DPL Debugger

Table of contents

Introduction

Executing the DPL Debugger

Using the DPL Debugger

- Open Debug Session File
- Packets List
- Filter
- Parser Graph
- Debug Points
- Variables
- Source Code
- Log
- System Info
- Packet Dissector
- Hex Dump
- Options

This page describes the DPL Debugger tools, which are used to debug pipeline packets processed by DPL programs on the NVIDIA® BlueField® DPU.

Introduction

The DPL Debugger is a GUI application that can run a live debug session or open a pre-recorded debug session file captured using the `dpl_nspect debug` command.

A debug session visualizes the flow of network packets through a DPL program's pipeline, allowing developers to trace each packet's path in detail. For more information, refer to the documentation on the nv_send_debug_pkt extern function.

A debug session visualizes the flow of network packets through a DPL program’s pipeline, allowing developers to trace packet details throughout its course. For more information, refer to the documentation on the nv_send_debug_pkt extern function.

> **Note**
>
> The current version of the debugger only displays debug packets received by the DPU (Rx path) or packets arriving from the primary wire port (P0) in the network-to-host direction.

Executing the DPL Debugger

Running `dpl_debugger.sh` without any arguments launches the GUI application.

To open and debug a pre-recorded session specify the session archive when executing the program:

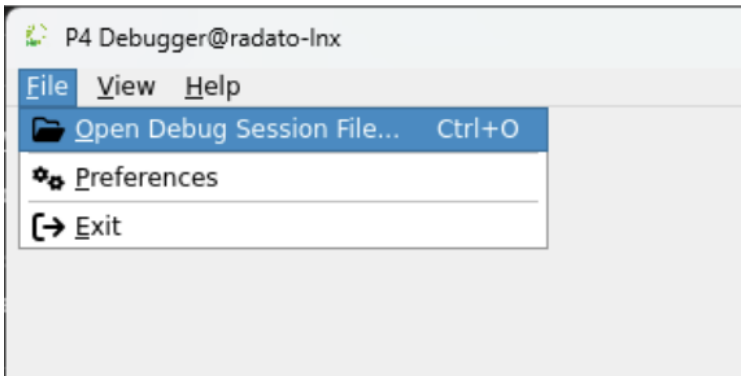
```
dpl_debugger.sh my-debug-archive.tar.gz
```

Using the DPL Debugger

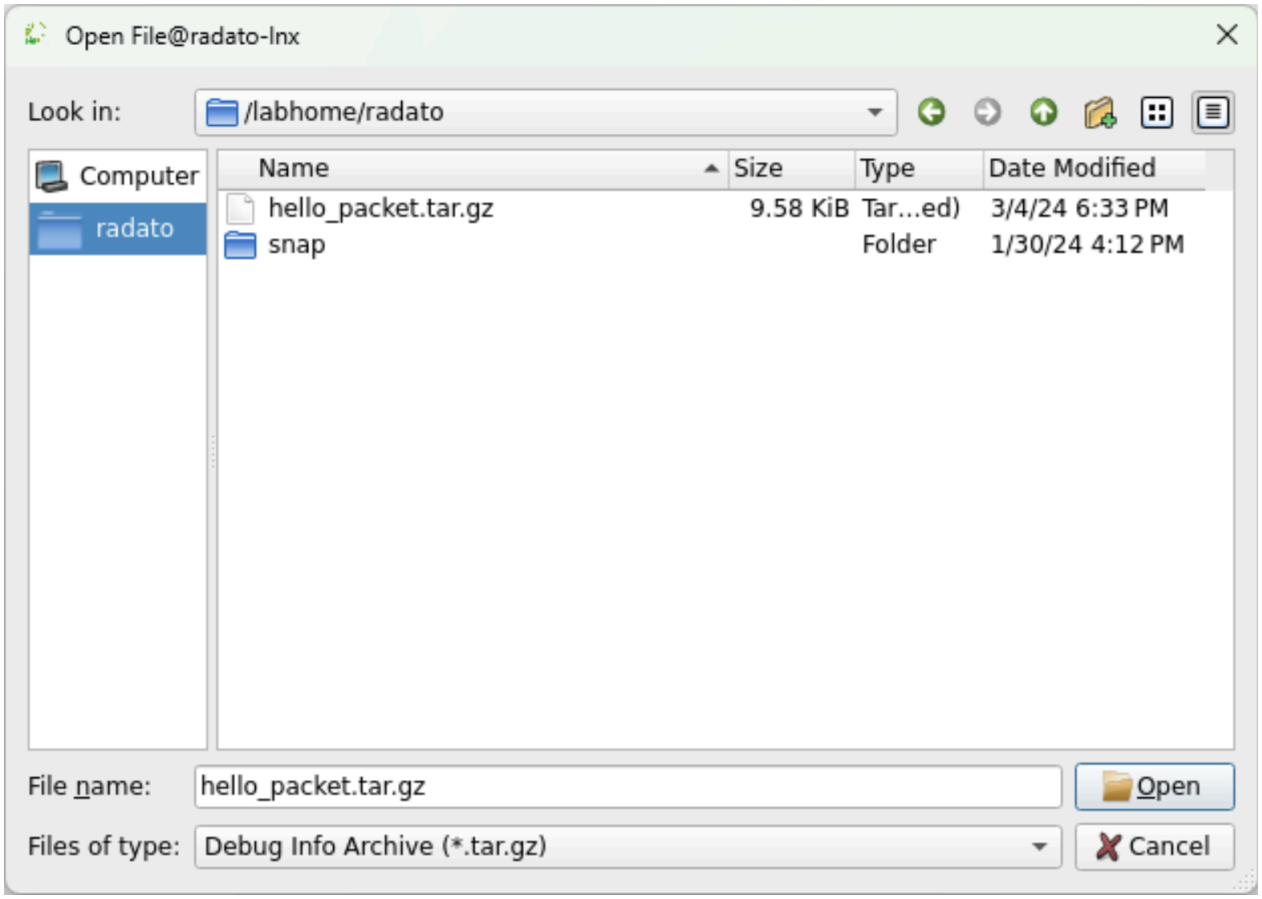
Open Debug Session File

To open a pre-recorded debug session in the DPL Debugger GUI:

1. Select the **File** → **Open Debug Session File**.



2. Choose the appropriate debug session file (`hello_packet.tar.gz` in this example).



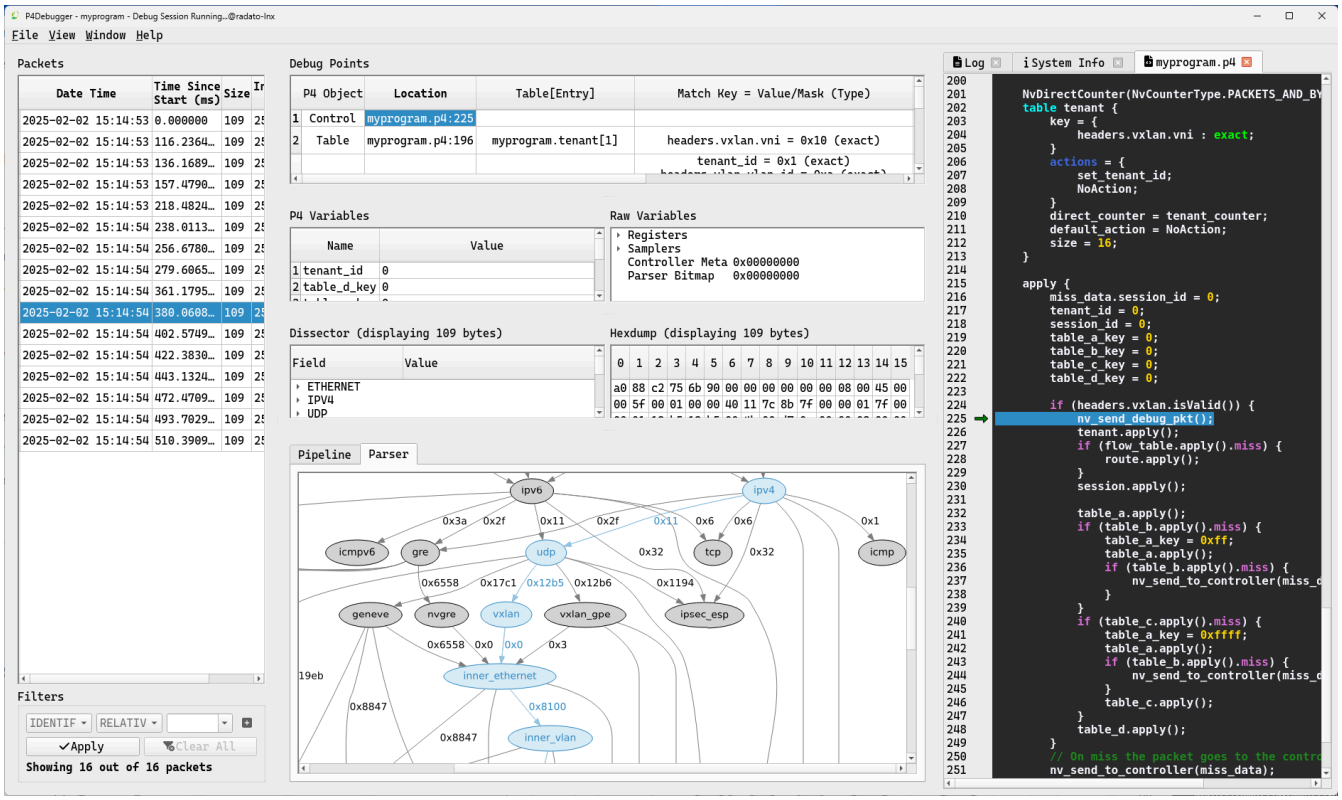
> **Info**

The `tar.gz` archive should be the output of a recorded debug session created using the `dpl_nspect debug` command.

Packets List

The left pane of the DPL Debugger displays the **Packets** list, which includes the following fields:

- Date Time – The timestamp when the packet entered the DPL pipeline
- Time Since Start – The elapsed time since the first packet entered the DPL pipeline
- Size – The packet size
- Ingress Port – The port through which the packet entered the pipeline
- Protocol – The network protocol associated with the packet

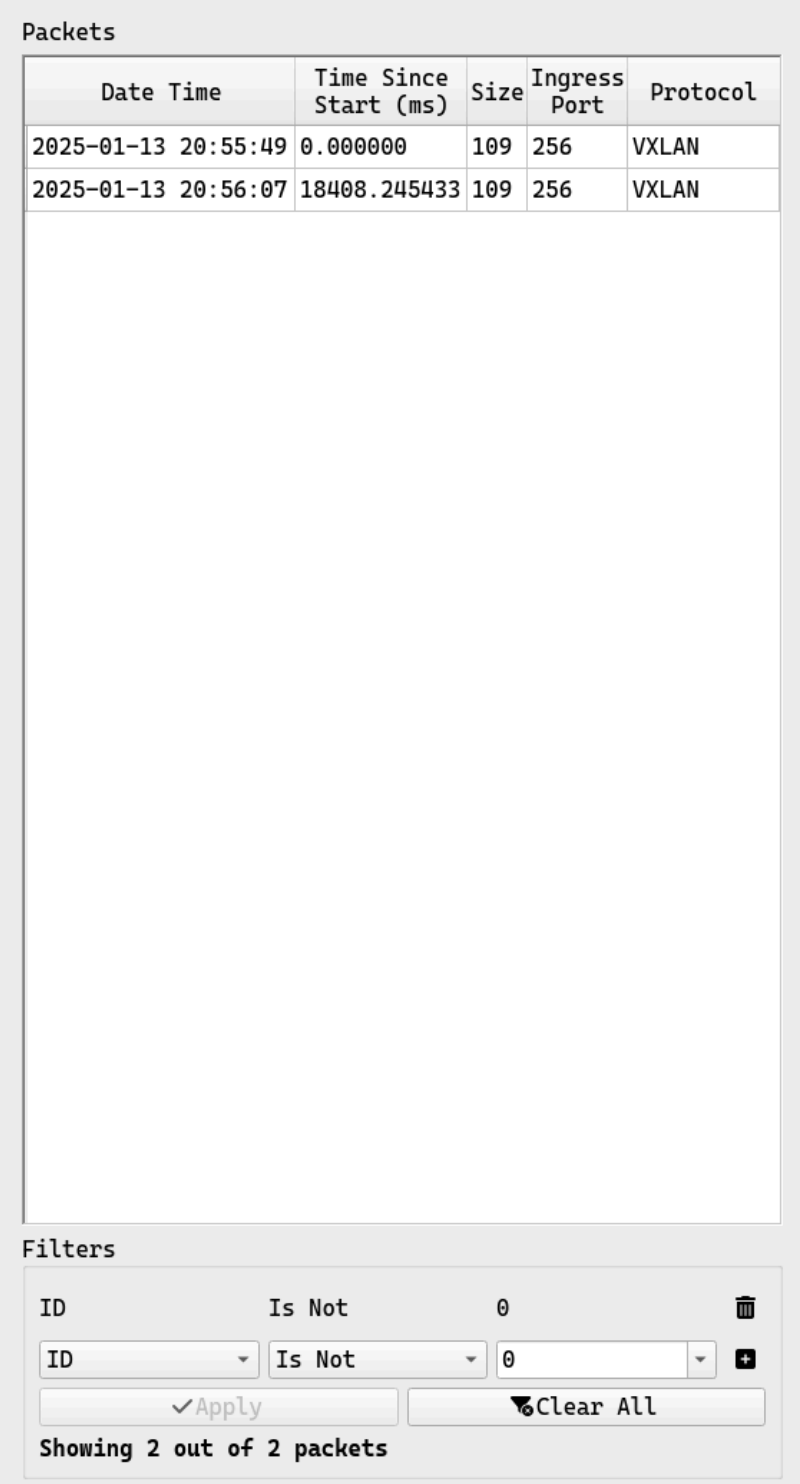


Filter

The **Packets** list can be filtered using the fields located at the bottom left of the GUI.

To apply a filter:

1. Enter a condition (e.g., "packet ID is not 0").
2. Click the **+** button to add the filter.
3. Click the **Apply** button to apply the filter.



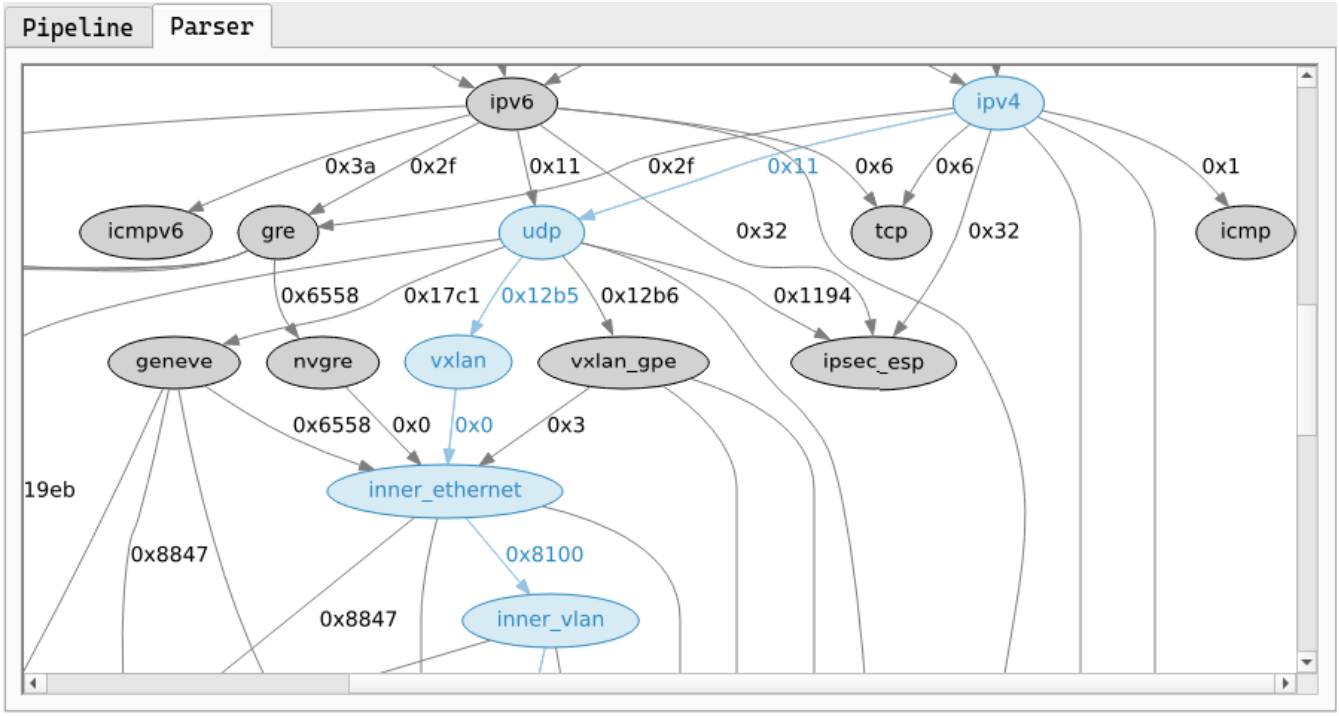
To reset the **Packets** list to its original state:

1. Click the **Clear All** button.
2. Click **Apply**.

Parser Graph

The **Parser** graph, located at the bottom of the GUI, outlines the possible routes a packet can take within the DPL pipeline.

- Blue nodes (`inner_ethernet` in the example) indicate the nodes the packet has traversed.
- The graph consists of two types of nodes:
    - Flex nodes – Defined by the DPL programmer
    - Static (fixed) nodes – Built-in components of the hardware



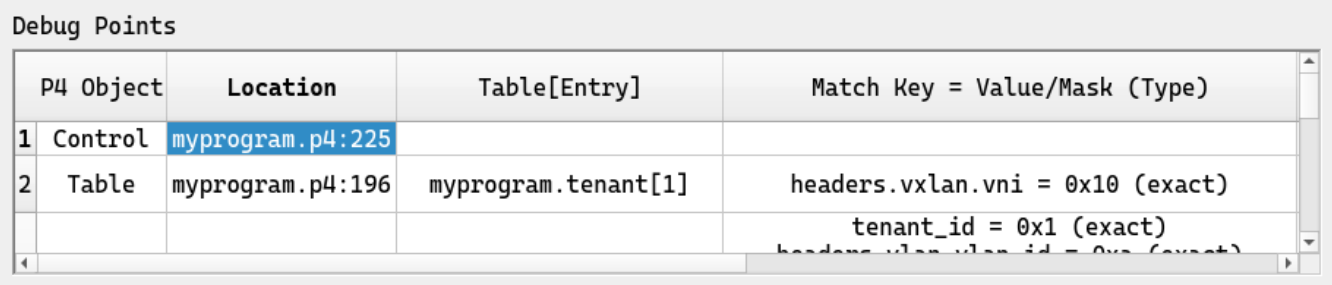
This visualization helps developers understand the packet's path through the parsing stages.

Debug Points

Selecting a packet from the **Packets** List displays its pipeline stages in the Debug Points view at the top of the GUI:

- Each row corresponds to a specific source location in the DPL program code

- The table cells are interactive—clicking a cell highlights the relevant source location, when applicable



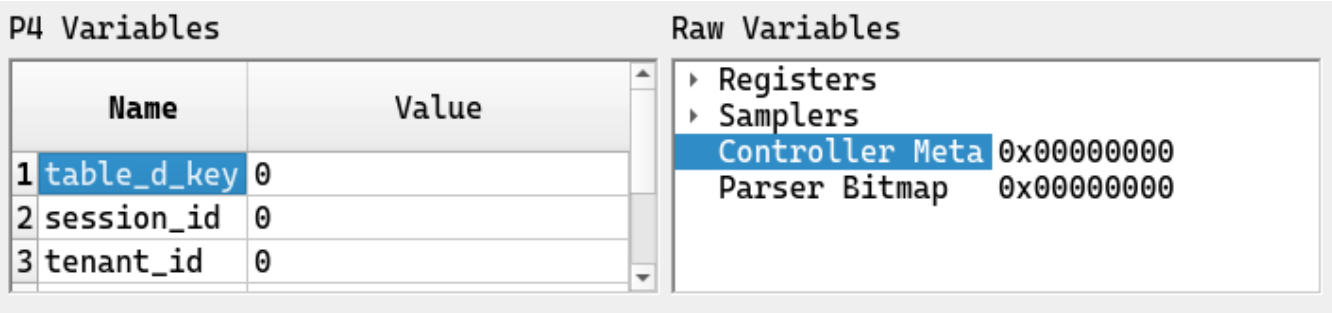
This allows developers to trace a packet's processing path and correlate it with the corresponding code sections.

Variables

Selecting a pipeline stage (either a single cell or an entire row) displays variable information in two tabs:

- P4 Variables – Shows DPL variables defined in the program and their values at the selected pipeline stage.

- Raw Variables – Provides a low-level view of internal variables, such as registers, samplers, and other hardware-related data.



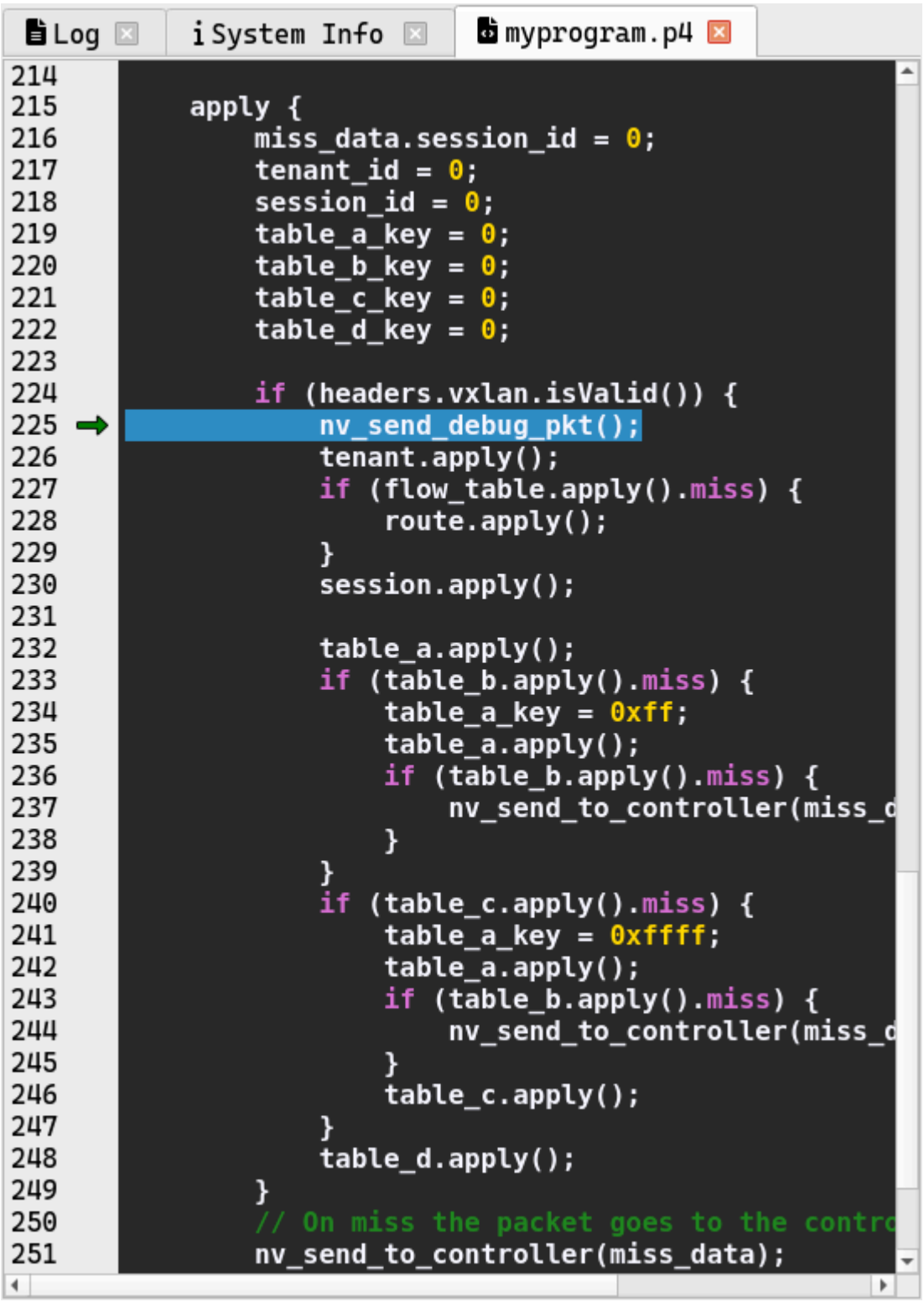
This helps developers inspect both high-level program variables and hardware-level details while debugging.

Source Code

Selecting a cell in a pipeline stage opens the DPL program source code in a new tab on the right-hand side of the DPL Debugger GUI. When applicable, the relevant source location is highlighted.

The debugger locates the DPL source file using:

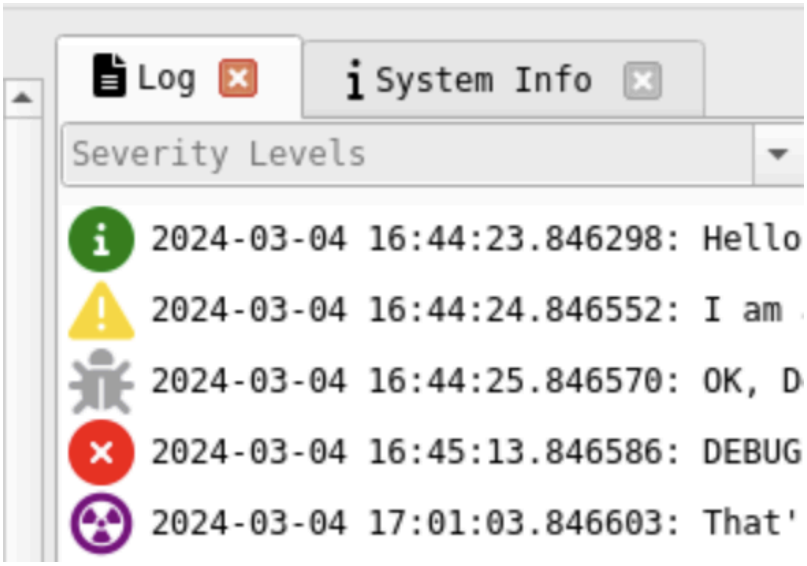
- Search paths defined in the **Settings**
- Manual file selection, if the source file is not found automatically


Log
System Info
myprogram.p4
214
215 apply {
216 miss_data.session_id = 0;
217 tenant_id = 0;
218 session_id = 0;
219 table_a_key = 0;
220 table_b_key = 0;
221 table_c_key = 0;
222 table_d_key = 0;
223
224 if (headers.vxlan.isValid()) {
225 nv_send_debug_pkt();
226 tenant.apply();
227 if (flow_table.apply().miss) {
228 route.apply();
229 }
230 session.apply();
231
232 table_a.apply();
233 if (table_b.apply().miss) {
234 table_a_key = 0xff;
235 table_a.apply();
236 if (table_b.apply().miss) {
237 nv_send_to_controller(miss_d
238 }
239 }
240 if (table_c.apply().miss) {
241 table_a_key = 0xffff;
242 table_a.apply();
243 if (table_b.apply().miss) {
244 nv_send_to_controller(miss_d
245 }
246 table_c.apply();
247 }
248 table_d.apply();
249 }
250 // On miss the packet goes to the contro
251 nv_send_to_controller(miss_data);

Log

The **Log** tab, located on the right side of the GUI, displays internal logs from the DPL Runtime daemon.

> **Note**
>
> Currently, no log messages are being sent.



System Info

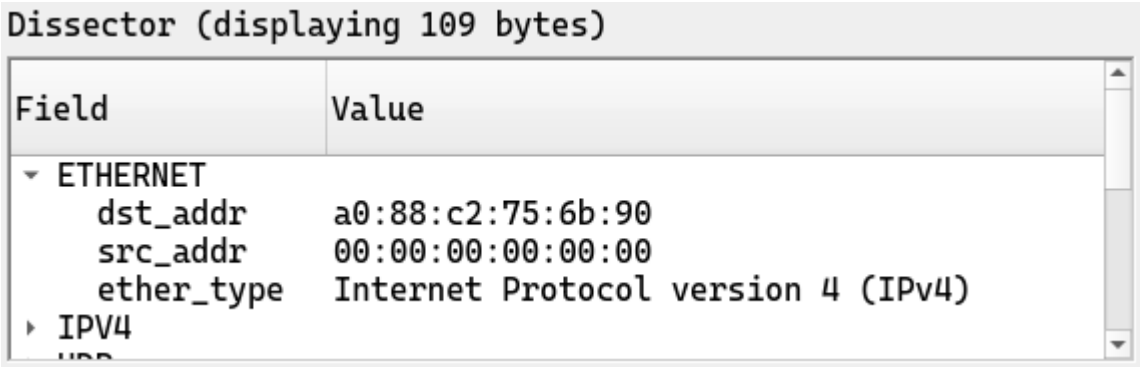
The **System Info** tab, located on the right side of the GUI, displays details about the system where debugging is performed.

Log | System Info | gtp_tunnel.p4

Key	Value
Linux OS version	"Ubuntu 22.04.5 LTS"
DOCA version	2.10.0087-1
OFED Version	OFED-internal-25.01-0.6.0:…
RDMA Core Version	2501mlnx56-1.2501060
DPL Runtime daemon version	1.0.0-1.rel-1.0.0-64
Debug Schema Version	Device ID: 1000, doctype: /nvp4c/devtools/base/v6, …

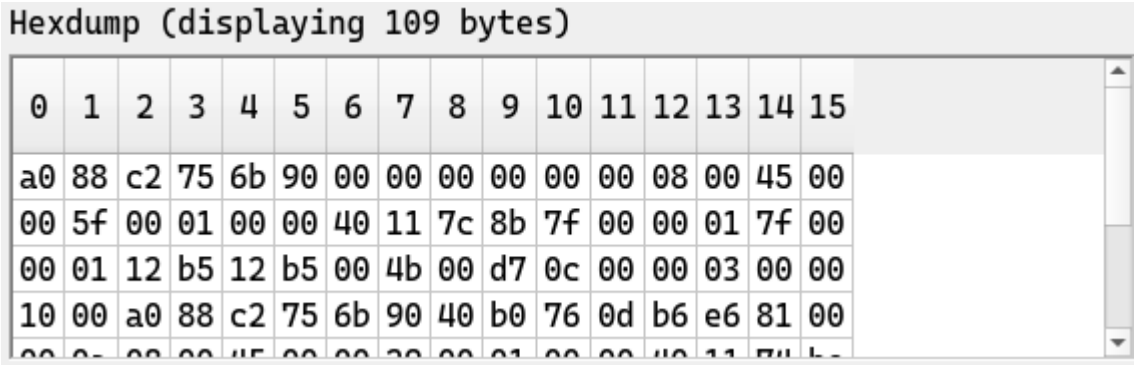
Packet Dissector

The **Dissector** view, located in the center of the GUI, analyzes a packet's structure when a pipeline stage is selected.



Hex Dump

The **Hex Dump** view, located in the center of the GUI, displays a raw hexadecimal representation of the packet when a pipeline stage is selected.



Options

The **File → Preferences** menu opens the **Preferences** window, where users can configure various debugger and debug session settings.