NVIDIA DOCA IPS

Application Guide

MLNX-15-060464 _v2.2.0 | October 2023

Table of Contents

Chapter 1. Introduction

Important: No updates were made to the DOCA IPS application in DOCA 2.2. Please refer to DOCA 2.5 for a note regarding future updates.

Intrusion prevention system (IPS) is an application that monitors a network for malicious activity or policy violations.

IPS uses the deep packet inspection (DPI) engine to scan network flow for malicious content based on predefined Suricata signatures. Packets that are deemed malicious are dropped and a corresponding message is printed.

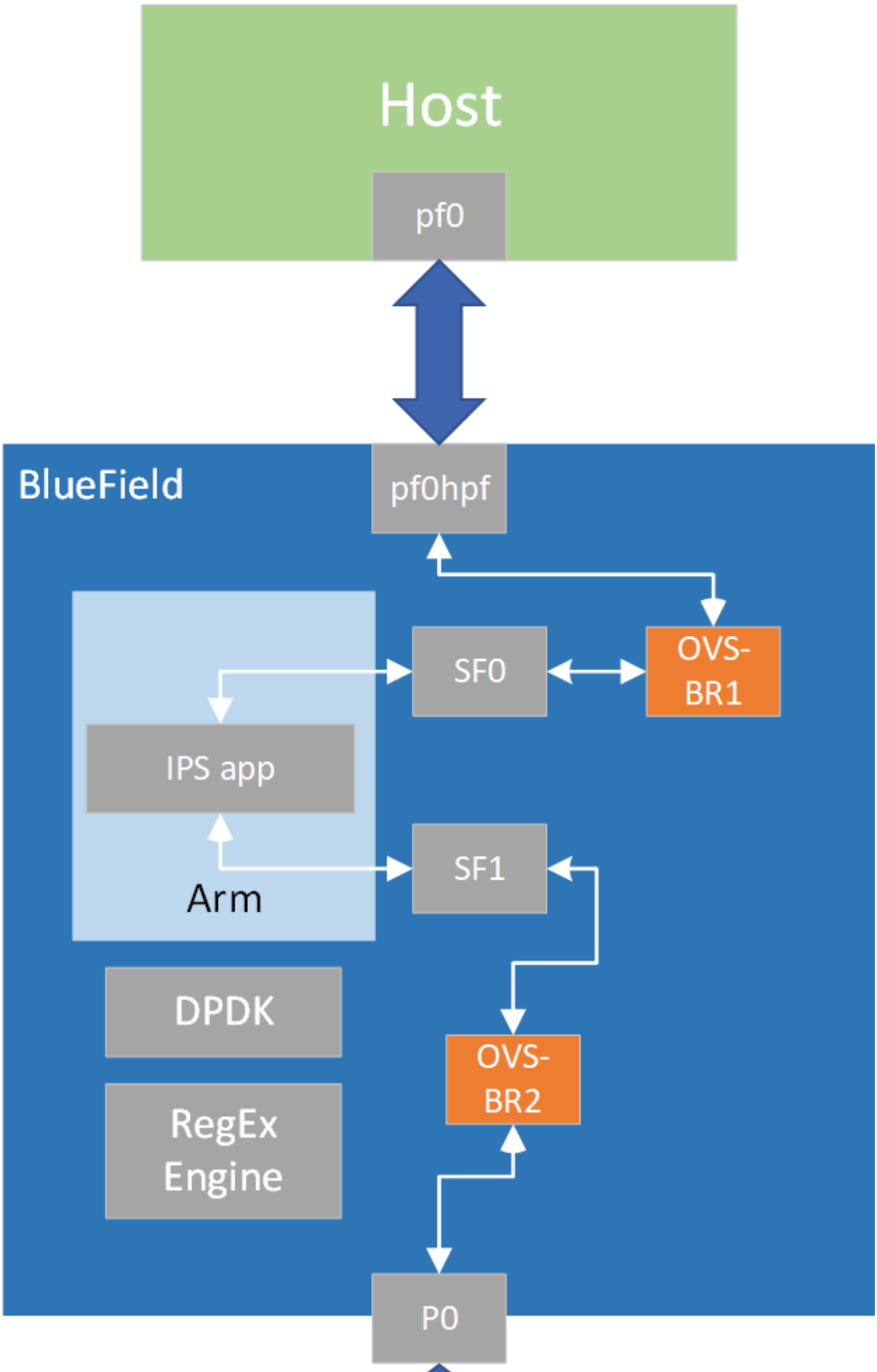
IPS supports NetFlow protocol for sending data from the DPU to remote NetFlow collector for further analysis.

Connection tracking is also supported for tracking all network connections or flows which helps the identification of all the packets that make up a flow for better handling of the network traffic.

This document describes how to build and run the IPS application both on the host and on the DPU.

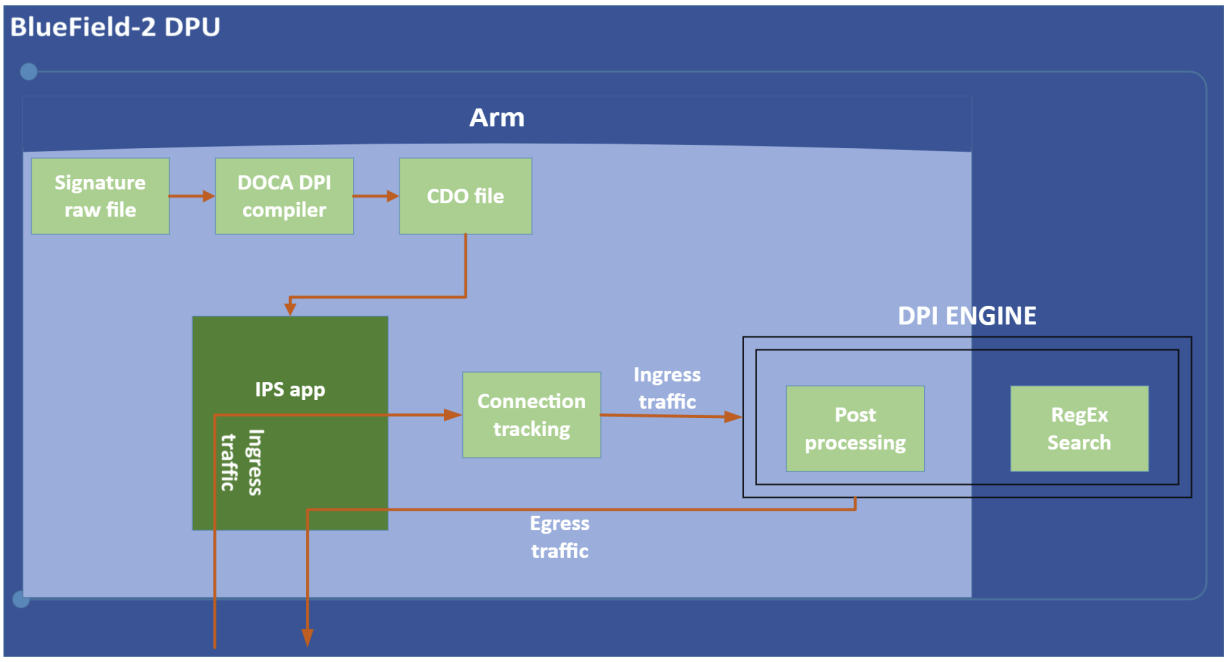
Chapter 2. System Design

The IPS application is designed to run as "bump-on-the-wire" on the BlueField instance, it intercepts the traffic coming from the wire, and passes it to peer port.


Host
pf0
BlueField
pf0hpf
SF0
OVS-
BR1
IPS app
Arm
SF1
DPDK
RegEx
Engine
OVS-
BR2
P0

Chapter 3. Application Architecture

IPS runs on top of DPDK-based stateful flow tracking (SFT) to identify the flow that each packet belongs to, then uses DPI to process L7 classification.



1. Signatures are compiled by DPI compiler and then loaded to DPI engine. See DOCA DPI Compiler for more information.
2. Ingress traffic is identified using the stateful table module which utilizes the connection tracking hardware offloads.
3. Traffic is scanned against DPI-engine-compiled signature DB.
4. Post-processing is performed for match decision.
5. Matched flows are identified and drop actions can be offloaded to the hardware to increase performance as no further inspection is needed.
6. Flow termination is done by a configurable aging timer set in the SFT to 60 seconds. When a flow is offloaded, it cannot be tracked and destroyed.

Chapter 4. DOCA Libraries

This application leverages the following DOCA libraries:

- DOCA DPI Library
- DOCA Telemetry Library

Chapter 5. Configuration Flow

1. Parse application argument.
   a). Initialize Arg Parser resources and register DOCA general parameters.
   ```
   doca_argp_init();
   ```
   b). Register application parameters.
   ```
   register_ips_params();
   ```
   c). Parse the arguments.
   ```
   doca_argp_start();
   ```
   i. Parse DPDK flags and invoke handler for calling the `rte_eal_init()` function
   ii. Parsing app parameters.
2. DPDK initialization.
   ```
   dpdk_init();
   ```
   Calls `rte_eal_init()` to initialize EAL resources with the provided EAL flags.
3. DPDK port initialization and start.
   ```
   dpdk_queues_and_ports_init();
   ```
   a). Initialize SFT.
   b). Initialize DPDK ports, including mempool allocation.
4. Initialize IPS application resources including DPI engine and NetFlow.
   ```
   ips_init();
   ```
5. Configure DPI packet processing.
   ```
   ips_worker_lcores_run();
   ```
   a). Configure DPI enqueue packets.
   b). Send jobs to RegEx engine.
   c). Configure DPI dequeue packets.
6. If Netflow is enabled.
   ```
   send_netflow_record();
   ```
7. IPS destroy.
   ```
   ips_destroy();
   ```
   a). Stop and free DPI resources.
   b). Destroy netflow resources.
   c). Stop SFT.
   d). Free IPS resources.

8. DPDK ports and queues destruction.
   ```
   dpdk_queues_and_ports_fini();
   ```
9. DPDK finish.
   ```
   dpdk_fini();
   ```
   Calls `rte_eal_destroy()` to destroy initialized EAL resources.
10. Arg parser destroy.
    ```
    doca_argp_destroy();
    ```

Chapter 6. Running the Application

1. Refer to the following documents:
   - NVIDIA DOCA Installation Guide for Linux for details on how to install BlueField-related software.
   - NVIDIA DOCA Troubleshooting Guide for any issue you may encounter with the installation, compilation, or execution of DOCA applications.
   - NVIDIA DOCA Applications Overview for additional compilation instructions and development tips for the DOCA applications.
2. The IPS application binary is located under `/opt/mellanox/doca/applications/ips/bin/doca_ips`. To build all the applications, run:

```
cd /opt/mellanox/doca/applications/
meson build
ninja -C build
```

3. To build the IPS application only:

   a). Edit the following flags in `/opt/mellanox/doca/applications/meson_options.txt`:

   - Set `enable_all_applications` to `false`
   - Set `enable_ips` to `true`

   b). Run the commands in step 2.

> Note: `doca_ips` is created under `./build/ips/src/`.

Application usage:

```
Usage: doca_ips [DPDK Flags] -- [DOCA Flags] [Program Flags]

DOCA Flags:
  -h, --help                        Print a help synopsis
  -v, --version                     Print program version information
  -l, --log-level                   Set the log level for the program
 <CRITICAL=20, ERROR=30, WARNING=40, INFO=50, DEBUG=60>

Program Flags:
  -p, --print-match                 Prints FID when matched in DPI engine
  -n, --netflow <source_id>         Collect netflow statistics and set
 source_id if value is set
  -o, --output-csv <path>           Path to the output of the CSV file
  -c, --cdo <path>                  Path to CDO file compiled from a valid PDD
  -f, --fragmented                  Enables processing fragmented packets
```

```
 -a, --pci-addr                      DOCA DPI device PCI address
```

Note: For additional information on available flags for DPDK, use `-h` before the `--` separator:

```
/opt/mellanox/doca/applications/ips/bin/doca_ips -h
```

Note: For additional information on the application, use `-h` after the `--` separator:

```
/opt/mellanox/doca/applications/ips/bin/doca_ips -- -h
```

4. Running the application on BlueField:
   - Pre-run setup.
     a). The IPS example is based on DPDK libraries. Therefore, the user is required to provide DPDK flags and allocate huge pages. Run:
     ```
     sudo echo 2048 > /sys/kernel/mm/hugepages/hugepages-2048kB/nr_hugepages
     ```
     b). Make sure the RegEx engine is active:
     ```
     systemctl status mlx-regex
     ```
     If the status is inactive (`Active: failed`), run:
     ```
     systemctl start mlx-regex
     ```
   - CLI example for running the app:

     Note: Make sure to compile signature before running the application. For more information, please refer to NVIDIA DOCA DPI Compiler.

     ```
     /opt/mellanox/doca/applications/ips/bin/doca_ips -a 0000:03:00.0,class=regex -a auxiliary:mlx5_core.sf.4,sft_en=1 -a auxiliary:mlx5_core.sf.5,sft_en=1 -- --cdo /root/ips.cdo -p -n
     ```

     Note: The SFT supports a maximum of 64 queues. Therefore, the application cannot be run with more than 64 cores. To limit the number of cores, run:

     ```
     /opt/mellanox/doca/applications/ips/bin/doca_ips -a
      0000:03:00.0,class=regex -a auxiliary:mlx5_core.sf.4,sft_en=1 -a
      auxiliary:mlx5_core.sf.5,sft_en=1 -l 0-64 -- --cdo /root/ips.cdo -p -n
     ```

     This limits the application to use 65 cores (core-0 to core-64). That is 1 core for the main thread and 64 cores to serve as workers.

     Note: The flags `-a 0000:03:00.0,class=regex -a auxiliary:mlx5_core.sf.4,sft_en=1 -a auxiliary:mlx5_core.sf.5,sft_en=1` are necessary for proper usage of the application. Modifying these flags results in unexpected behavior as only 2 ports are supported. The SF numbers are arbitrary and configurable. The RegEx device, however, is not and must be initiated on port 0.

     Note: Sub-functions must be enabled according to Scalable Function Setup Guide.

5. Running the application on the host, CLI example:
   ```
   cd /opt/mellanox/doca/applications/ips/
   ```

```
./doca_ips-a 0000:21:00.0,class=regex -a 0000:21:00.3 -a 0000:21:00.4 -- --cdo ~/
ips.cdo
```

Note: Refer to section "Running DOCA Application on Host" in NVIDIA DOCA Virtual Functions User Guide.

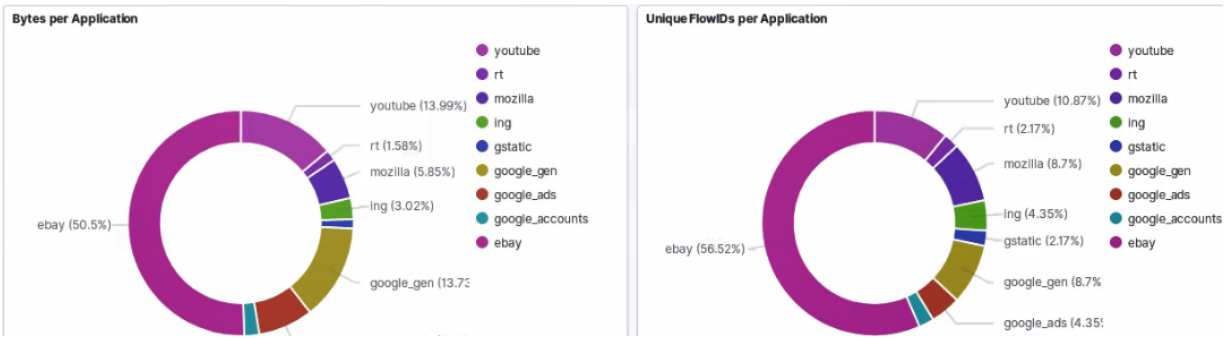
6. To run `doca_ips` using a JSON file:

   ```
   doca_ips --json [json_file]
   ```

   For example:

   ```
   cd /opt/mellanox/doca/applications/ips/bin
   ./doca_ips --json ips_params.json
   ```

NetFlow collector UI example:



The NetFlow module uses the DOCA's Telemetry NetFlow library to export NetFlow packets in the NetFlow v9 format. The usage of telemetry is hardcoded to send packets to a collector set on the host connected to the Bluefield device through the RShim interface, 192.168.100.2:2055.

It is recommended to use the DOCA telemetry service as an aggregator service to export records instead of exporting directly from the client side which requires enabling IPC.

Refer to the NVIDIA DOCA Telemetry Service Guide for more information.

Chapter 7. Arg Parser DOCA Flags

Refer to NVIDIA DOCA Arg Parser Programming Guide for more information.

Flag Type	Short Flag	Long Flag/ JSON Key	Description	JSON Content
DPDK flags	a	devices	Adds a PCIe device into the list of devices to probe	"devices": [{"device": "regex", "id": "00 {"device": "sf", "id": "4","s true}, {"device": "sf", "id": "5","s true},]
	l	core-list	Lists cores to run on	"core-list": "0-4"
General flags	l	log-level	Sets the log level for the application: ▶ CRITICAL=20 ▶ ERROR=30 ▶ WARNING=40 ▶ INFO=50 ▶ DEBUG=60	"log-level": 60
	v	version	Print program version information	N/A
	h	help	Prints a help synopsis	N/A
Program flags	p	print-match	Prints FID when matched in DPI engine	"print-match": true
	n	netflow	Exports data from BlueField to remote DTS, IP is set to 192.168.100.2	"netflow": 0

Flag Type	Short Flag	Long Flag/ JSON Key	Description	JSON Content
			which is the host's IP using the RShim interface. Also sets `source_id` to be written to the NetFlow packet.	
	`o`	`output-csv`	Path to the output of the CSV file	`"output-csv": "/tmp/ips_stats.csv"`
	`c`	`cdo`	Path to CDO file compiled from a valid PDD Note: This flag is mandatory.	`"cdo": "/tmp/ips.cdo"`
	`f`	`fragmented`	Enables processing fragmented packets	`"fragmented": false`

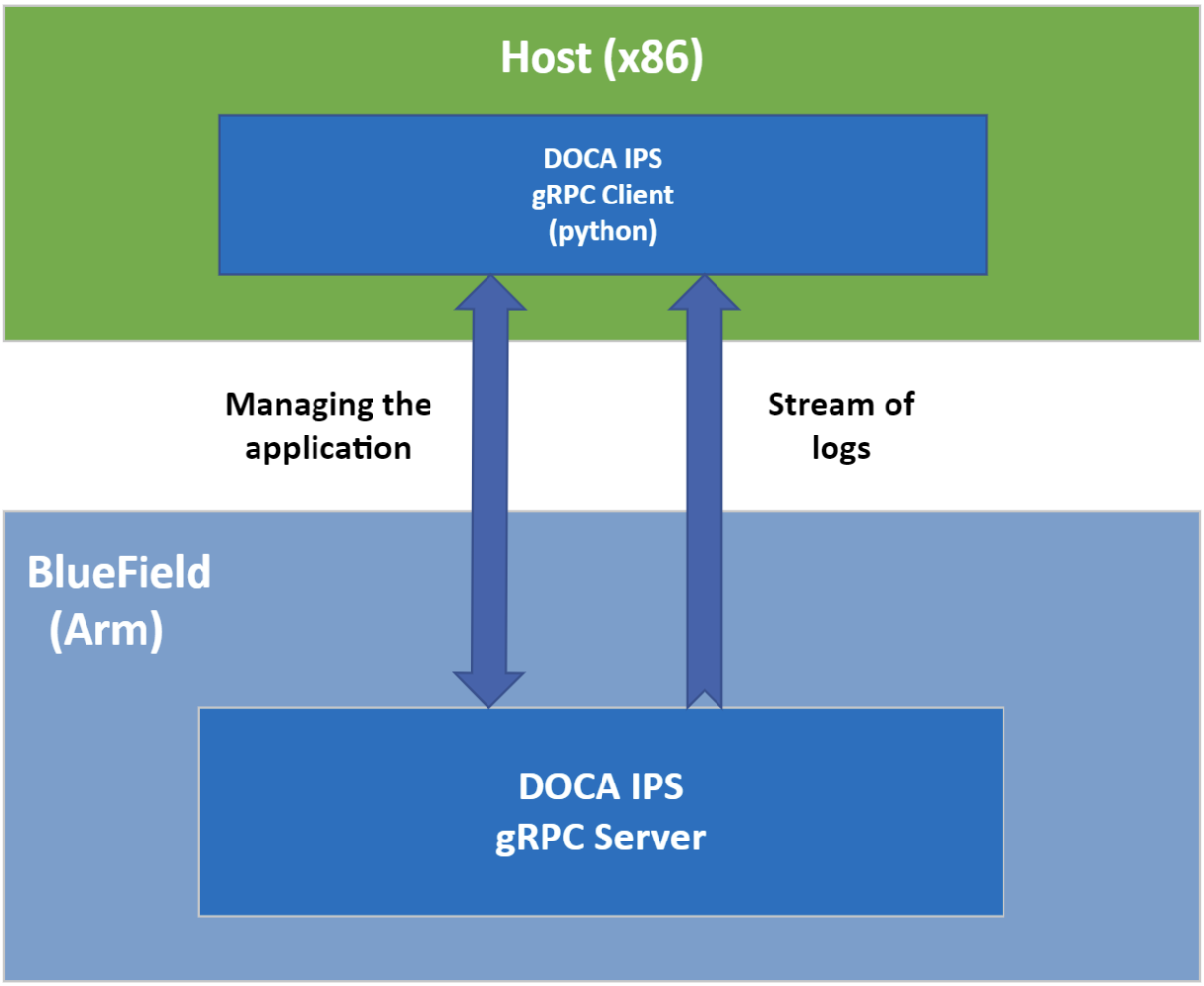
Chapter 8. Deploying Containerized Application

The IPS example supports a container-based deployment. Refer to the NVIDIA DOCA Container Deployment Guide for more information.

Application-specific configuration steps may be found on NGC under the application's container page.

Chapter 9. Managing gRPC-Enabled Application from Host

For instructions on running the gRPC application server on BlueField, refer to NVIDIA DOCA gRPC Infrastructure User Guide.



To run the Python client of the gRPC-enabled application:

```
./doca_ips_gRPC_client.py -d/--debug <server address[:server port]>
```

For example:

```
/opt/mellanox/doca/examples/ips/bin/grpc/client/doca_ips_gRPC_client.py
 192.168.104.2
```

Chapter 10. References

- /opt/mellanox/doca/applications/ips/src
- /opt/mellanox/doca/applications/ips/src/grpc/ips.proto
- /opt/mellanox/doca/applications/ips/bin/ips_suricata_rules_example

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 nor any of its direct or indirect subsidiaries and affiliates (collectively: “NVIDIA”) make no representations or warranties, expressed or implied, as to the accuracy or completeness of the information contained in this document and assume 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, the NVIDIA logo, and Mellanox are trademarks and/or registered trademarks of Mellanox Technologies Ltd. and/or NVIDIA Corporation in the U.S. and in other countries. The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a world¬wide basis. Other company and product names may be trademarks of the respective companies with which they are associated.

Copyright

NVIDIA Corporation | 2788 San Tomas Expressway, Santa Clara, CA 95051
http://www.nvidia.com