NVIDIA DOCA NetFlow Exporter

Reference Application Guide

MLNX-15-060473 _v1.1 | August 2021

Table of Contents

Chapter 1. Introduction

NetFlow is a protocol used for exporting device network flows information to a NetFlow collector. A flow refers to any connection or connection-like communication channel between two communication nodes.

The most common definition of a flow is by a standard 5-tuple.

Using NetFlow requires three network nodes:

- NetFlow exporter – a network device in charge of collecting flow information and exporting it to a flow collector
- NetFlow collector – a server that receives exported flow information
- NetFlow analyzer – an application which runs on the collector and analyzes flow information collected by the flow collector

Using a NetFlow monitoring solution allows monitoring and analyzing these flow records more efficiently and effectively for traffic within the network.

Netflow main use-case:

- Bandwidth utilization and network resource allocation (QoS)

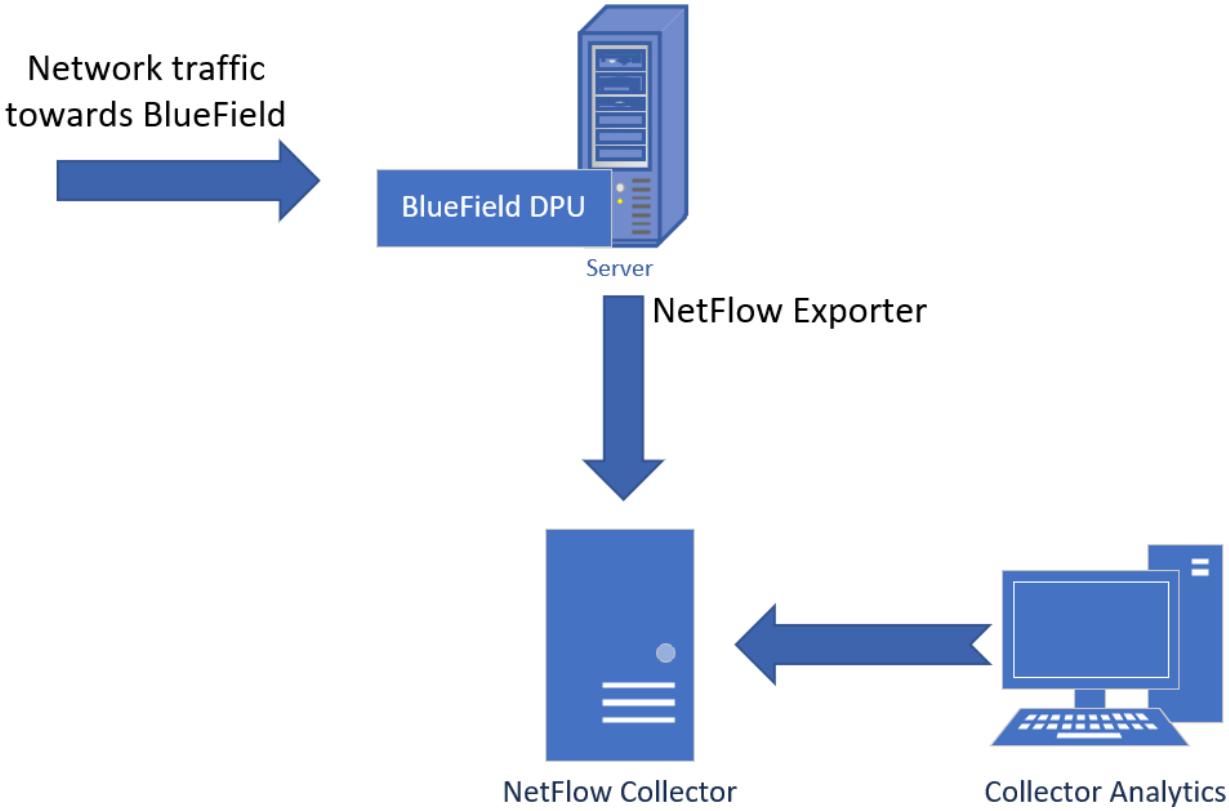
  NetFlow data allows network administrators to view a complete report on traffic which allows them to understand bandwidth consumption, network resource utilization, etc.
- Anomaly-based attack detection

  Flow-based analysis relies on algorithms and behavior rather than signature matching. This gives the flow analyzer the ability to detect attacks before a signature is available.
- Network visibility

  NetFlow provides the ability to drill down into network traffic to see source and destination address and ports, protocol, and more. With this information, it is possible to identify traffic patterns throughout the entire network. Full view of traffic flow allows network operation and security operation teams to monitor when and how often users access applications in the network.

Chapter 2. System Design

The following diagram illustrates NetFlow architecture and nodes:



Chapter 3. Application Architecture

1. Define record struct and template.

   The application must define a TLV NetFlow v9 template. This is done by using `doca_netflow_types.h` defined macros. The order has no significance for the collector but must be maintained for the record struct. The struct should be in packed format without pointers. The app can use the default record struct and template by using structs `doca_netflow_default_record` and `doca_netflow_template_default_get()`.

   The app can use the default record struct and template by using structs `doca_netflow_default_record` and `doca_netflow_template_default_get()`.

2. Initiate NetFlow Exporter.

   Call `doca_netflow_exporter_init()` to initiate the exporter. Pass the configuration file or pass NULL for the default path.

   The configuration file is located by default at `/etc/doca-netflow.conf`. The configutation file should contain:

   - Title – [doca_netflow_conf]
   - Target – the IP and port of the collector (to send the NetFlow records)
   - Source ID – the source ID of the NetFlow exporter (defined by the user) to send alongside the NetFlow records to identify the source of the record
   - Version – 9 (for future development)

   For example:

   ```
   [doca_netflow_conf]
   target = 127.0.0.1:2055
   source_id = 10
   version = 9
   ```

3. Update record information.

   The app should update the flow information with information that extracted from the packets. For information on this process, please refer to the NVIDIA DOCA Application Recognition Reference Guide.

This part should be periodically updated before sending the records.

4. Send records.

   Call `doca_netflow_exporter_send()` to send the records to the collector.

   This function should be periodically called, and must contain all the current active flows or flow-sampled records in that period of time.

5. Destroy.

   Call `doca_netflow_exporter_destroy()` to close the exporter before exiting the application.

The NetFlow example source file shows how to set up the exporter and send an array of record flows to a collector. It contains 2 functions:

- `default_template_example()` – this example uses the default template with the default struct
- `custom_template_example()` – this example creates a custom template that can be altered to meet the application's needs

  For simplicity and generalization, the example sends a static dummy record duplicated over an array of records. This is the part the app that includes the NetFlow lib should supply.

Chapter 4. Configuration Flow

1. Create a new record with a default template.

   Note: It is possible to configure a custom record.

```
struct doca_netflow_default_record record = {
        .src_addr_v4,/* Source IP Address */
        .dst_addr_v4,/* Destination IP Address */
        .next_hop,/* Next hop router's IP Address */
        .input,/* Input interface index */
        .output,/* Output interface index */
        .src_port,/* TCP/UDP source port number or equivalent */
        .dst_port,/* TCP/UDP destination port number or equivalent */
        .tcp_flags,/* Cumulative OR of tcp flags */
        .protocol,/* IP protocol type (for example, TCP = 6, UDP = 17) */
        .tos,/* IP Type-of-Service */
        .src_as,/* originating AS of source address */
        .dst_as,/* originating AS of destination address */
        .src_mask,/* source address prefix mask bits */
        .dst_mask,/* destination address prefix mask bits */
        .d_pkts,/* Packets sent in Duration */
        .d_octets,/* Octets sent in Duration. */
        .first,/* SysUptime at start of flow */
        .last,/* and of last packet of flow */
        .flow_id,/* This identifies a transaction within a connection */
        .application_name /* Name associated with a classification */
        };
```

2. Initialize NetFlow exporter. Run:

```
doca_netflow_exporter_init(file_path);
```

3. Send NetFlow records. Run:

```
doca_netflow_exporter_send(&template, (const void **)(records), records_length,
 &err)
```

Chapter 5. Running Application on BlueField

1. Please refer to the DOCA Installation Guide for details on how to install BlueField related software.
2. To build the application:
   a). Modify the code. Change the "netflow.c" example, the configuration file location, the record to send, the template, etc.
   b). Prepare the configuration file which may be found by default at `/etc/doca_netflow.conf`.
   c). Compile the example. Run:
   ```
   cd /opt/mellanox/doca/examples/netflow/src
   meson /tmp/build
   ninja -C /tmp/build
   ```
3. To run the application, simply run `doca_netflow`.

Chapter 6. References

- `/opt/mellanox/doca/examples/netflow/src/netflow.c`

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. Other company and product names may be trademarks of the respective companies with which they are associated.

Copyright

© 2021 NVIDIA Corporation & affiliates. All rights reserved.

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