# **✅ Unitree L2 LiDAR Setup Guide for UDP over Ethernet**

This guide outlines the **correct procedure to configure, compile, and test** Unitree L2 LiDAR in **UDP mode over Ethernet** using the Unitree SDK.

## **🧾 System Setup Summary**

* **LiDAR IP**: 192.168.1.4
* **PC IP**: 192.168.1.3
* **Serial Port**: /dev/ttyACM0
* **Baudrate**: 4000000
* **UDP Mode WorkMode**: 0
* **Serial Mode WorkMode**: 8

## **🛠️ Edit Required Example Files (in unitree\_lidar\_sdk/examples)**

### **1. set\_to\_udp\_mode.cpp ✅**

**No changes needed**. Uses:

initializeSerial("/dev/ttyACM0", 4000000);

setLidarWorkMode(0); // UDP

### **2. set\_to\_serial\_mode.cpp ❌**

Update IPs:

std::string lidar\_ip = "192.168.1.4";

std::string local\_ip = "192.168.1.3";

### **3. set\_ip\_address.cpp ❌**

* Replace UDP init with Serial:

lreader->initializeSerial("/dev/ttyACM0", 4000000);

* Update IP config:

config.lidar\_ip = {192, 168, 1, 4};

config.user\_ip = {192, 168, 1, 3};

### **4. example\_lidar\_udp.cpp ✅**

Already correct:

lidar\_ip = "192.168.1.4";

local\_ip = "192.168.1.3";

workMode = 0;

### **5. example\_lidar\_serial.cpp ✅**

Already correct:

initializeSerial("/dev/ttyACM0", 4000000);

workMode = 8;

## **🔧 Build Instructions**

cd ~/unilidar\_sdk2/unitree\_lidar\_sdk/build

cmake ..

make -j4

## **🔁 Configuration and Execution Sequence**

### **1. Set IP Address Over Serial**

../bin/set\_ip\_address

Expect:

Unilidar initialization succeed!

Lidar IP is reset! Please reboot lidar!

### **2. Power Cycle the LiDAR**

Unplug and replug the LiDAR power. Wait 10 seconds.

### **3. Set to UDP Mode**

../bin/set\_to\_udp\_mode

### **4. Configure Network on PC**

sudo ip addr add 192.168.1.3/24 dev enp0s31f6

sudo ip link set dev enp0s31f6 up

### **5. Ping the LiDAR**

ping 192.168.1.4

Expect 0% packet loss.

### **6. Test UDP Communication**

../bin/example\_lidar\_udp

Expect to see live data streaming.

## **🚀 ROS2 Node Launch Setup**

### **launch.py Parameter Block (must match exactly):**

parameters=[

{'initialize\_type': 2},

{'work\_mode': 0},

{'use\_system\_timestamp': True},

{'range\_min': 0.0},

{'range\_max': 100.0},

{'cloud\_scan\_num': 18},

{'lidar\_ip': '192.168.1.4'},

{'local\_ip': '192.168.1.3'},

{'lidar\_port': 6101},

{'local\_port': 6201},

{'cloud\_frame': 'unilidar\_lidar'},

{'cloud\_topic': 'unilidar/cloud'},

{'imu\_frame': 'unilidar\_imu'},

{'imu\_topic': 'unilidar/imu'},

]

❗ Remove any serial\_port or baudrate entries unless using serial mode.

### **✅ Support both Serial and UDP (Optional Dynamic Setup)**

Instead of hardcoding, use:

use\_udp = True

params = [

{'initialize\_type': 2 if use\_udp else 1},

{'work\_mode': 0 if use\_udp else 8},

{'use\_system\_timestamp': True},

{'range\_min': 0.0},

{'range\_max': 100.0},

{'cloud\_scan\_num': 18},

]

if use\_udp:

params += [

{'lidar\_ip': '192.168.1.4'},

{'local\_ip': '192.168.1.3'},

{'lidar\_port': 6101},

{'local\_port': 6201},

]

else:

params += [

{'serial\_port': '/dev/ttyACM0'},

{'baudrate': 4000000},

]

### **🔧 Fix ROS2 Node Name Bug**

In unitree\_lidar\_ros2.h, update:

: Node("unitree\_lidar\_ros2\_node", options)

This name must match name='unitree\_lidar\_ros2\_node' in launch.py.

If it's incorrect (like unitre\_lidar\_sdk\_node), parameters won't be passed.

### **🔧 Pass NodeOptions in Main**

In unitree\_lidar\_ros2\_node.cpp, ensure:

rclcpp::spin(std::make\_shared<UnitreeLidarSDKNode>(options));

## **✅ Confirm Success**

* ping 192.168.1.4 works
* ../bin/example\_lidar\_udp streams data
* ros2 topic echo /unilidar/cloud shows point cloud
* RViz displays /unilidar/cloud with frame unilidar\_lidar

You’re done! 🎉

Let me know if you want to automate this whole setup with a shell script.