED          STATE          NAME
NAMESPACE      ATTEMPT         RUNTIME
3162b71e67677  4 seconds ago  Ready
doca-urom-my-dpu  default
0              (default)

$ sudo crictl images
IMAGE          TAG
IMAGE ID      SIZE
k8s.gcr.io/pause  3.2
  2a060e2e7101d  487kB
nvcv.io/nvidia/doca/doca_urom  1.0.0-doca2.7.0
2af1e539eb7ab  86.8MB

$ sudo crictl ps
CONTAINER      IMAGE          CREATED          STATE
NAME          ATTEMPT       POD ID
POD
```

### Solution

In most cases, the container did start but immediately exited. This could be checked using the following command:

```
$ sudo crictl ps -a
```

CONTAINER NAME	IMAGE	CREATED	STATE
556bb78281e1d	2af1e539eb7ab	6 seconds ago	Exited
3162b71e67677	doca-urom-my-dpu		

Should the container fail (i.e., reporting a state of `Exited`), it is recommended to examine the UROM's main log at `/var/log/doca/urom/urom_daemon_dev.log`.

In addition, for a short period of time after termination, the container logs could also be viewed using the container's ID:

```
$ sudo crictl logs 556bb78281e1d
...
```

## Pod is Not Listed

### Error

When placing the container's YAML file in the Kubelet's input folder, the service pod is not listed in the list of pods:

```
$ sudo crictl pods
```

POD ID	CREATED	STATE	NAME
	ATTEMPT	RUNTIME	



## Solution

In most cases, the pod has not started because of the absence of the requested hugepages. This can be verified using the following command:

```
$ sudo journalctl -u kubelet -e. . .
Oct 04 12:12:19 <my-dpu> kubelet[2442376]: I1004 12:12:19.905064
2442376 predicate.go:103] "Failed to admit pod, unexpected error while attempting to
recover from admission failure" pod="default/doca-urom-service-<my-dpu>" err="preemption: error
finding a set of pods to preempt: no set of running pods found to reclaim resources: [(res: hugepages-2Mi,
q: 104563999874), ]"
```

**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 NONINFRINGEMENT, 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 2025, NVIDIA. PDF Generated on 05/05/2025