GO1 goes internet

Network config hints for Unitree Go1

2023-07-14 Markus Schepp schepplications.de

Abstract and disclaimer

The Unitree Go1 has a closed system of network devices accessible via an AP.
To extend the bot with other devices and sensors an external ethernet interface is available.
But how can i connect to internal and external devices from the internet or from my local wifi?

This little guide wants to give you some help.

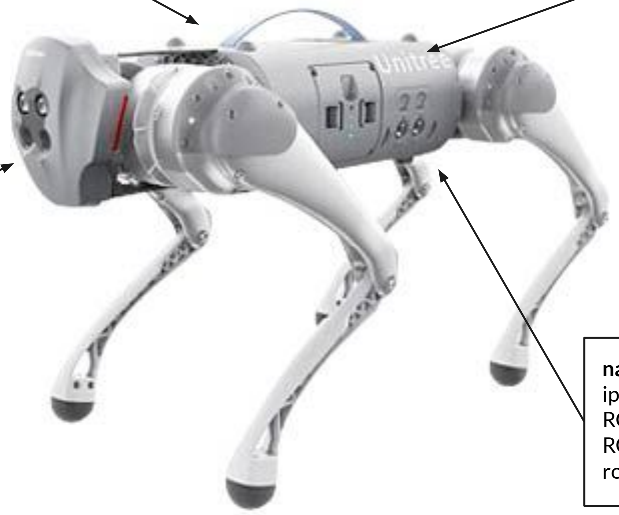
Important!
Connecting the GO1 to the internet implies a potential security issue as long as the installed Ubuntu 18.0.4 is NOT anymore supported and no security mechanisms are implemented on the GO1.
You should NOT try to update or upgrade the GO1 systems as long as all functionalities are only testet with the delivered Ubuntu 18.0.4 .
Any changes on the systemor part of the system are your own responsibility.

base config

nano2 192.168.123.14/24
ip r: default via 192.168.123.1
ROS_MASTER_URL=http://192.168.123.161:11311
ROS_IP=192.168.123.14
ros node: left/rightcam

raspi 192.168.123.161/24
ip r: default via 192.168.123.1
ip r: default via 192.168.12.1
ROS_MASTER_URL=http://192.168.123.161:11311
ROS_IP=192.168.123.161
ros node: right side cam
this is the ros master!

head nano (nano1) 192.168.123.13/24
ip r: default via 192.168.123.1
ROS_MASTER_URL=http://192.168.123.161:11311
ROS_IP=192.168.123.13
ros node: front cam, pointcloud
other processes: wsaudio (speaker/mic)



nano3 192.168.123.15/24
ip r: default via 192.168.123.1
ROS_MASTER_URL=http://192.168.123.161:11311
ROS_IP=192.168.123.14
ros node: bottom cam

General hints

Just use ip-adresses instead of names. For local and internet name resolution you have take care of proper installation of host configs or dns configs. If you only work with the sdk, ros and mqtt there is no need to use names.

Take a look at energy consumption of connected devices or sensors/actors. The USB-ports of the internal raspi/nano seem to not support more than about 500mA stable.

Using internal wlan0 on the raspi

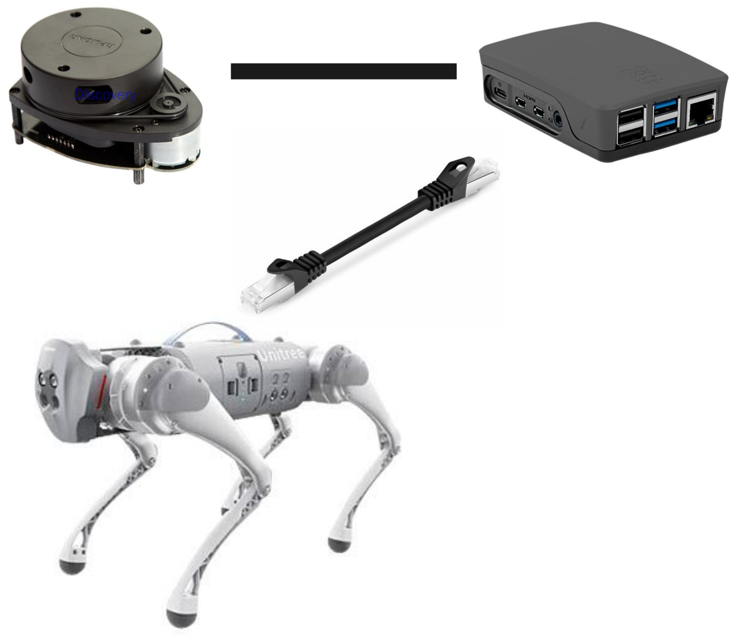
Edit hostpad.conf to let the wlan0 know the credentials of your AP

```
# allow forwarding on the external raspi
# can be set static in /etc/sysctl.conf or called in a shell script

sysctl -w net.ipv4.ip_forward=1

# change ip settings so packages are routed from eth0 to wlan0 and vice versa:
# forward ip traffic from internal lan to the wlan

iptables -A FORWARD -i eth0 -o wlan0 -j ACCEPT
iptables -A FORWARD -i wlan0 -o eth0 -m state --state RELATED,ESTABLISHED -j ACCEPT
```

Using GSM on the internal raspi USB port

```
# delete existing ip routes before plugging in the usb modem

ip r delete default via 192.168.123.1
ip r delete  default via 192.168.12.1

# change ip settings so packages are routed from eth0 to wlan0 and vice versa:
# allow forwarding on the external raspi
# can be set static in /etc/sysctl.conf or called in a shell script

sysctl -w net.ipv4.ip_forward=1

# forward ip traffic from internal lan to the wlan

iptables -A FORWARD -i eth0 -o <your gsm device> -j ACCEPT
iptables -A FORWARD -i <your gsm device> -o eth0 -m state --state RELATED,ESTABLISHED -j ACCEPT

# masquerade nano traffic behind raspberry traffic for GSM

iptables -t nat -A POSTROUTING -o usb0 -j MASQUERADE

# note: in my case the connected LTE/USB modem device was
usb0
```

Adding a “backplate”



external raspi 192.168.123.1/24
ip r: default via <your connected wlan>
# publish to the existing ros-master (internal raspi) like the internal nanos do)
ROS_MASTER_URL=http://192.168.123.161:11311
ROS_IP=192.168.123.1
ros node: <your ros nodes e.g. LiDar>

external raspi 192.168.123.1/24

```
# allow forwarding on the external raspi
# can be set static in /etc/sysctl.conf or called in a shell script
sysctl -w net.ipv4.ip_forward=1


# forward ip traffic from internal lan to the wlan
iptables -A FORWARD -i eth0 -o <your wlan device> -j ACCEPT
iptables -A FORWARD -i <your wlan device> -o eth0 -m state --state RELATED,ESTABLISHED -j ACCEPT

# for GSM connections:
# masquerade nano traffic behind raspberry traffic for the external connection
iptables -t nat -A POSTROUTING -o <your GSM device> -j MASQUERADE
```