FRAMOS Industrial Depth Camera D400e Series - Tuning System For Best Performance

APPLICATION NOTE

Version 1.2.0 from 2021-10-15

www.framos.com

Table of Contents

List of Figures

List of Tables

List of Abbreviations

Abbreviation	Explanation
GEV	GigE Vision
NIC	Network Interface Card
PCI	Peripheral Component Interconnect
PCI-E	PCI Express
SDK	Software Development Kit
UDP	User Datagram Protocol

1. Overview

The *FRAMOS Industrial Depth Camera D400e Series - Tuning System For Best Performance* application note provides tips and recommendations on setting system and camera settings to achieve best performance from FRAMOS D400e camera series.

2. System Settings

2.1 General Recommendations

There are number of system related parameters and conditions that could negatively affect the system performance and consequently limit the maximum achievable performance of the FRAMOS Industrial Depth camera D400e series used in such system. Perhaps the most important factor to consider is selection and configuration of the Network Interface Card (NIC). It is recommended to use a Gigabit Ethernet NIC with PCI-E interface supporting Jumbo Frames feature.

Information: For best performance FRAMOS recommends Intel® PCI-E network interface cards.

2.2 Network Related Settings

The following network related settings should be adjusted to achieve optimal performance with FRAMOS Industrial RealSense D400e camera series:

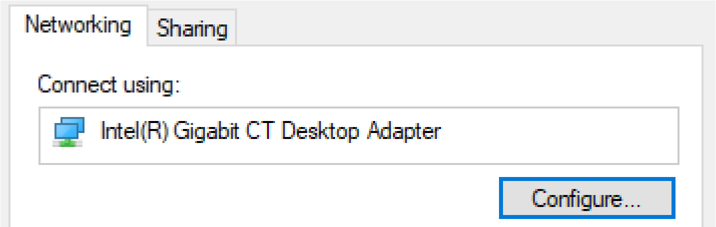
- Jumbo Frames - Using Jumbo Frames reduces the packet overhead and workload on the target PC. This is achieved by transmitting smaller number of large packets instead of large number of small packets.
- Receive Buffers - The Receive Buffers feature of the NIC represents a reserved memory in the system memory, which is used to buffer incoming data. It is recommended to increase buffer size for the incoming packets (Receive Buffers) to the maximum supported value in order to decrease probability of packet loss in conditions of high system load.

The availability of the features and their limits can differ between NICs vendors and operating systems. The descriptions of driver settings in following chapters are based on Intel® NICs. The settings are shown for Microsoft Windows and Linux operating systems.

2.2.1 Network Interface Optimization for Windows Operating System

Driver settings are configured through the driver properties dialog. On the Microsoft Windows operating systems this can be accessed the following way:

- Open *Control Panel* ->*Network Connections* window (can be invoked directly by executing command "ncpa.cpl" from Windows Run application or from command prompt).
- Right-click the target network interface card and choose Properties.
- Press the Configure button bellow the name of the physical network interface, as shown in Figure 1.

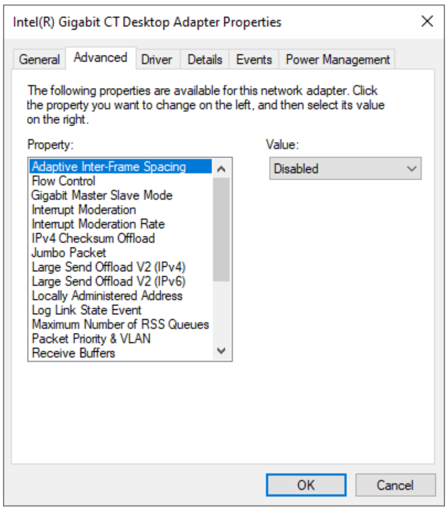


Connect using:

Intel(R) Gigabit CT Desktop Adapter

Figure 1 – Access to NIC driver settings

- The settings of the driver can be accessed by choosing the *Advanced Tab* of the pop-up window.



Intel(R) Gigabit CT Desktop Adapter Properties

The following properties are available for this network adapter. Click the property you want to change on the left, and then select its value on the right.

Property:

Adaptive Inter-Frame Spacing
Flow Control
Gigabit Master Slave Mode
Interrupt Moderation
Interrupt Moderation Rate
IPv4 Checksum Offload
Jumbo Packet
Large Send Offload V2 (IPv4)
Large Send Offload V2 (IPv6)
Locally Administered Address
Log Link State Event
Maximum Number of RSS Queues
Packet Priority & VLAN
Receive Buffers

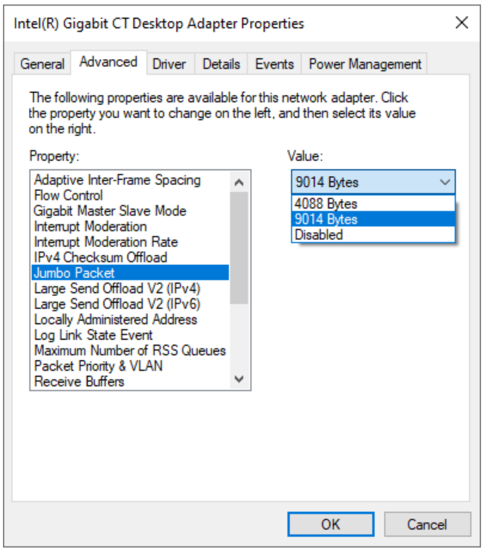
Value:

Disabled

Figure 2 – NIC - advanced driver settings

2.2.1.1 Jumbo Frames/Packets - Network Interface Cards

The Jumbo Frames (or Jumbo Packet) feature is disabled by default on most NICs, so it needs to be manually enabled within the device driver property window as shown on Figure 3. To enable Jumbo Frames, set the property to its maximum value.



The following properties are available for this network adapter. Click the property you want to change on the left, and then select its value on the right.

Property:
- Adaptive Inter-Frame Spacing
- Flow Control
- Gigabit Master Slave Mode
- Interrupt Moderation
- Interrupt Moderation Rate
- IPv4 Checksum Offload
- Jumbo Packet
- Large Send Offload V2 (IPv4)
- Large Send Offload V2 (IPv6)
- Locally Administered Address
- Log Link State Event
- Maximum Number of RSS Queues
- Packet Priority & VLAN
- Receive Buffers

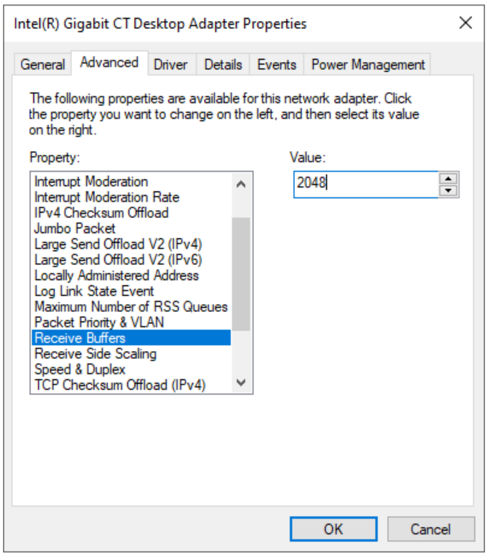
Value: 9014 Bytes
- 4088 Bytes
- 9014 Bytes
- Disabled

Figure 3 – NIC - Enabling Jumbo Packet on Windows

Information: In network configurations where camera is not connected directly to NIC but via other network components (i.e., gigabit switches), it is required to enable Jumbo Frames on all the components used in the network path.

2.2.1.2 Receive Buffers - Network Interface Cards

Receive Buffers property can be accessed via driver property window as shown on Figure 4. It is recommended to raise the value of the Receive Buffers property to its maximum value.



The following properties are available for this network adapter. Click the property you want to change on the left, and then select its value on the right.

Property:
- Interrupt Moderation
- Interrupt Moderation Rate
- IPv4 Checksum Offload
- Jumbo Packet
- Large Send Offload V2 (IPv4)
- Large Send Offload V2 (IPv6)
- Locally Administered Address
- Log Link State Event
- Maximum Number of RSS Queues
- Packet Priority & VLAN
- Receive Buffers
- Receive Side Scaling
- Speed & Duplex
- TCP Checksum Offload (IPv4)

Value: 2048

2.2.2 Network Interface Optimization for Linux Operating System

2.2.2.1 Jumbo Frames - Network Interface Cards

On the Linux operating system Jumbo Frames can be enabled temporarily or it can be done at boot time by modifying appropriate configuration files.

Enabling Jumbo Frames temporarily can be done using "ip" tool:

```
sudo ip link set <ethx> mtu <size>
```

where `<ethx>` represents the name of network interface that connects PC to FRAMOS D400e camera series, and `<size>` the desired frame/packet size in bytes (i.e., 9000). Alternative approach could be using the "ifconfig" command:

```
sudo ifconfig <ethx> mtu <size>
```

This setting will last until the system is rebooted.

ℹ **Information:** "Ip" tool may not be available on all platforms. Refer to the documentation of used Linux distribution for more details.

ℹ **Information:** Refer to documentation of used Linux distribution for information on how to enable Jumbo Frames permanently.

2.2.2.2 Receive Buffers - Network Interface Cards

To increase Receive Buffers size on the Linux operating system "ethtool" provided by many Linux distributions can be used to configure the property, as per the following command:

```
sudo ethtool -G <ethx> rx <size>
```

where `<ethx>` represents the name of network interface that is being configured, and `<size>` the desired number of Receive Buffers (i.e., 2048). If "ethtool" tool is not provided by used distribution, please refer to distribution documentation for alternatives.

2.3 Application Environment Considerations

For best performance it is recommended to run applications using FRAMOS Industrial Depth camera D400e series on a dedicated system with freshly installed operating system. Having lots of concurrent processes and background services might have negative effect on software application performance. Additionally, it was observed that some applications have negative impact on network subsystem performance and overall system latency resulting in excessive packet loss and unstable streaming performance. Therefore, it is recommended to work on a freshly installed system to reduce the possibility of unwanted performance degradation.

In addition, anti-virus software, firewalls, third-party filter drivers and network traffic monitoring software can affect the performance of the system as they monitor the incoming packets at different network stack layers. It is recommended to disable or uninstall such software if performance degradation is observed.

3. Camera Settings

3.1 Inter Packet Delay

The Inter Packet Delay (GevSCPD GenICam feature) is the delay introduced by the camera between sending two consecutive packets on the stream channel. This delay reduces the effective network load (on NIC or switch) and creates timeslots for processing packets from other devices on the network. The Inter Packet Delay is very useful when multiple cameras are streaming to one PC on the same Network Interface Card (NIC).

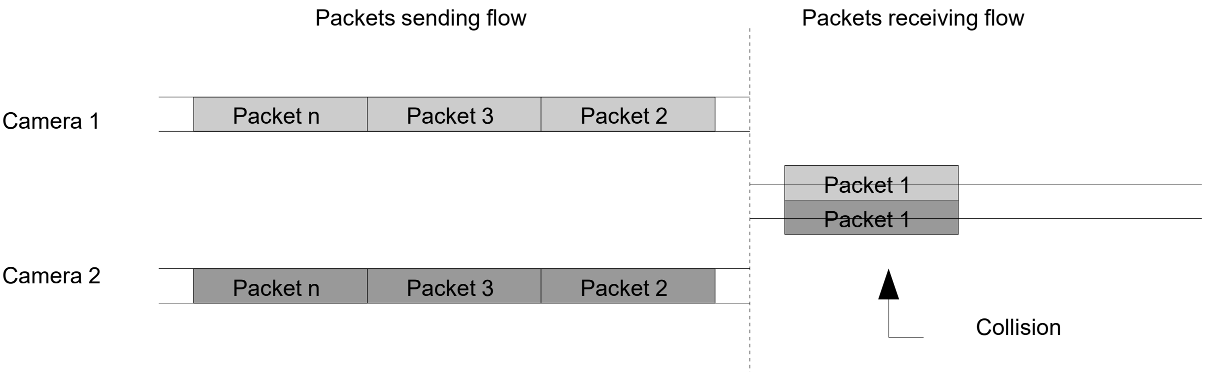


Figure 5 – Packet flow when inter packet delay is not used

If Inter Packet Delay feature is not used, excessive packet collisions may occur which can result in data loss, like illustrated in Figure 5. Packets from two cameras are sent to PC over the same network connection. Without any Inter Packet Delay set, collision between packets from different cameras may occur in case both cameras stream packets at the same time. Figure 6 illustrates a well configured Inter Packet Delay to prevent collisions.

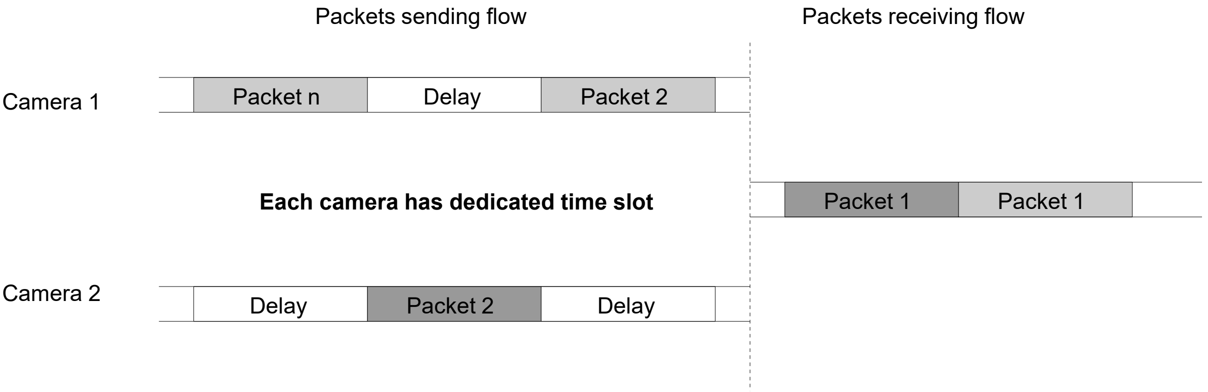


Figure 6 – Packet flow with properly configured inter packet delay

Assuming that the Packet Size (GevSCPSPacketSize GenICam feature) is 1500 bytes, effective Ethernet packet size including inter-frame gap, preamble, header and CRC on the wire is 1538 bytes. It takes 8ns to transfer one byte over Gigabit Ethernet network, so time required to transfer one packet of 1538 bytes is 12,3us. In other words, camera is able to send maximum of 81274 packets via the Ethernet interface every second. In case two cameras are streaming to one NIC, the Inter Packet Delay should be a bit larger than the time required to transfer one packet, in order to ensure that packets from second camera will fit in the vacant time slot.

Increasing Inter Packet Delay extends frame transfer time which means higher latency but can also result in lower framerate. For example, if one camera is producing 30000 packets per second (50 frames per second, 600 packets per frame), total packet transfer time (packet + inter packet delay) must not exceed 33,3us to preserve frame rate.

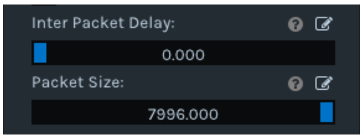
In the more complex system, including multiple cameras with multiple streams, determining Inter Packet Delay is not so intuitive anymore. It is advised to iteratively increase Inter Packet Delay until stable stream is achieved. On the other side, Inter Packet Delay must not be set too high due the increased latency and possible lower framerate. With the empirical approach, Inter Packet Delay should be determined quickly.

3.1.1 Example for Adjusting Inter Packet Delay

One camera is configured with following stream profile: 1280x720, @30FPS for both Stereo Module and RGB Camera. Wireshark is used for network protocol analysis. Results may variate with different NICs and its configurations.

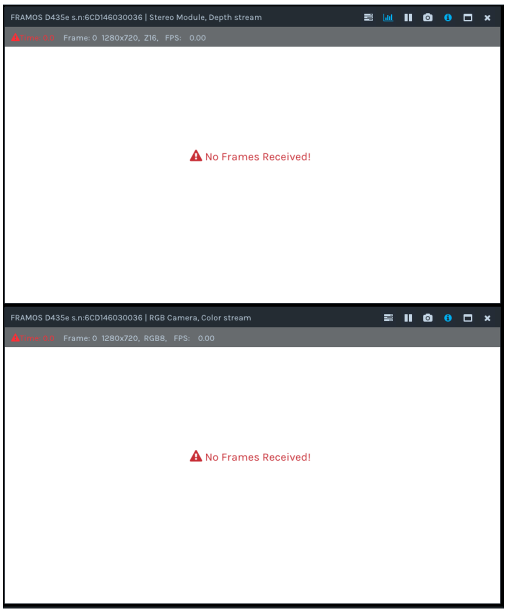
Information: Wireshark is a widely used network protocol analyzer. It is used for network analysis and troubleshooting. When adjusting Inter Packet Delay, it is advised to use Wireshark for analysis.

Case 1:
Stereo Module Inter Packet Delay: 0.
Stereo Module Packet Size: 7996.
RGB Camera Inter Packet Delay: 0.
RGB Camera Packet Size: 7996.



Inter Packet Delay: 0.000
Packet Size: 7996.000

As consequence of inadequate Inter Packet Delay there are no streams as shown below.



FRAMOS D435e s.n:6CD146030036 | Stereo Module, Depth stream

Time: 0.0 Frame: 0 1280x720, Z16, FPS: 0.00

No Frames Received!

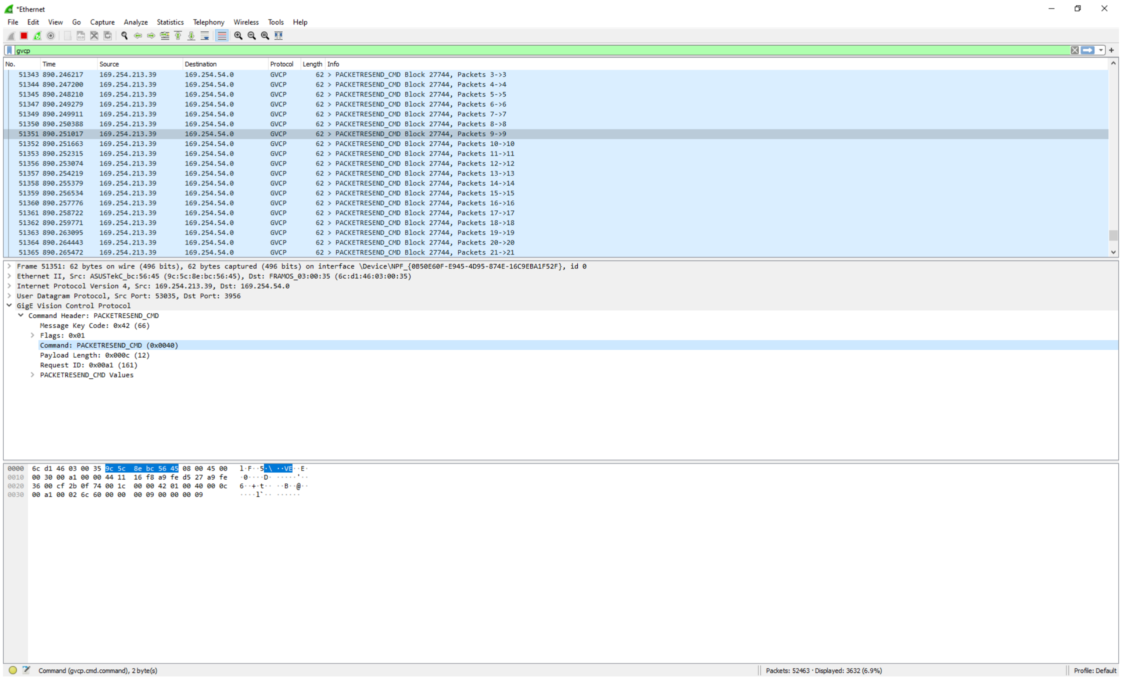
FRAMOS D435e s.n:6CD146030036 | RGB Camera, Color stream

Time: 0.0 Frame: 0 1280x720, RGB8, FPS: 0.00

No Frames Received!

Figure 7 – Inter Packet Delay set to 0 - no frames received

In addition, there is an excessive number of Packet Resend requests issued from PC to camera as shown on Figure 8. Packet Resend is the mechanism used to recover the missing packets in case of packet loss.



*Ethernet
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
gvcp

```
No.   Time       Source          Destination   Protocol Length Info
51343 890.246217 169.254.213.39  169.254.54.0  GVCP     62 > PACKETRESEND_CMD Block 27744, Packets 3->3
51344 890.247200 169.254.213.39  169.254.54.0  GVCP     62 > PACKETRESEND_CMD Block 27744, Packets 4->4
51345 890.248210 169.254.213.39  169.254.54.0  GVCP     62 > PACKETRESEND_CMD Block 27744, Packets 5->5
51347 890.249279 169.254.213.39  169.254.54.0  GVCP     62 > PACKETRESEND_CMD Block 27744, Packets 6->6
51349 890.249911 169.254.213.39  169.254.54.0  GVCP     62 > PACKETRESEND_CMD Block 27744, Packets 7->7
51350 890.250388 169.254.213.39  169.254.54.0  GVCP     62 > PACKETRESEND_CMD Block 27744, Packets 8->8
51351 890.251017 169.254.213.39  169.254.54.0  GVCP     62 > PACKETRESEND_CMD Block 27744, Packets 9->9
51352 890.251663 169.254.213.39  169.254.54.0  GVCP     62 > PACKETRESEND_CMD Block 27744, Packets 10->10
51353 890.252315 169.254.213.39  169.254.54.0  GVCP     62 > PACKETRESEND_CMD Block 27744, Packets 11->11
51356 890.253074 169.254.213.39  169.254.54.0  GVCP     62 > PACKETRESEND_CMD Block 27744, Packets 12->12
51357 890.254219 169.254.213.39  169.254.54.0  GVCP     62 > PACKETRESEND_CMD Block 27744, Packets 13->13
51358 890.255379 169.254.213.39  169.254.54.0  GVCP     62 > PACKETRESEND_CMD Block 27744, Packets 14->14
51359 890.256534 169.254.213.39  169.254.54.0  GVCP     62 > PACKETRESEND_CMD Block 27744, Packets 15->15
51360 890.257776 169.254.213.39  169.254.54.0  GVCP     62 > PACKETRESEND_CMD Block 27744, Packets 16->16
51361 890.258722 169.254.213.39  169.254.54.0  GVCP     62 > PACKETRESEND_CMD Block 27744, Packets 17->17
51362 890.259771 169.254.213.39  169.254.54.0  GVCP     62 > PACKETRESEND_CMD Block 27744, Packets 18->18
51363 890.263095 169.254.213.39  169.254.54.0  GVCP     62 > PACKETRESEND_CMD Block 27744, Packets 19->19
51364 890.264443 169.254.213.39  169.254.54.0  GVCP     62 > PACKETRESEND_CMD Block 27744, Packets 20->20
51365 890.265472 169.254.213.39  169.254.54.0  GVCP     62 > PACKETRESEND_CMD Block 27744, Packets 21->21
```

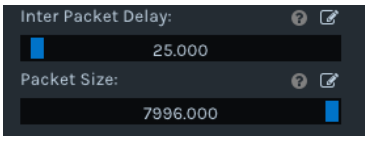
```
> Frame 51351: 62 bytes on wire (496 bits), 62 bytes captured (496 bits) on interface \Device\NPF_{0B50E60F-E945-4D95-874E-16C9EBA1F52F}, id 0
> Ethernet II, Src: ASUSTekC_bc:56:45 (9c:5c:8e:bc:56:45), Dst: FRAMOS_03:00:35 (6c:d1:46:03:00:35)
> Internet Protocol Version 4, Src: 169.254.213.39, Dst: 169.254.54.0
> User Datagram Protocol, Src Port: 53035, Dst Port: 3956
v GigE Vision Control Protocol
  v Command Header: PACKETRESEND_CMD
      Message Key Code: 0x42 (66)
    > Flags: 0x01
      Command: PACKETRESEND_CMD (0x0040)
      Payload Length: 0x000c (12)
      Request ID: 0x00a1 (161)
    > PACKETRESEND_CMD Values
```

```
0000  6c d1 46 03 00 35 9c 5c  8e bc 56 45 08 00 45 00   l·F··5·\ ··VE··E·
0010  00 30 00 a1 00 00 44 11  16 f8 a9 fe d5 27 a9 fe   ·0····D· ·····'··
0020  36 00 cf 2b 0f 74 00 1c  00 00 42 01 00 40 00 0c   6··+·t·· ··B··@··
0030  00 a1 00 02 6c 60 00 00  00 09 00 00 00 09         ····l`·· ······
```

Command (gvcp.cmd.command), 2 byte(s) | Packets: 52463 · Displayed: 3632 (6.9%) | Profile: Default

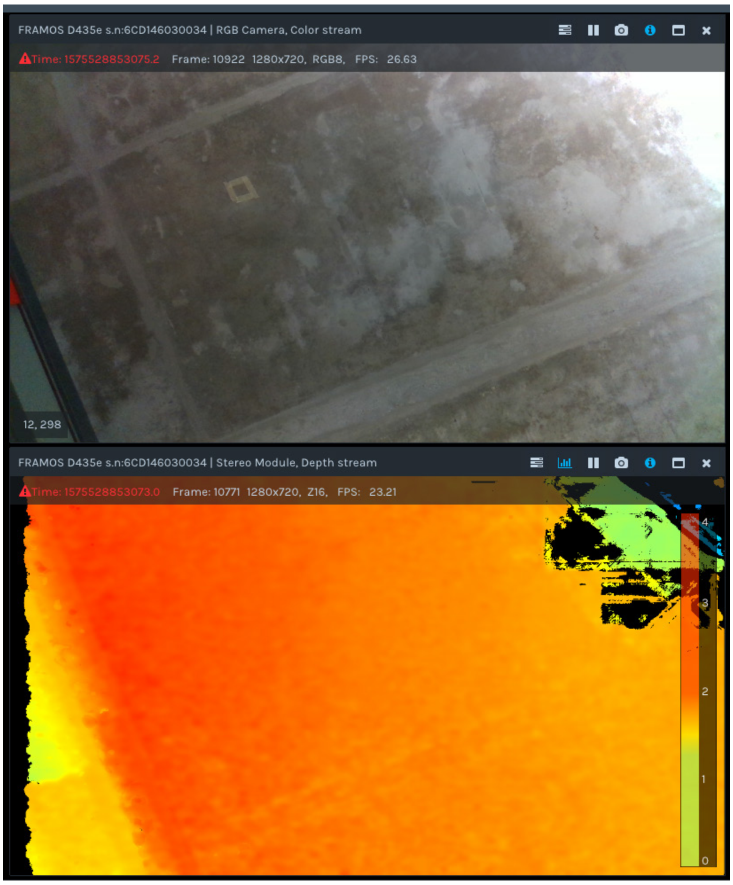
Figure 8 – Wireshark log for Packet Resend requests, Case 1

Case 2:
Stereo Module Inter Packet Delay: 25.
Stereo Module Packet Size: 7996.
RGB Camera Inter Packet Delay: 25.
RGB Camera Packet Size: 7996.



Inter Packet Delay: 25.000
Packet Size: 7996.000

With increased Inter Packet Delay value for both Stereo and RGB modules, streams are active, but framerate drop is noticeable due the insufficient Inter Packet Delay value.

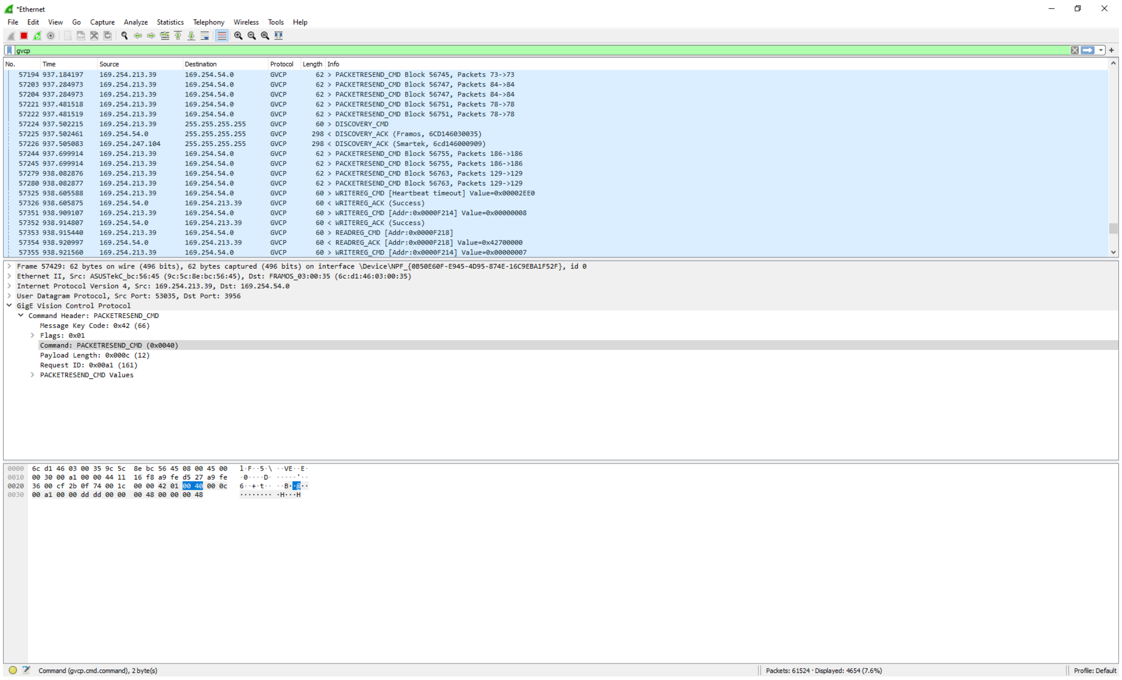


FRAMOS D435e s.n:6CD146030034 | RGB Camera, Color stream
Time: 1575528853075.2 Frame: 10922 1280x720, RGB8, FPS: 26.63

FRAMOS D435e s.n:6CD146030034 | Stereo Module, Depth stream
Time: 1575528853073.0 Frame: 10771 1280x720, Z16, FPS: 23.21

Figure 9 – Inter Packet Delay too small - frames received, frame rate drops

For Case 2, there are a smaller number of Packet Resend requests comparing to the Case 1, as shown on Figure 10.


*Ethernet
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
gvcp

```
No.   Time        Source            Destination       Protocol Length Info
57194 937.184197  169.254.213.39    169.254.54.0      GVCP     62 > PACKETRESEND_CMD Block 56745, Packets 73->73
57203 937.284973  169.254.213.39    169.254.54.0      GVCP     62 > PACKETRESEND_CMD Block 56747, Packets 84->84
57204 937.284973  169.254.213.39    169.254.54.0      GVCP     62 > PACKETRESEND_CMD Block 56747, Packets 84->84
57221 937.481518  169.254.213.39    169.254.54.0      GVCP     62 > PACKETRESEND_CMD Block 56751, Packets 78->78
57222 937.481519  169.254.213.39    169.254.54.0      GVCP     62 > PACKETRESEND_CMD Block 56751, Packets 78->78
57224 937.502215  169.254.213.39    255.255.255.255   GVCP     60 > DISCOVERY_CMD
57225 937.502461  169.254.54.0      255.255.255.255   GVCP    298 < DISCOVERY_ACK (Framos, 6CD146030035)
57226 937.505083  169.254.247.104   255.255.255.255   GVCP    298 < DISCOVERY_ACK (Smartek, 6cd146000909)
57244 937.699914  169.254.213.39    169.254.54.0      GVCP     62 > PACKETRESEND_CMD Block 56755, Packets 186->186
57245 937.699914  169.254.213.39    169.254.54.0      GVCP     62 > PACKETRESEND_CMD Block 56755, Packets 186->186
57279 938.082876  169.254.213.39    169.254.54.0      GVCP     62 > PACKETRESEND_CMD Block 56763, Packets 129->129
57280 938.082877  169.254.213.39    169.254.54.0      GVCP     62 > PACKETRESEND_CMD Block 56763, Packets 129->129
57325 938.605588  169.254.213.39    169.254.54.0      GVCP     60 > WRITEREG_CMD [Heartbeat timeout] Value=0x00002EE0
57326 938.605875  169.254.54.0      169.254.213.39    GVCP     60 < WRITEREG_ACK (Success)
57351 938.909107  169.254.213.39    169.254.54.0      GVCP     60 > WRITEREG_CMD [Addr:0x0000F214] Value=0x00000008
57352 938.914807  169.254.54.0      169.254.213.39    GVCP     60 < WRITEREG_ACK (Success)
57353 938.915440  169.254.213.39    169.254.54.0      GVCP     60 > READREG_CMD [Addr:0x0000F218]
57354 938.920997  169.254.54.0      169.254.213.39    GVCP     60 < READREG_ACK [Addr:0x0000F218] Value=0x42700000
57355 938.921560  169.254.213.39    169.254.54.0      GVCP     60 > WRITEREG_CMD [Addr:0x0000F214] Value=0x00000007
```

```
> Frame 57429: 62 bytes on wire (496 bits), 62 bytes captured (496 bits) on interface \Device\NPF_{0B50E60F-E945-4D95-874E-16C9EBA1F52F}, id 0
> Ethernet II, Src: ASUSTekC_bc:56:45 (9c:5c:8e:bc:56:45), Dst: FRAMOS_03:00:35 (6c:d1:46:03:00:35)
> Internet Protocol Version 4, Src: 169.254.213.39, Dst: 169.254.54.0
> User Datagram Protocol, Src Port: 53035, Dst Port: 3956
v GigE Vision Control Protocol
  v Command Header: PACKETRESEND_CMD
      Message Key Code: 0x42 (66)
    > Flags: 0x01
      Command: PACKETRESEND_CMD (0x0040)
      Payload Length: 0x000c (12)
      Request ID: 0x00a1 (161)
    > PACKETRESEND_CMD Values
```

```
0000  6c d1 46 03 00 35 9c 5c  8e bc 56 45 08 00 45 00   l·F··5·\ ··VE··E·
0010  00 30 00 a1 00 00 44 11  16 f8 a9 fe d5 27 a9 fe   ·0····D· ·····'··
0020  36 00 cf 2b 0f 74 00 1c  00 00 42 01 00 40 00 0c   6··+·t·· ··B··@··
0030  00 a1 00 00 dd dd 00 00  00 48 00 00 00 48         ········ ·H···H
```

```
Command (gvcp.cmd.command), 2 byte(s)          Packets: 61524 · Displayed: 4654 (7.6%)          Profile: Default
```

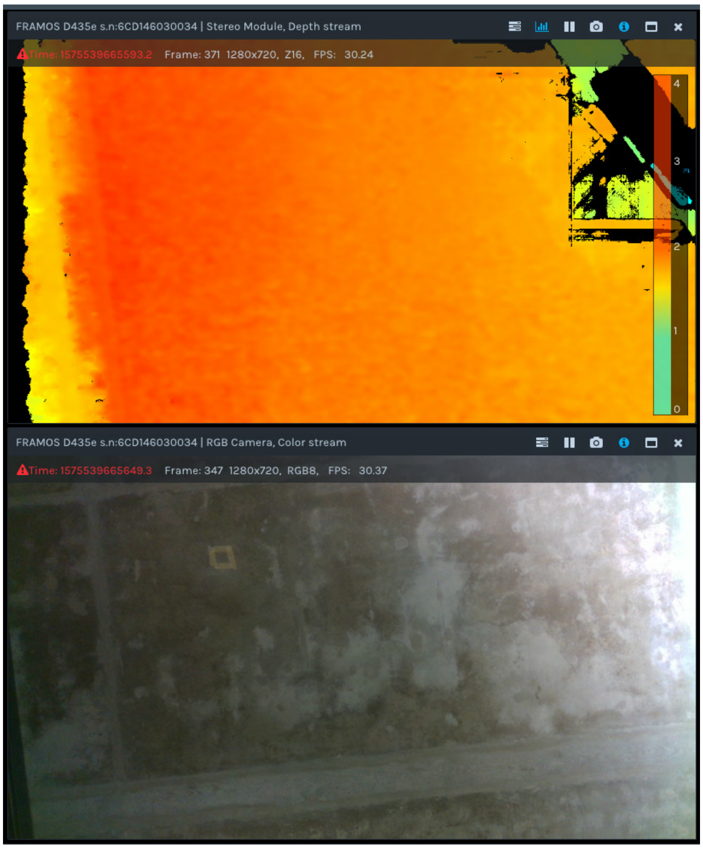
Figure 10 – Wireshark log for Packet Resend requests, Case 2

Case 3:
Stereo Module Inter Packet Delay: 65.
Stereo Module Packet Size: 7996.
RGB Camera Inter Packet Delay: 65.
RGB Camera Packet Size: 7996.

Inter Packet Delay: 65.000

Packet Size: 7996.000

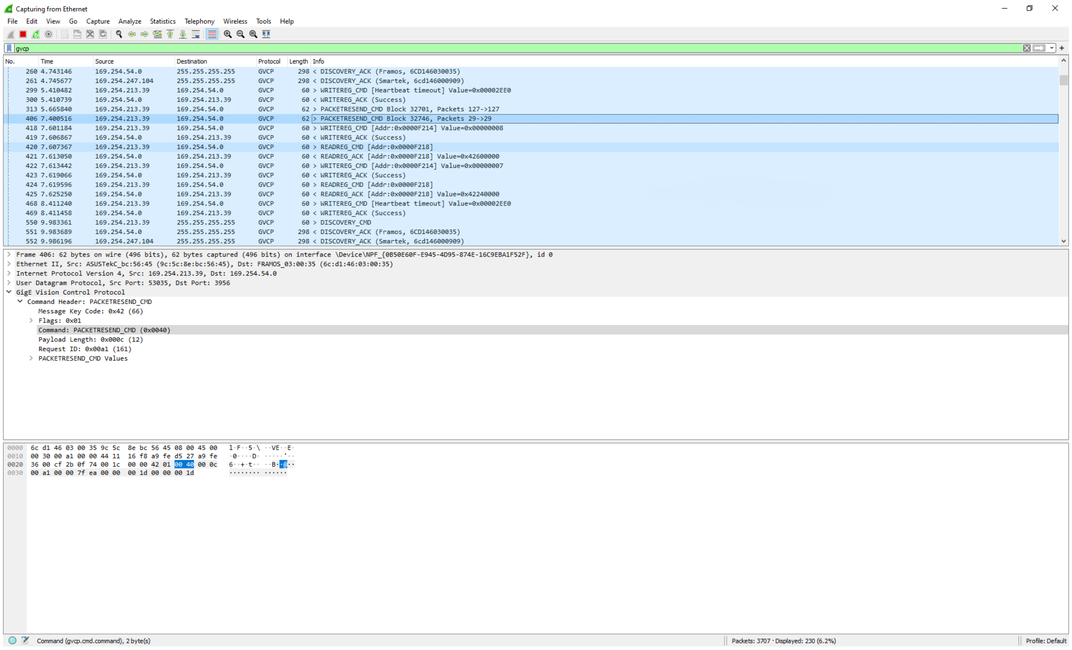
With further increased Inter Packet Delay value for both Stereo and RGB modules, streams are stable at 30FPS.



FRAMOS D435e s.n:6CD146030034 | Stereo Module, Depth stream
Time: 1575539665593.2 Frame: 371 1280x720, Z16, FPS: 30.24
4
3
2
1
0
FRAMOS D435e s.n:6CD146030034 | RGB Camera, Color stream
Time: 1575539665649.3 Frame: 347 1280x720, RGB8, FPS: 30.37

Figure 11 – Inter Packet Delay optimal - frames received, frame rate stable

Analyzing the network traffic with Wireshark for Case 3 shows that the number of Packet Resend requests dropped to minimum as shown on Figure 12.



Capturing from Ethernet
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
gvcp

```
No.   Time       Source            Destination       Protocol  Length Info
  260 4.743146   169.254.54.0      255.255.255.255   GVCP      298 < DISCOVERY_ACK (Framos, 6CD146030035)
  261 4.745677   169.254.247.104   255.255.255.255   GVCP      298 < DISCOVERY_ACK (Smartek, 6cd146000909)
  299 5.410482   169.254.213.39    169.254.54.0      GVCP       60 > WRITEREG_CMD [Heartbeat timeout] Value=0x00002EE0
  300 5.410739   169.254.54.0      169.254.213.39    GVCP       60 < WRITEREG_ACK (Success)
  313 5.665840   169.254.213.39    169.254.54.0      GVCP       62 > PACKETRESEND_CMD Block 32701, Packets 127->127
  406 7.400516   169.254.213.39    169.254.54.0      GVCP       62 > PACKETRESEND_CMD Block 32746, Packets 29->29
  418 7.601184   169.254.213.39    169.254.54.0      GVCP       60 > WRITEREG_CMD [Addr:0x0000F214] Value=0x00000008
  419 7.606867   169.254.54.0      169.254.213.39    GVCP       60 < WRITEREG_ACK (Success)
  420 7.607367   169.254.213.39    169.254.54.0      GVCP       60 > READREG_CMD [Addr:0x0000F218]
  421 7.613050   169.254.54.0      169.254.213.39    GVCP       60 < READREG_ACK [Addr:0x0000F218] Value=0x42600000
  422 7.613442   169.254.213.39    169.254.54.0      GVCP       60 > WRITEREG_CMD [Addr:0x0000F214] Value=0x00000007
  423 7.619066   169.254.54.0      169.254.213.39    GVCP       60 < WRITEREG_ACK (Success)
  424 7.619596   169.254.213.39    169.254.54.0      GVCP       60 > READREG_CMD [Addr:0x0000F218]
  425 7.625250   169.254.54.0      169.254.213.39    GVCP       60 < READREG_ACK [Addr:0x0000F218] Value=0x42240000
  468 8.411240   169.254.213.39    169.254.54.0      GVCP       60 > WRITEREG_CMD [Heartbeat timeout] Value=0x00002EE0
  469 8.411458   169.254.54.0      169.254.213.39    GVCP       60 < WRITEREG_ACK (Success)
  550 9.983361   169.254.213.39    255.255.255.255   GVCP       60 > DISCOVERY_CMD
  551 9.983689   169.254.54.0      255.255.255.255   GVCP      298 < DISCOVERY_ACK (Framos, 6CD146030035)
  552 9.986196   169.254.247.104   255.255.255.255   GVCP      298 < DISCOVERY_ACK (Smartek, 6cd146000909)
```

```
> Frame 406: 62 bytes on wire (496 bits), 62 bytes captured (496 bits) on interface \Device\NPF_{0B50E60F-E945-4D95-874E-16C9EBA1F52F}, id 0
> Ethernet II, Src: ASUSTekC_bc:56:45 (9c:5c:8e:bc:56:45), Dst: FRAMOS_03:00:35 (6c:d1:46:03:00:35)
> Internet Protocol Version 4, Src: 169.254.213.39, Dst: 169.254.54.0
> User Datagram Protocol, Src Port: 53035, Dst Port: 3956
v GigE Vision Control Protocol
  v Command Header: PACKETRESEND_CMD
      Message Key Code: 0x42 (66)
    > Flags: 0x01
      Command: PACKETRESEND_CMD (0x0040)
      Payload Length: 0x000c (12)
      Request ID: 0x00a1 (161)
    > PACKETRESEND_CMD Values
```

```
0000  6c d1 46 03 00 35 9c 5c  8e bc 56 45 08 00 45 00   l·F··5·\ ··VE··E·
0010  00 30 00 a1 00 00 44 11  16 f8 a9 fe d5 27 a9 fe   ·0····D· ·····'··
0020  36 00 cf 2b 0f 74 00 1c  00 00 42 01 00 40 00 0c   6··+·t·· ··B··@··
0030  00 a1 00 00 7f ea 00 00  00 1d 00 00 00 1d         ········ ······
```

Command (gvcp.cmd.command), 2 byte(s) | Packets: 3707 · Displayed: 230 (6.2%) | Profile: Default

Figure 12 – Wireshark log for Packet Resend requests, Case 3

Information: Regardless of the Inter Packet Delay value set, Packet Resend requests can be observed occasionally as a reaction to packet loss, which is typically a result of high system or network load.

3.1.2 Example for Three Cameras Streaming to One NIC

Three cameras are streaming packets of size 1500 bytes to one NIC simultaneously. Inter Packet Delay should be set such that packets from all three cameras are serialized to the PC's Network Interface Card. Setting inter packet delay to 25us (12,3us + 12,3us ~= 25us) on each camera will ensure that packets from other two cameras will fit in the gap between two consecutive packets.

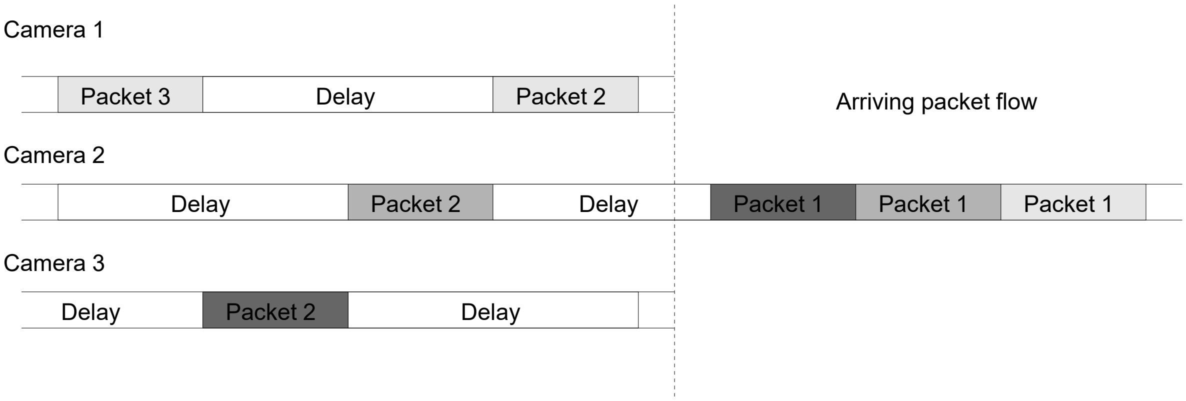
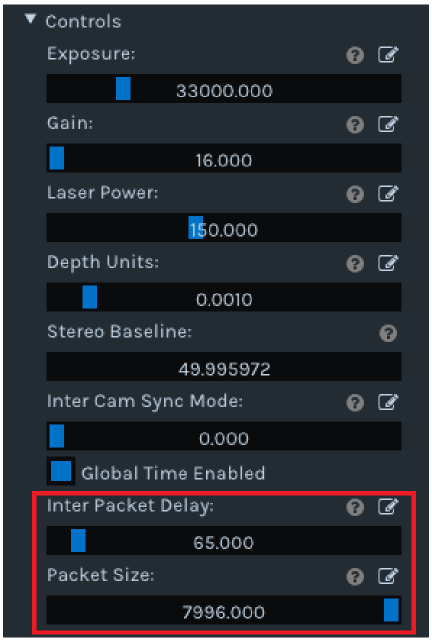


Figure 13 – Packet flow example with three cameras and properly configured inter packet delay

3.2 Setting Inter Packet Delay

Option "RS2_OPTION_INTER_PACKET_DELAY" is used for setting camera Inter Packet Delay. Example of configuring Inter Packet Delay through Intel RealSense Viewer is shown on Figure 14.

In relation with Jumbo Frames/Packets, FRAMOS D400e camera series Packet Size can be adjusted using RealSense Viewer as well.



▼ Controls

Exposure: 33000.000

Gain: 16.000

Laser Power: 150.000

Depth Units: 0.0010

Stereo Baseline: 49.995972

Inter Cam Sync Mode: 0.000

☑ Global Time Enabled

Inter Packet Delay: 65.000

Packet Size: 7996.000

Figure 14 – Inter Packet Delay feature in RealSense Viewer

To set Inter Packet Delay in code, call "set_option" function with option name and requested value as shown below.

```
// To set an option to a different value, call set_option with a new value
sensor.set_option(RS2_OPTION_INTER_PACKET_DELAY, inter_packet_delay);
```

Configuring Packet Size can be done in the same way using the following code.

```
// To set an option to a different value, call set_option with a new value
sensor.set_option(RS2_OPTION_PACKET_SIZE, packet_size);
```

3.3 Tuning Inter Packet Delay with Packet Statistic Feature

Tuning the Inter Packet Delay is an adjusting process aiming to reduce the amount of Packet Resend requests while preserving the configured frame rate.

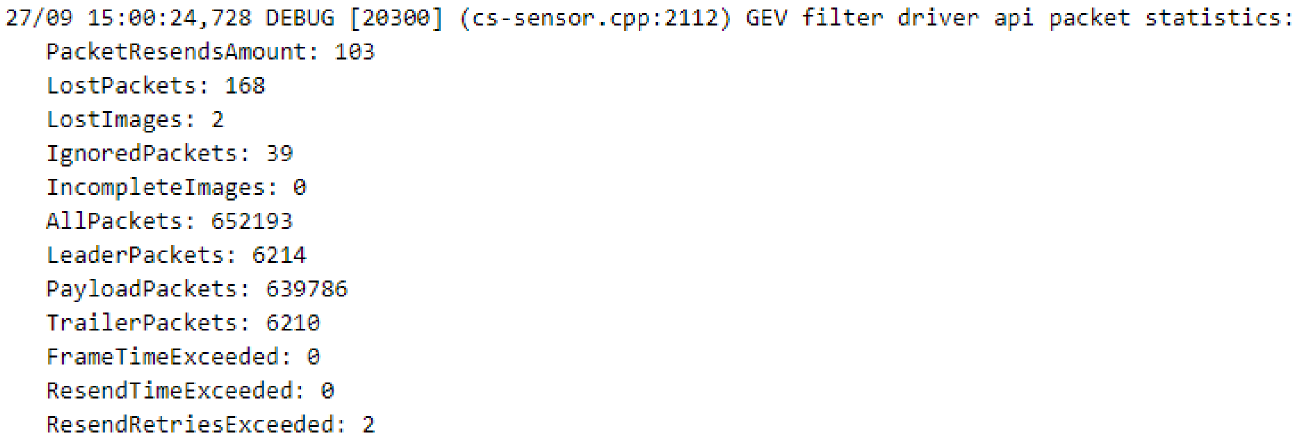
As mentioned in the Chapter “Example for adjusting Inter Packet Delay” Wireshark logging can be used to measure the amount of the packet resend requests.

Another approach is to use the Packet Statistic feature. By enabling the RealSense logging (DEBUG severity), packet statistic is logged every 10 seconds in the RealSense log file or terminal.

To enable logging in code, use the code snippet as shown below.

```
//C++
int main(int argc, char * argv[]) try
{
    rs2::log_to_file(RS2_LOG_SEVERITY_DEBUG, "/path/to/file.txt");
    ...
}

#Python
rs.log_to_file(rs.log_severity.debug, file_path='./path/to/file.txt'):
```

Example of a logged packet statistic is given below (screenshot from the log file).



```
27/09 15:00:24,728 DEBUG [20300] (cs-sensor.cpp:2112) GEV filter driver api packet statistics:
   PacketResendsAmount: 103
   LostPackets: 168
   LostImages: 2
   IgnoredPackets: 39
   IncompleteImages: 0
   AllPackets: 652193
   LeaderPackets: 6214
   PayloadPackets: 639786
   TrailerPackets: 6210
   FrameTimeExceeded: 0
   ResendTimeExceeded: 0
   ResendRetriesExceeded: 2
```

Tune the Inter Packet Delay while monitoring the PacketResendsAmount. This can be done during the stream.

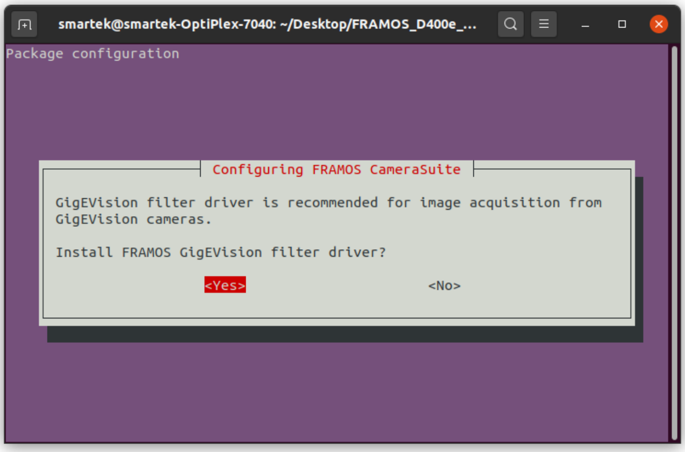
Information: Packet Statistic feature is available from the D400e software package v2.2.0 (or higher).

3.4 GigE Vision Drivers

D400e software package comes with two different GigE Vision (GEV) drivers:

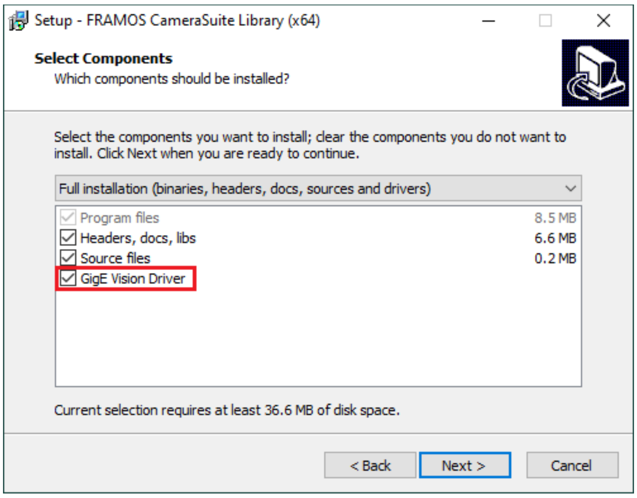
- FRAMOS GigE Vision filter driver – high-performance network filter driver
- GigE Vision socket driver – a driver that uses system socket API

The GigE Vision socket driver is part of the CameraSuite library while the FRAMOS GigE Vision filter driver is standalone installation package. During the installation of a Camera Suite library, user will be prompted to install the FRAMOS GigE Vision filter driver as shown in Figure 15 and Figure 16.



```
Package configuration

┤ Configuring FRAMOS CameraSuite ├

GigEVision filter driver is recommended for image acquisition from
GigEVision cameras.

Install FRAMOS GigEVision filter driver?

              <Yes>                    <No>
```

Figure 15 – CameraSuite – Filter driver installation [Linux]



Setup - FRAMOS CameraSuite Library (x64)

Select Components

Which components should be installed?

Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.

Full installation (binaries, headers, docs, sources and drivers)

☑ Program files	8.5 MB
☑ Headers, docs, libs	6.6 MB
☑ Source files	0.2 MB
☑ GigE Vision Driver	

Current selection requires at least 36.6 MB of disk space.

Figure 16 – CameraSuite – Filter driver installation [Windows]

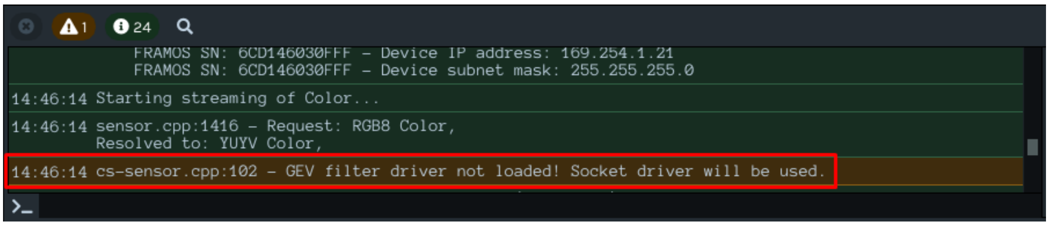
Information: Only one driver can be used on a system. FRAMOS recommends the usage of a filter driver which is designed to ensure optimal performance.

Information: Tunning of system settings should be applied regardless of which version of the driver is used.

Information: Socket driver is available from the D400e software package v2.4.0. (or higher).

3.4.1 Checking the Driver Used

The filter driver has precedence over the socket driver. The CameraSuite library will check if the filter driver is present on the system and will try to load it. If the filter driver is not installed or fails to load, socket driver will be used. This information is available through a RealSense log mechanism. The RealSense log mechanism will indicate this information on start of an image stream. An example of a log from RealSense Viewer is depicted in Figure 17.



```
             FRAMOS SN: 6CD146030FFF - Device IP address: 169.254.1.21
             FRAMOS SN: 6CD146030FFF - Device subnet mask: 255.255.255.0
14:46:14 Starting streaming of Color...
14:46:14 sensor.cpp:1416 - Request: RGB8 Color,
         Resolved to: YUYV Color,
14:46:14 cs-sensor.cpp:102 - GEV filter driver not loaded! Socket driver will be used.
>_
```

Figure 17 – RealSense Viewer using socket driver

4. Revision History

Date	Version	Changes
2020-06-30	1.0.0	Initial release
2021-07-15	1.1.0	Added chapter "Tuning Inter Packet Delay with Packet Statistic Feature"
2021-10-15	1.2.0	Updated chapter "Tuning Inter Packet Delay with Packet Statistic Feature"; added Chapter "GigE Vision Drivers"

Table 1 – Revision History

NOTE: *This document replaces and supersedes the application note "FRAMOS Industrial Depth Camera D435e - Tuning System for Best Performance" v1.0.0.*