The ODROID H2 Net card brings 4 additional 2.5 GbE Ethernet ports for an incredible low price. It is compatible with the H2, H2 rev. B and H2+. Using a H2+ you end up with a total of 6 x 2.5 GbE Ethernet ports. With the H2 and H2 rev. B you get 2 x 1 GbE and 4 x 2.5 GbE Ethernet ports.

The ODROID H2/H2+ with the ODROID H2 Net card brings new capabilities and applications. Examples (non-exhaustive list):

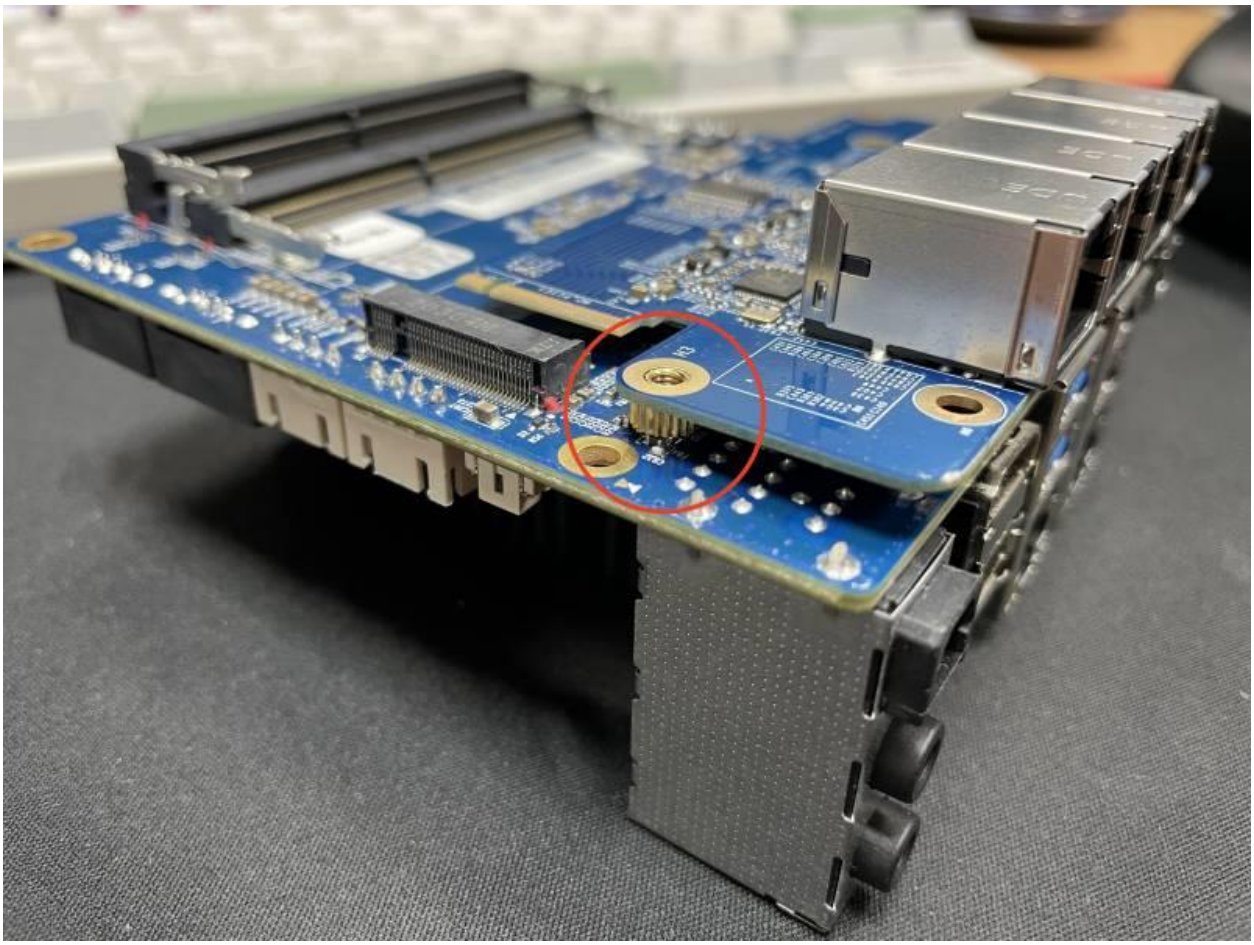
- Implementing a DIY router or gateway for up to 5 (H2+) clients at 2.5 GbE speed on each port.
- Setup a DIY server, e.g. a NAS, for up to 6 (H2+) clients at 2.5 GbE speed on each port.
- Brings 2.5 GbE Ethernet connectivity to the H2 and H2 rev. B boards making them able to talk at full speed (usually 2.35 Gbits/sec) with H2+ boards.

Hardware Information

- Schematics
- AutoCAD top
- AutoCAD bottom

Installation Precautions

- We strongly RECOMMEND you watch the **(WIP) How to install the H2 Net Card** video to prevent any damages, like for instance touching the H2/H2+ components with the two brass nuts on the H2 net card. The installation is rather a simple and easy affair, you just have to be gentle. The card provides a quite compact assembly solution using the M.2 slot and the two brass nuts as attach points (no extra M.2 to PCIe cable adapter necessary).



Compatible Odroid-H2 Cases

We are releasing updated cases slightly taller to accommodate the H2 Net card. These new case types are:

- Odroid-H2 Case Type 5
- Odroid-H2 Case Type 6
- Odroid-H2 Case Type 7

ESF BIOS

While the M.2 slot is primarily intended to be used with a NVMe PCIe x4 SSD it can alternatively be used to plug in a PCIe Gen 2 card like graphics cards, IO cards, network cards, etc. Our users use a M.2 to PCIe cable adapter to do so. You can buy these cables

from 3rd party vendors or Hard Kernel products distributors (e.g. https://ameridroid.com/collections/new/products/m-2-to-pcie-adapter-straight).

The H2 Net Card connects to the H2/H2+ board using the M.2 slot to leverage the PCIe Gen2 x4 lanes. The H2 Net Card expects the M.2 slot to provide 4 bifurcated x1 lanes. However, the default M.2 slot configuration provides 1 non-bifurcated x4 lane by default.

The M.2/PCIe Gen 2 slot lanes bifurcation is configured in the ESF BIOS itself.

Starting with version 1.22 of the H2/H2+ ESF BIOS, we provide and will provide two versions:

- First one with the M.2 slot configured as 1 non-bifurcated x4 lane (for SSD or 3r party PCIe cards)
- Second one with the M.2 slot configured as 4 bifurcated x1 lanes for the H2 Net Card.

To use the H2 Net Card you MUST flash the H2/H2+ BIOS with the second version.

You can download the latest ESF BIOS from our regular release page.

- H2 BIOS Release Notes
- BIOS Archive

- Make sure the process of flashing the H2/H2+ BIOS is not interrupted by a power outage. This will render your board inoperable and you will have to return it to Hardkernel or the distributor to be restored. The best is to perform the flashing with the H2/H2+ power brick behind a UPS.

- If you do not flash the H2/H2+ BIOS with the right version, nothing bad happens but you will only see 3 Ethernet ports, the two from the H2/H2+ and only the first one from the H2 Net Card. If you only see 3 Ethernet ports after installing the correct version, reboot the H2/H2+. If again you only see 3 Ethernet ports after rebooting, try plugging out the power cord and reconnecting it.
- If later on you decide to move the H2 Net card from one H2/H2+ to another and start using back an SSD just re-flash the H2/H2+ BIOS with the M.2 slot configured as 1 non-bifurcated x4 lane version.

Flashing the H2/H2+ BIOS with the bifurcated version and installing the H2 Net card changes the PCIe buses configuration. Consequently you need to reconfigure your Ethernet ports accordingly.

On Linux for instance:

- With the non-bifurcated M.2 BIOS and without the H2 Net card the two onboard Ethernet ports show up as enp2s0 and enp3s0.

- With the bifurcated M.2 BIOS and the H2 Net card plugged in, the card 4 Ethernet ports show up as enp1s0, enp2s0, enp3s0 and enp4s0 and the two onboard NICs show up as enp5s0 and enp6s0.

- If you write yourself networked apps or scripts supposed to work in both environments you therefore have to test the configuration.

Performance Test

- The Realtek r8125 driver (for the RTL8125 and RTL8125B chipset) has had several issues on Linux and Realtek has released several versions over the last year. The most recent version, v9.004.01, does not currently lead to the best performance when using concurrent multiple 2.5GbE client sessions. So we recommend to use the previous driver, v9.003.05 for now.
    - realtek-r8125-dkms Debian package download link: https://github.com/awesometic/realtek-r8125-dkms/releases/tag/9.003.05-1

---

iperf3 multi session

To measure the network throughput with iperf3 we prepared six H2+ boards. One H2+ has a H2+ Net card (thus 6 x 2.5 GbE ports) the five others have 2 x 2.5 GbE ports. One port on each of the five H2+ is connected to a port on the H2+ with the H2+ Net card. Each IP connection is defined on a separate sub-net so that the internal routing (Windows and Linux) will use each connection independently from the others.

The five H2+ ran iperf3 as servers receiving data (iperf3 -s ...). The H2+ with the H2+ Net card ran iperf3 as client sending data (iperf -c ...).

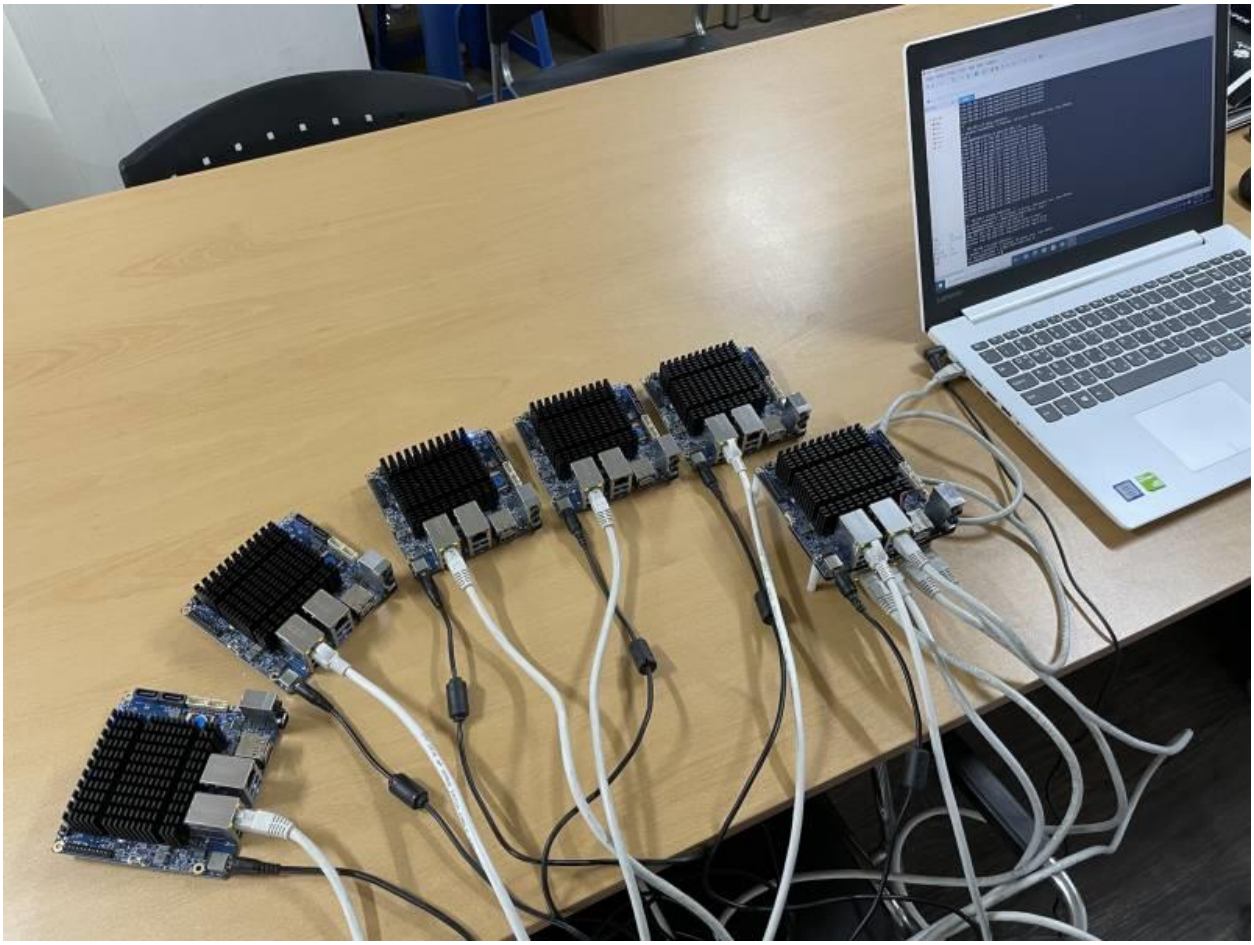
Sending mode (iperf -c ...) corresponds to a real world usage where 5 “clients” are reading data from the H2+ with the H2+ Net card, e.g. 5 clients copying file(s) from the same NAS.

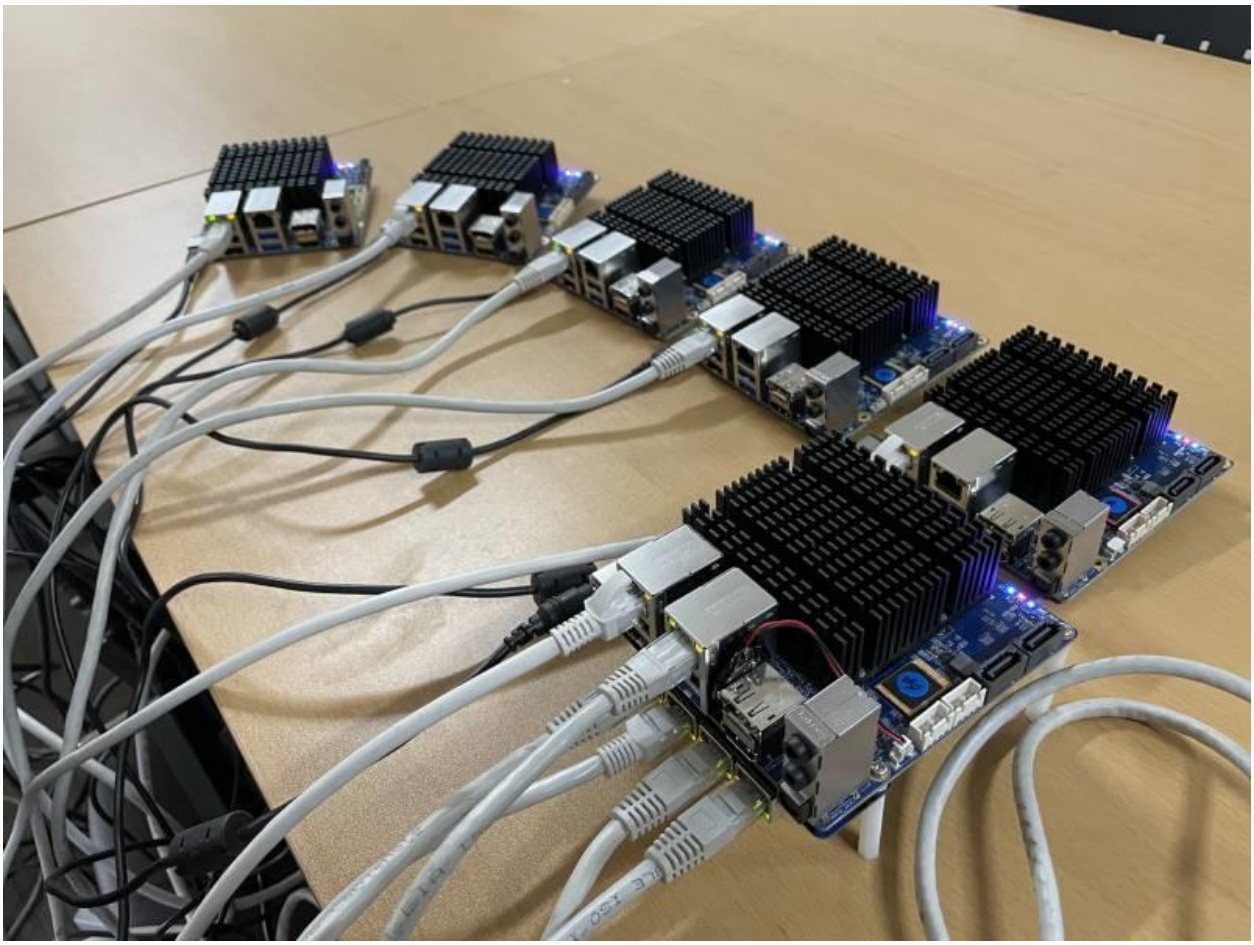
Receiving mode (iperf -c ... **-R**) corresponds to a real usage situation where 5 “clients” are sending data to the H2+ with the H2+ Net card, e.g. 5 clients copying file(s) to the same NAS.

- The iperf3 -R flag reverses the roles of the two machines. The machine which was sending data is now receiving data and the machine which as receiving data is now sending data.

The difference with real usage situations is that iperf3 does nothing with the data on the receiver side, no disk IO involved for instance: iperf3 only measures network throughput at the Internet Protocol (IP) level.

See photos shown below.





The inxi output shown below correctly reports the 6 x 2.5 GbE ports on the H2+ with the H2+ Net card, the ports being driven by the Realtek r8125 v9.003.05 driver.

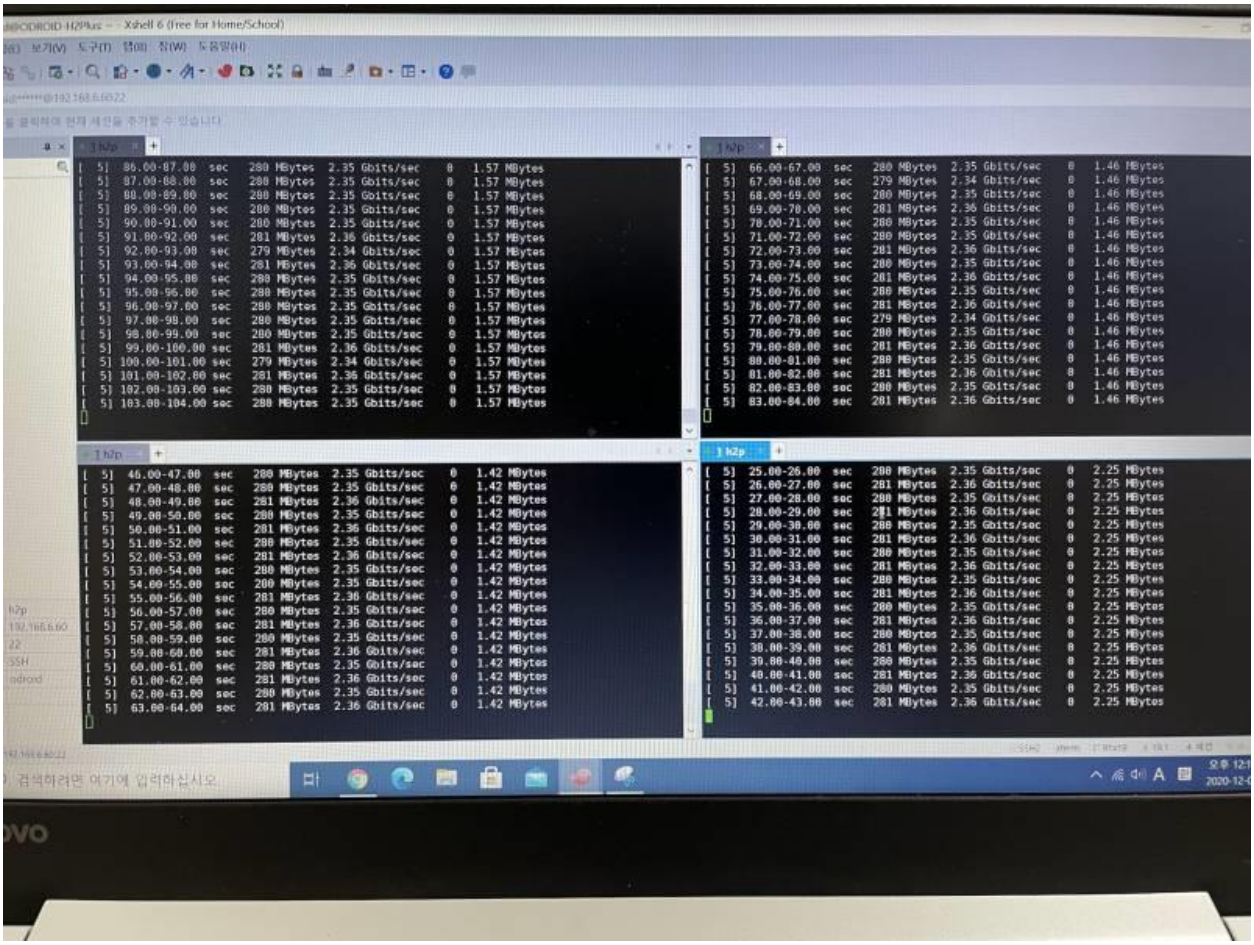
```
odroid@ODROID-H2Plus:~$ sudo inxi -CMmNS -x
System:    Host: ODROID-H2Plus Kernel: 5.4.0-56-generic x86_64 bits: 64 compiler: gcc v: 9.3.0 Console: tty 8
           Distro: Ubuntu 20.04.1 LTS (Focal Fossa)
Machine:   Type: Desktop Mobo: HARDKERNEL model: ODROID-H2 v: 1.0 serial: N/A UEFI: American Megatrends v: 1.22
           date: 11/13/2020
Memory:    RAM: total: 31.19 GiB used: 710.3 MiB (2.2%)
           Array-1: capacity: 32 GiB slots: 2 EC: None max module size: 16 GiB note: est.
           Device-1: A1_DIMM0 size: 16 GiB speed: 2400 MT/s type: DDR4
           Device-2: A1_DIMM1 size: 16 GiB speed: 2400 MT/s type: DDR4
CPU:       Topology: Quad Core model: Intel Celeron J4115 bits: 64 type: MCP arch: Goldmont Plus rev: 1 L2 cache: 4096 KiB
           flags: lm nx pae sse sse2 sse3 sse4_1 sse4_2 ssse3 vmx bogomips: 14284
           Speed: 992 MHz min/max: 800/2500 MHz Core speeds (MHz): 1: 959 2: 966 3: 960 4: 925
Network:   Device-1: Realtek RTL8125 2.5GbE driver: r8125 v: 9.003.05-NAPI port: e000 bus ID: 01:00.0
           Device-2: Realtek RTL8125 2.5GbE driver: r8125 v: 9.003.05-NAPI port: d000 bus ID: 02:00.0
           Device-3: Realtek RTL8125 2.5GbE driver: r8125 v: 9.003.05-NAPI port: c000 bus ID: 03:00.0
           Device-4: Realtek RTL8125 2.5GbE driver: r8125 v: 9.003.05-NAPI port: b000 bus ID: 04:00.0
           Device-5: Realtek RTL8125 2.5GbE driver: r8125 v: 9.003.05-NAPI port: a000 bus ID: 05:00.0
           Device-6: Realtek RTL8125 2.5GbE driver: r8125 v: 9.003.05-NAPI port: 9000 bus ID: 06:00.0
odroid@ODROID-H2Plus:~$
```

In other words, the H2+ with the H2 Net card includes:

- an Intel Celeron J4115 (Odroid-H2+),
- the Hardkernel Odroid-H2 1.22 BIOS (M.2 slot configured as 4 bifurcated x1 lanes),
- a Ubuntu 20.04.1 + 5.4.0-56 Kernel (running on a 64GB eMMC module),

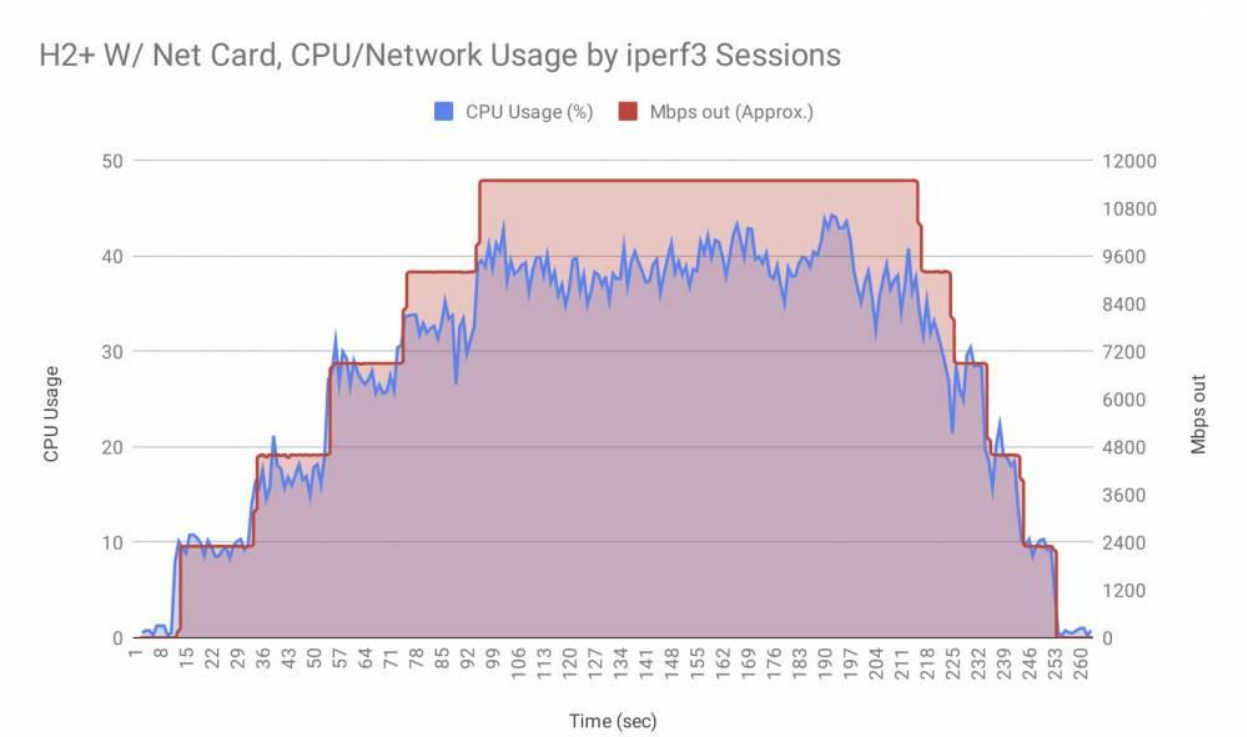
- two Samsung DDR4 2400MHz 16GB in dual channel,
- two built-in 2.5GbE and four additional 2.5GbE from the Net card. Total six 2.5 GbE NICs configured with the Realtek r8125 9.003.05 driver to get the best performance in multi-sessions.

Once the five other H2+ run iperf3 -s ..., making them able to receive data, we can run multiple sessions on the H2+ with the H2+ Net card as displayed in the screenshot shown below. Note that the sessions running in parallel are all cruising at 2.35 Gbits/sec :)

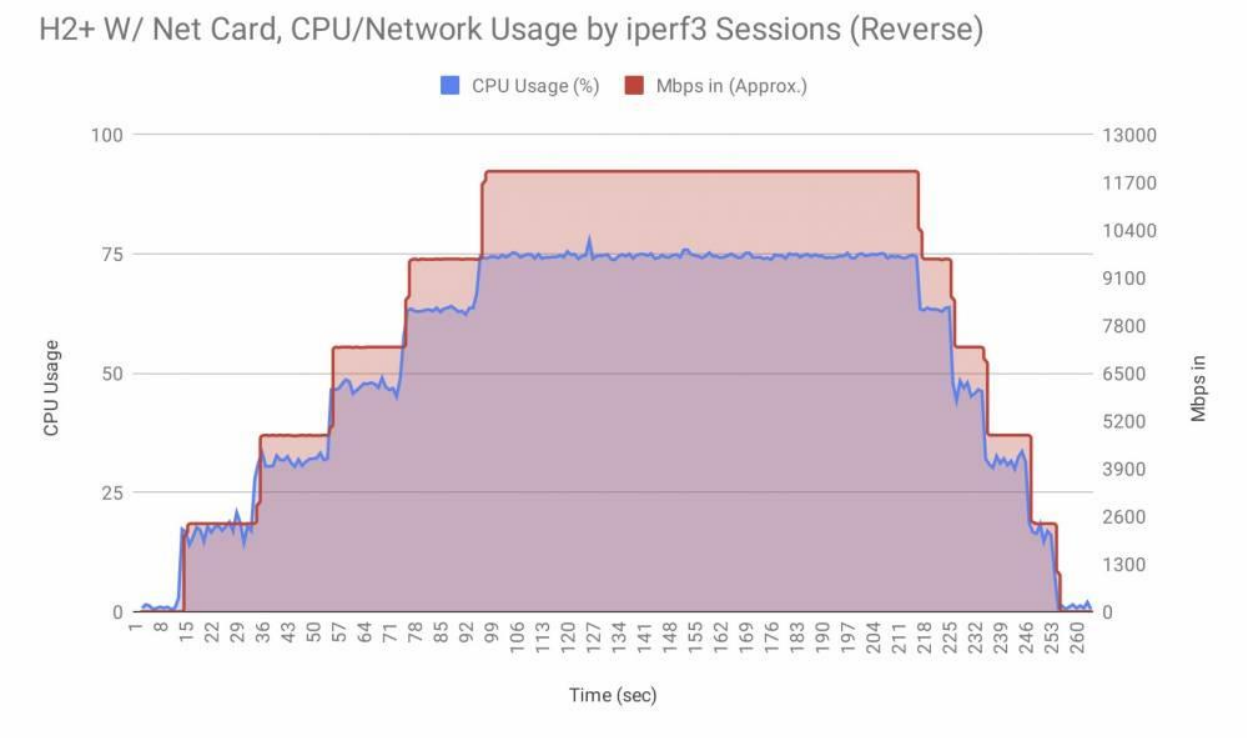


Let us now look at the results measuring 1, 2, 3, 4 and 5 parallel sessions. We measured the network throughput (Mbps) and CPU usage (%) on the H2+ with the H2 Net card.

Sending mode



Receiving mode



The measurements were made with 1 session alone for about 30 seconds, then started a 2nd session while the 1st session continued, then started a 3rd session while the two previous sessions continued and so on up to running 5 parallel sessions. As a result the network throughput (red line) has a staircase look and the CPU usage (blue line) follows the same pattern with a jagged line.

Results

In **Sending** mode the testing consumes up to 45% of the H2+ with the H2 Net card CPU resources with five 2.5 GbE iperf3 sessions, about 9% per session. All of the sessions show a solid 2.35 Gbits/sec network throughput.

In **Receiving** mode the testing consumes up to 75% of the H2+ with the H2 Net card CPU resources with five 2.5 GbE iperf3 sessions, about 15% per session. All of the sessions show a solid 2.35 Gbits/sec network throughput.

A H2+ with the H2 Net card gladly handles more than 11 Gbits/sec total. Due to this rough upper limit, it's only when running SIX parallel sessions that the throughput declines under 2.35 Gbits/sec per session.

Conclusion

A H2+ can therefore provide excellent support for parallel clients with enough remaining CPU resources to proceed with computation and disk IO.

In other words, the H2 Net card brings significant added value to a H2/H2+-based NAS or Router/Gateway solution where parallel clients can enjoy the full 2.5 GbE speed independently from each other. Especially to H2 and H2 rev. B which don't have onboard 2.5 GbE connectivity in the first place!

All with a very low cost network card :)

Samba

Samba is available on all the major Linux distributions and supports the SMB/CIFS protocol (used on Windows for file sharing) as well as Active Directory. Samba is the major protocol for Linux-based networked file servers also known as NAS (Network Attached Storage). Using Samba on Linux is a very low-cost way to share files between Windows, Linux and Mac OS clients. Linux + Samba is the tool of choice for creating DIY NAS appliances at home in SOHO environment, start-ups and small businesses.

To measure the performance of the H2 Net card with Samba we setup the following sub-net:

- One H2+ with Net Card running Linux,
- Another H2+ running Windows 10,
- A laptop also running Windows 10.

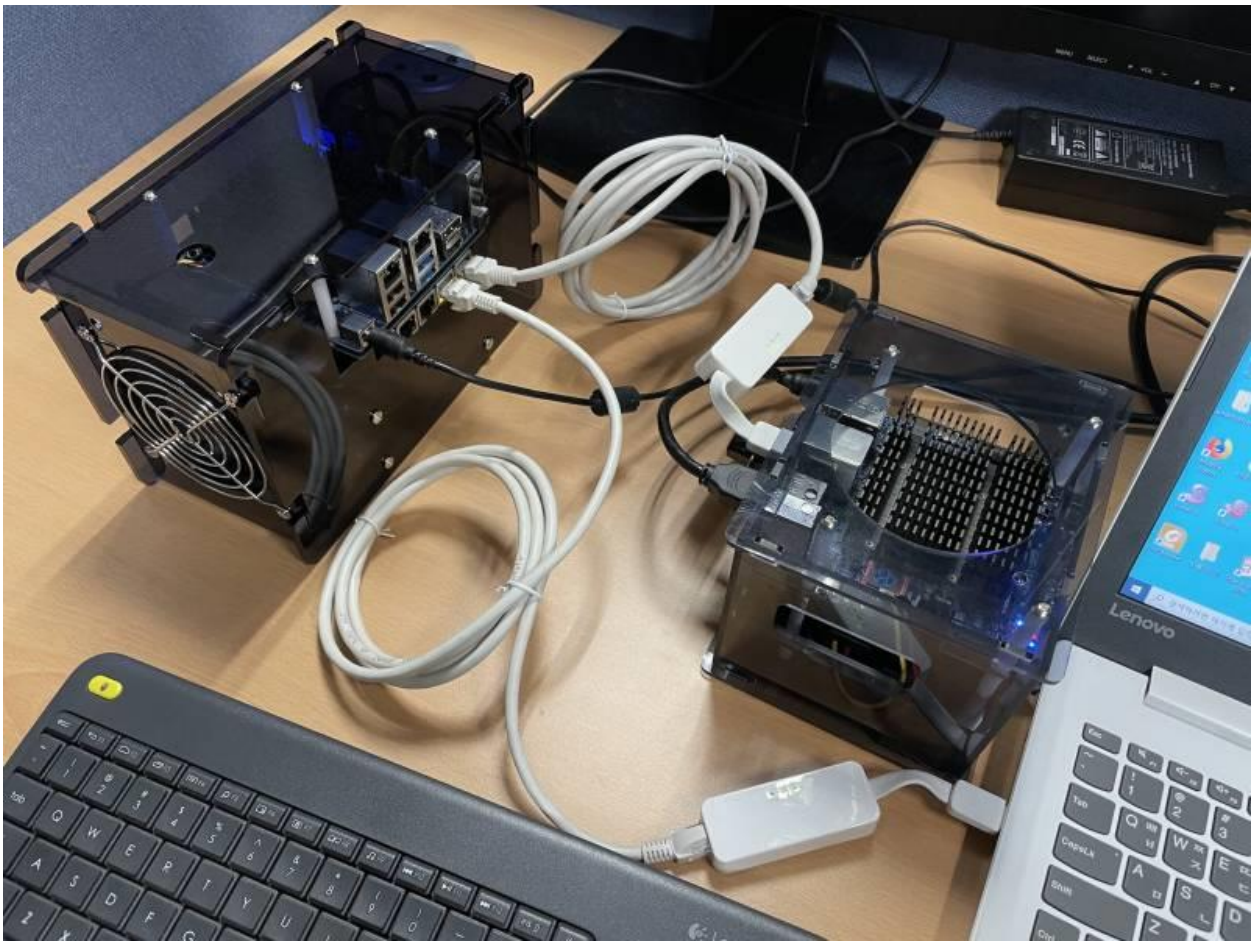
The two machines running Windows 10 are the clients. Because motherboards with an onboard 2.5 GbE NIC are still a small percentage of the market (expensive top tier motherboards), we decided to run our testing with USB 3 Gen 1 2.5 GbE USB adapters sporting a Realtek RTL RTL8156B chipset on both clients. These USB adapters can be used on any motherboard or SBC with at least one USB 3 Gen 1 port.

- Smaller ARM-based boards such as the N2/N2+ or C4, the practical max speed for a USB 3 Gen 1 2.5 GbE adapter is about 2.0 Gbits/sec or less, depending on the ARM processor power and frequency. On the other hand, as we are going to see below, USB 3 Gen 1 2.5 GbE adapters on a PC (and H2/H2+) deliver a pretty good 2.35 Gbits/sec practical max speed.

- Do not confuse the RTL8156B chipset with the RTL8125B. The former is the USB version of the Realtek implementation of 2.5 GbE, the later is the PCIe version (the RTL8125B is what Hard Kernel uses on the H2+ and H2 Net card as onboard NICs).

- Realtek released two versions of the RTL8156 chipset as of this writing, the original RTL8156 and the RTL8156B update. We strongly recommend you use USB 3 Gen 1 2.5 GbE USB adapters based on the **B** version.
- For good speed and stability, we advise you to install the Linux or Windows driver from Realtek for the RTL8156B-based NICs if the OS as is driver version does not provide optimal results (issues or inconsistent speed) with the USB adapter. You can find the Realtek RTL8156B driver there:
    - https://www.realtek.com/en/component/zoo/category/network-interface-controllers-10-100-1000m-gigabit-ethernet-usb-3-0-software.

See photo of the connected test machines shown below.



Setup

The inxi output shown below correctly reports the 6 x 2.5 GbE ports on the H2+ with the H2+ Net card, the ports being driven by the Realtek r8125 v9.003.05 driver.

```
odroid@ODROID-H2Plus:~$ sudo inxi -CMmNSD -x
System:    Host: ODROID-H2Plus Kernel: 5.4.0-58-generic x86_64 bits: 64 compiler: gcc v: 9.3.0 Console: tty 0
           Distro: Ubuntu 20.04.1 LTS (Focal Fossa)
Machine:   Type: Desktop Mobo: HARDKERNEL model: ODROID-H2 v: 1.0 serial: N/A UEFI: American Megatrends v: 1.22
           date: 11/13/2020
Memory:    RAM: total: 31.19 GiB used: 765.7 MiB (2.4%)
           Array-1: capacity: 32 GiB slots: 2 EC: None max module size: 16 GiB note: est.
           Device-1: A1_DIMM0 size: 16 GiB speed: 2400 MT/s type: DDR4
           Device-2: A1_DIMM1 size: 16 GiB speed: 2400 MT/s type: DDR4
CPU:       Topology: Quad Core model: Intel Celeron J4115 bits: 64 type: MCP arch: Goldmont Plus rev: 1
           L2 cache: 4096 KiB
           flags: lm nx pae sse sse2 sse3 sse4_1 sse4_2 ssse3 vmx bogomips: 14284
           Speed: 749 MHz min/max: 800/2500 MHz Core speeds (MHz): 1: 866 2: 862 3: 850 4: 788
Network:   Device-1: Realtek RTL8125 2.5GbE driver: r8125 v: 9.003.05-NAPI port: e000 bus ID: 01:00.0
           Device-2: Realtek RTL8125 2.5GbE driver: r8125 v: 9.003.05-NAPI port: d000 bus ID: 02:00.0
           Device-3: Realtek RTL8125 2.5GbE driver: r8125 v: 9.003.05-NAPI port: c000 bus ID: 03:00.0
           Device-4: Realtek RTL8125 2.5GbE driver: r8125 v: 9.003.05-NAPI port: b000 bus ID: 04:00.0
           Device-5: Realtek RTL8125 2.5GbE driver: r8125 v: 9.003.05-NAPI port: a000 bus ID: 05:00.0
           Device-6: Realtek RTL8125 2.5GbE driver: r8125 v: 9.003.05-NAPI port: 9000 bus ID: 06:00.0
Drives:    Local Storage: total: 3.70 TiB used: 79.23 GiB (2.1%)
           ID-1: /dev/mmcblk0 model: CJTD4R size: 58.24 GiB
           ID-2: /dev/sda vendor: Seagate model: ST2000DM006-2DM164 size: 1.82 TiB temp: 30 C
           ID-3: /dev/sdb vendor: Seagate model: ST2000DM006-2DM164 size: 1.82 TiB temp: 29 C
```

The H2+ with the H2 Net card, which is in our testing the Linux/Samba-based NAS, includes:

- an Intel Celeron J4115,
- the Hardkernel Odroid-H2 1.22 BIOS (M.2 slot configured as 4 bifurcated x1 lanes),
- a Ubuntu 20.04.1 + 5.4.0-56 Kernel running on a 64GB eMMC module,
- two Seagate Barracuda 2TB HDD for data storage,
- two Samsung DDR4 2400MHz 16GB in dual channel,
- two built-in 2.5GbE and four additional 2.5GbE from the H2 Net card. Total six 2.5 GbE NICs configured with the Realtek r8125 9.003.05 driver to get the best performance in multi-sessions.

The first Windows 10 client (Odroid-H2+) includes:

- an Intel Celeron J4115,
- the Hardkernel Odroid-H2 1.22 BIOS (M.2 slot configured as 1 x4 lanes, not important for this test),
- Windows 10 20H2 running on a 128GB eMMC module,
- Two Samsung DDR4 2400MHz 16GB RAM
- An RTL8156B-based 2.5 GbE USB 3 Gen 1 adapter (with Realtek driver)

The second Windows 10 client (laptop) includes:

- an Intel Core i7-7500U
- Windows 10 1909 running on an internal 128GB SSD
- Onboard DDR4 4GB RAM + Samsung DDR4 2400MHz 16GB RAM
- An RTL8156B-based 2.5 GbE USB 3 Gen 1 adapter (with Realtek driver)

Static IP

We set static IP addresses on the 2.5 GbE NICs using two different sub-nets on the clients:

- First Windows 10 client USB adapter: 192.168.**3**.100/**24**
- Second Windows 10 client USB adapter: 192.168.**4**.100/**24**

We accordingly set static IP addresses on the server NICs as shown below:

- Server NIC connected to first Windows 10 client: 192.168.**3**.40/**24**
- Server NIC connected to second Windows 10 client: 192.168.**4**.40/**24**

Because we use two different sub-nets with network mask 24, the server Linux kernel routing table is correctly automatically set:

- Anything transmitted between the server and the first Windows 10 client will go through the 192.168.**3**.0/24 sub-net
- Anything transmitted between the server and the second Windows 10 client will go through the 192.168.**4**.0/24 sub-net

A similar automatic routing is performed by the Windows kernel on each client.

In other words, simultaneous communication between the server and the two clients will happen on the respective NICs and provide 2.35 Gbits/sec for each client as we have seen with the iperf3 performance test (see previous section) no matter what the other client is doing.

Samba configuration

On the server side, we edit the Samba configuration as shown below:

```
odroid@ODROID-H2Plus:~$ testparm

...


[H2P-HDD1]

        create mask = 0755

        path = /mnt/hdd1

        read only = No

        valid users = odroid
```

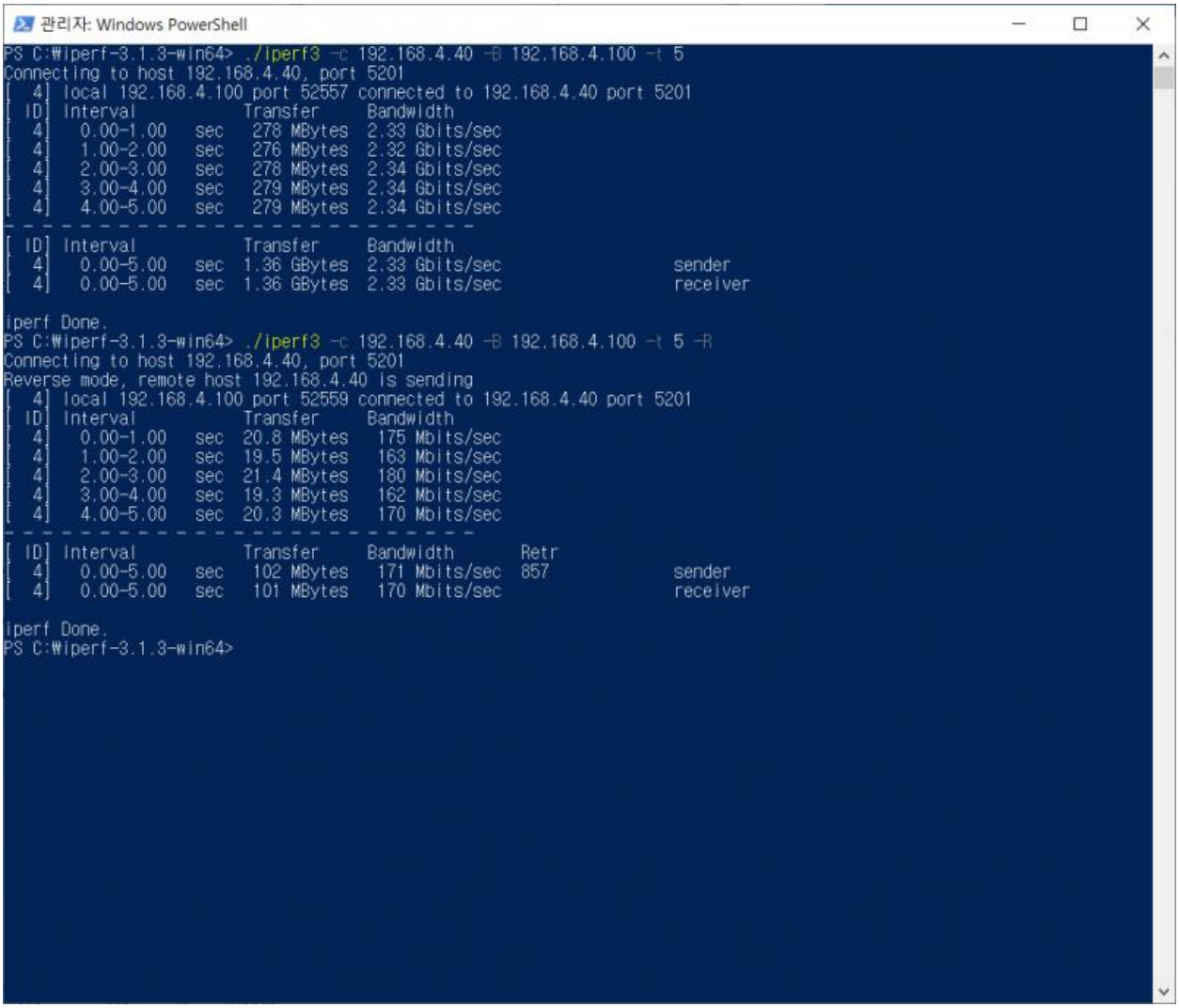
```
[H2P-HDD2]
        create mask = 0755
        path = /mnt/hdd2
        read only = No
        valid users = odroid
```

- On Ubuntu 18.04 or 20.04 the Samba configuration is stored in the file /etc/samba/smb.conf which you can edit with a “sudo vi /etc/samba/smb.conf”, or whatever text only editor you prefer. The location of the smb.conf file may vary depending on the Linux distribution but it is usually inside /etc/samba, at the first level or in a sub-folder. For more details about the Samba configuration file: man smb.conf. For a less terse or tutorial-like documentation, Google or Bing for “smb.conf tutorial”.
- Make sure to restart the smbd and nmbd daemons after you modified the Samba configuration.

```
sudo systemctl restart smbd nmbd
```

Installing the Realtek USB Ethernet driver on Windows 10

Windows 10 includes a driver to handle most of the Realtek network chipset. However this built-in driver may not contain the right code and settings for the new RTL8156B chipset. As a result you may witness inconsistent speed or behavior as it appears in the screenshot shown below where the sending speed using iperf3 is an expected 2.33 Gbits/sec while the receiving speed is a pretty sad 170 Gbits/sec.

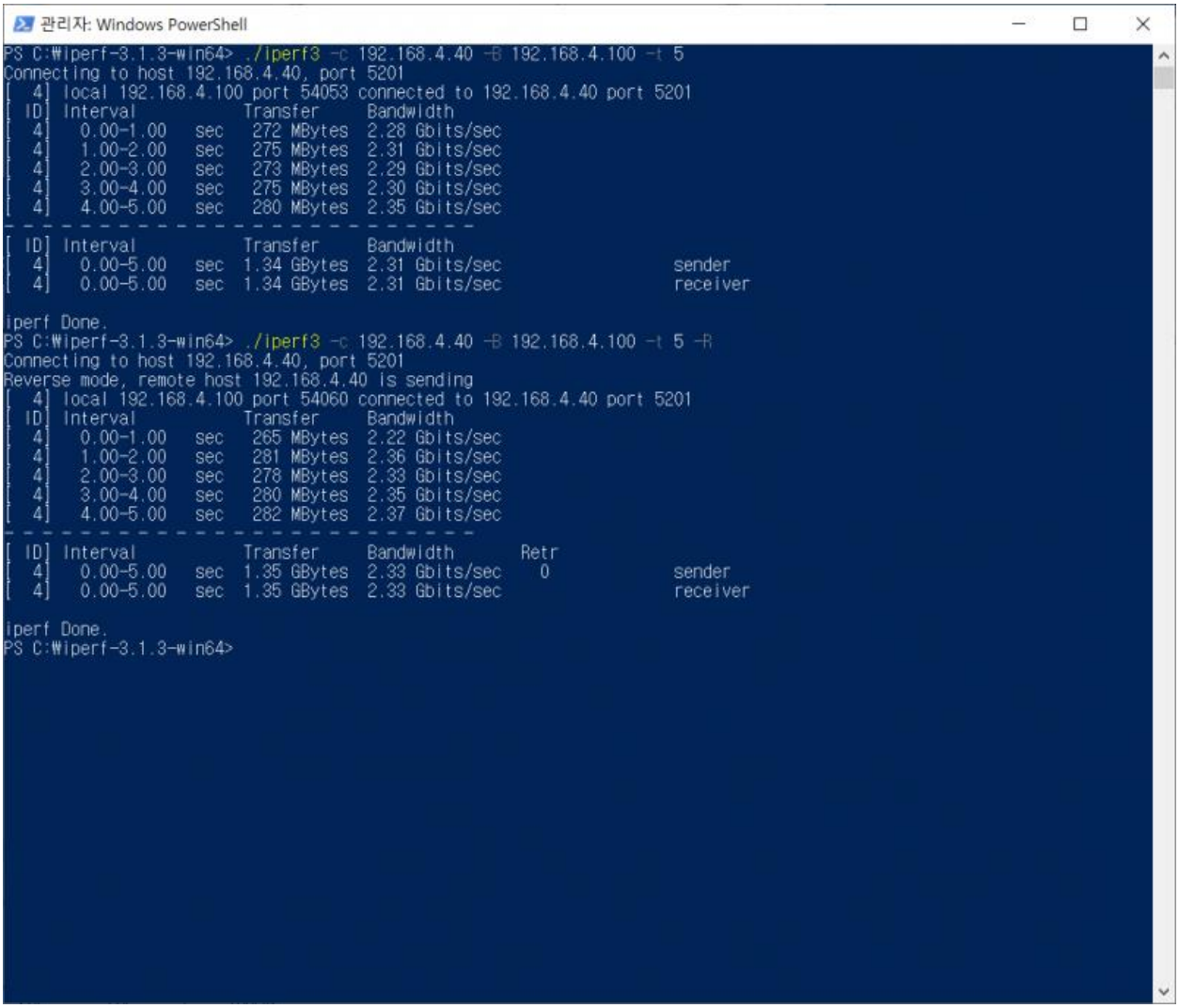


In the iperf3 test, this only shows up to 200 Mbps in download bandwidth.

To fix this issue (as well as others), we downloaded the driver from the Realtek website: https://www.realtek.com/en/component/zoo/category/network-interface-controllers-10-100-1000m-gigabit-ethernet-usb-3-0-software

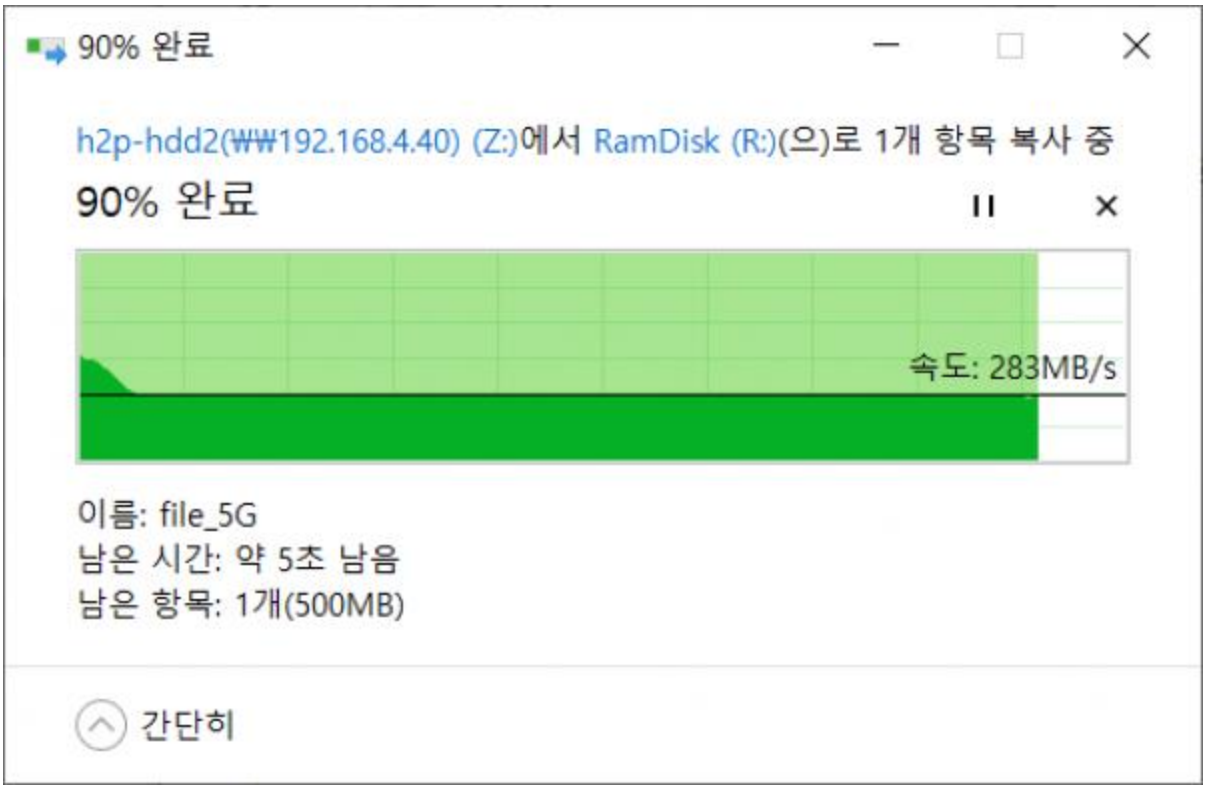
- We used version 10.43.20 (2020.11.25) for this hands-on. Realtek may have published a more recent version since.

Once the driver has been installed, iperf3 shows 2.3x Gbits/sec in both sending and receiving modes, which is what is to be expected.

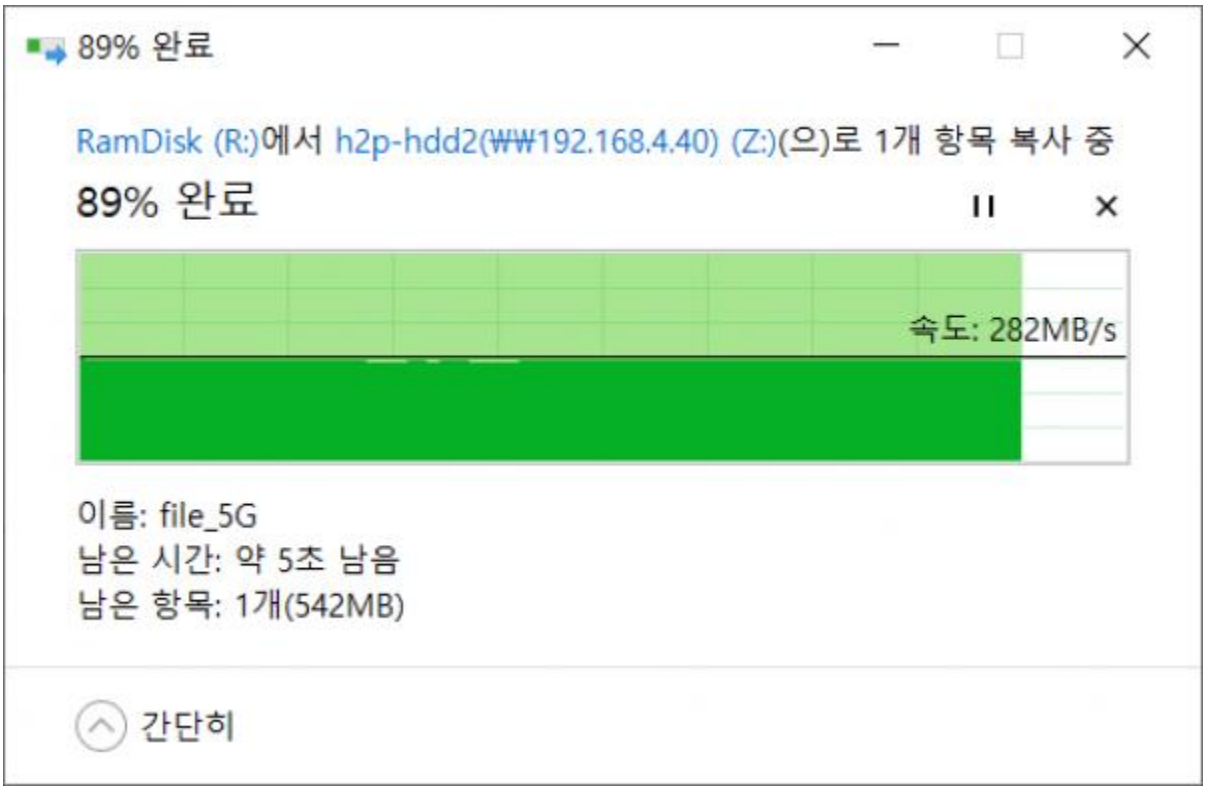


Samba Performance Test - Single Session

Copying a 5 GB file from the NAS to a client

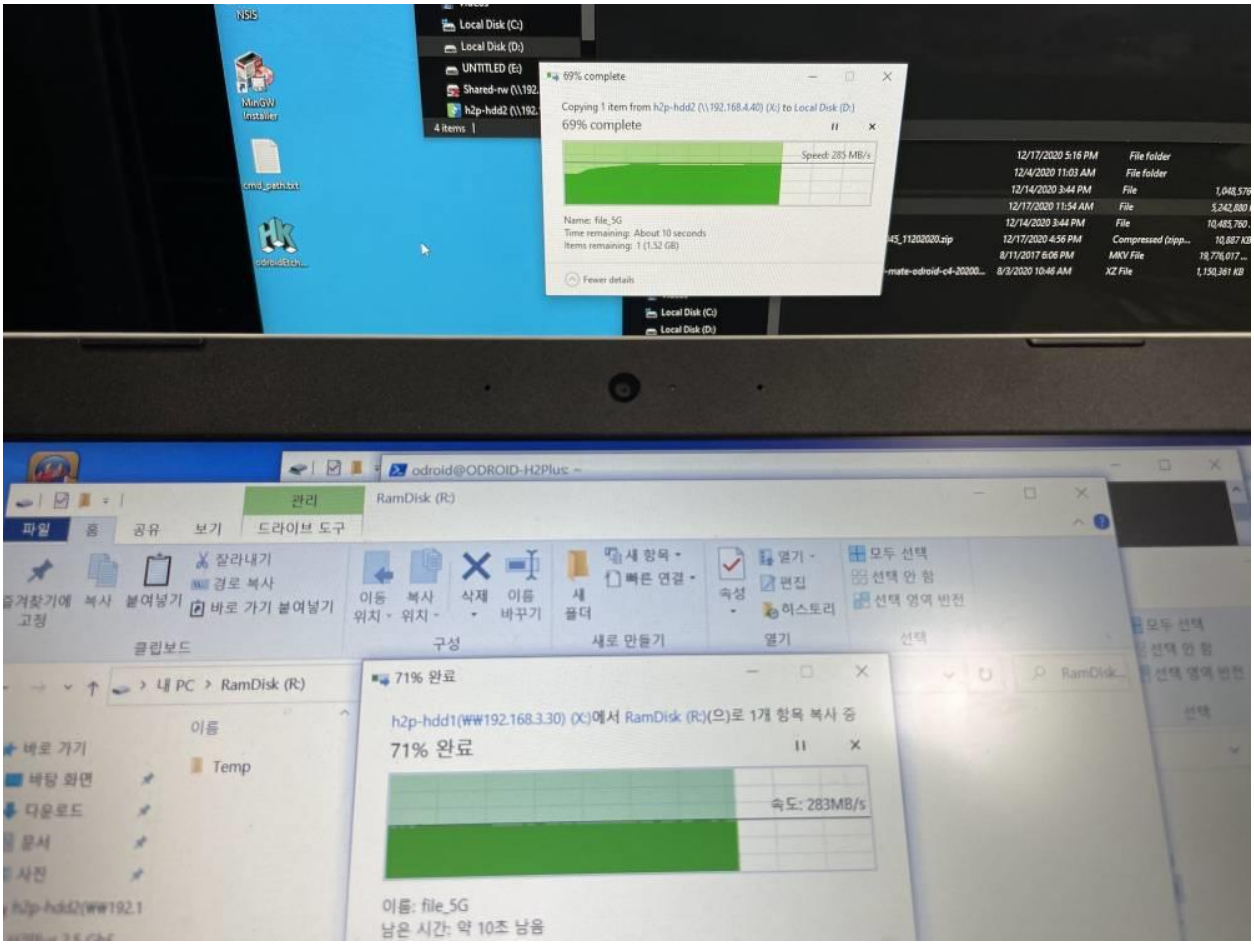


Copying a 5 GB file from a client to the NAS



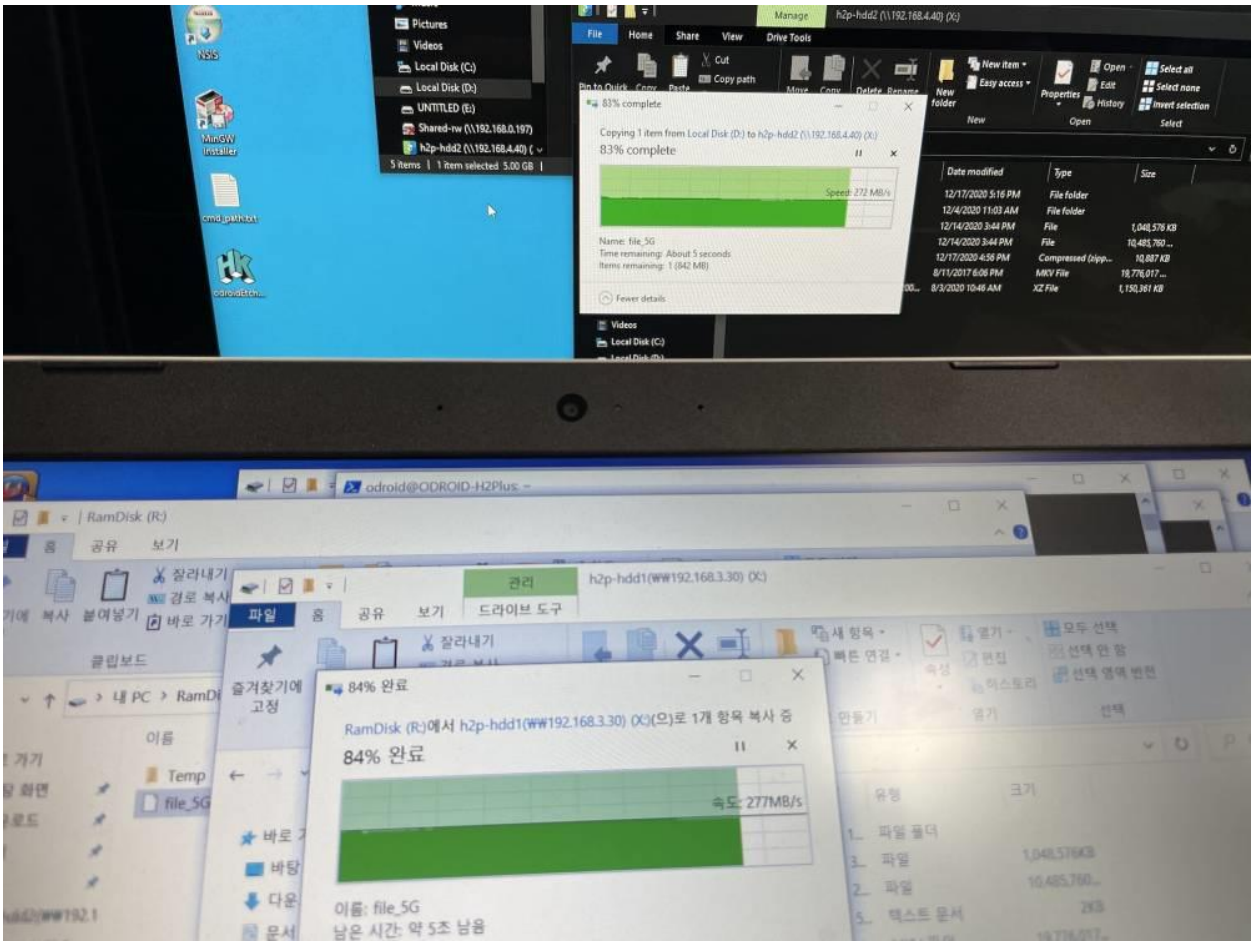
Samba Performance Test – Simultaneous Double Sessions

Copying 5 GB files from the NAS to two clients at the same time



- As we have seen with the iperf3 testing (see previous section) both Samba sessions are running at full speed at the same time: 285 MB/s and 283 MB/s, which means 285 x 8 = 2.28 Gbits/sec and 283 x 8 = 2.26 Gbits/sec which are very closed to the 2.35 Gbits/sec we saw with iperf3, from which you substract the SMB protocol overhead, caching overhead and time spent in disk IO.

Copying 5 GB files from two clients to the NAS at the same time



- Again we see the expected full speed for both sessions (272 and 277 MB/s) minus some overhead.

- One of the clients in this hands-on used a RAM disk in both tests, the other used the internal SSD so that we could minimize the effects of too slow disk IO.
- The speed one can achieve depends on the type of disks involved both on the clients and server sides as well as the nature of the file(s) being copied. Copying a large file will go faster than copying dozens or hundreds of files even though the overall size being transferred ends up to be about the same.
- Know also that the SMB protocol implementations on Windows, Linux and Mac OS will use memory (if available) to cache files recently written or read. The more memory on the server side, the better. The H2/H2+ accepts up to 32 GB of memory.
- While an SSD will support the practical max speed of 285 MB/s as is, you may see similar performance from time to time with large size rotating hard disk whose better speed is combined by the various caching buffers that Samba is using, as well as the caching buffers that a Linux-based NAS uses. A 10TB has more heads and density than a 4TB and therefore a much better throughput during both reading and writing operations.

- For a NAS application that you know will include a lot of file rewriting, make sure you use a CMR disk, not an SMR disk on the server. For details, Google or Bing for “cmr vs. smr”.

Conclusion

The H2 Net card brings significant added value to a H2/H2+-based NAS solution where simultaneous clients can enjoy the full 2.5 GbE speed independently from each other, especially to H2 and H2 rev. B which don’t have onboard 2.5 GbE connectivity in the first place!