# Network Performance Evaluation Between Virtual/Native Nodes Running on ARM-based SBCs Using KVM as Hypervisor

Eric Gamess
Jacksonville State University
Jacksonville, Alabama, USA
egamess@jsu.edu

## ABSTRACT

Single-Board Computers (SBCs) are increasingly being used due to their small form factor, reduced energy consumption, versatility, affordability, and increasing computational power. Therefore, they are now used in projects where they were not initially contemplated, such as running a Virtual Machine Manager (VMM). In this work, two ARM-based SBCs were selected (Raspberry Pi 4 Model B and ODROID-N2+) and an empirical evaluation was carried out to evaluate the network performance between two nodes running in the same SBC, or in different SBCs directly connected through WiFi or Ethernet. Although several hypervisors are suitable for these SBCs, Kernel-based Virtual Machine (KVM) was chosen since it seems to be the most active project that is developed for the ARM-based architecture. The metrics reported in this study include the TCP latency, UDP latency, TCP throughput, and HTTP latency. In general, the network performance of the ODROID-N2+ exceeded the Raspberry Pi 4 Model B. However, the latter has an indisputable advantage over the former with a much larger and more active community, making the development and deployment of applications much faster and straightforward. Hence, selecting the suitable SBCs should be done cautiously, considering the required software and additional hardware that the project is planning to connect to the SBCs.

## CCS CONCEPTS

• **Networks** → **Network performance analysis**; *Network measurement*; • **Computer systems organization** → *Embedded systems.*

## KEYWORDS

Performance Evaluation, Benchmarks, Virtualization, KVM, Single-Board Computer, Raspberry Pi, ODROID

**ACM Reference Format:**
Eric Gamess. 2023. Network Performance Evaluation Between Virtual/Native Nodes Running on ARM-based SBCs Using KVM as Hypervisor. In *2023 ACM Southeast Conference (ACMSE 2023), April 12–14, 2023, Virtual Event, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3564746.3587015


*ACMSE 2023, April 12–14, 2023, Virtual Event, USA*

ACM ISBN 978-1-4503-9921-0/23/04…$15.00
https://doi.org/10.1145/3564746.3587015

## 1 INTRODUCTION

Virtualization [9, 31] enables a single physical machine to act as multiple virtual computers. This allows users to run many independent Virtual Machines (VMs) with their own operating systems and applications, on the same hardware. Virtualization has many advantages such as slashing IT expenses, reducing energy consumption, quicker deployment/redeployment, and enhancing resiliency in disaster recovery. For all these reasons, virtualization is present in almost all the data centers worldwide. Moreover, due to its numerous benefits, virtualization is also gaining acceptance in smaller projects, with much lower budgets, such as home automation. The community has proposed several Virtual Machine Managers (VMMs), also known as hypervisors, such as KVM [11, 25], Xen [10, 20], Proxmox VE [18, 32], and VMware ESXi [15].

On the other hand, Single-Board Computers (SBCs) represent a growing portion of the computing markets [17], as their power increases and they become more affordable. Today, many manufacturers have developed and are marketing SBCs. This paper focuses on ARM-based SBCs, particularly those proposed by the Raspberry Pi Foundation and Hardkernel.

A few years ago, hypervisors were installed in expensive servers, mainly deployed in data centers. With the increasing power of SBCs, VMMs are now deployed on top-of-the-line SBCs, such as the Raspberry Pi 4 Model B [33] and the Hardkernel ODROID-N2+ [21, 22]. This paper studies the network performance between nodes (VM/Native) deployed in the same SBC, or within different SBCs. The work is based on assessments done with popular benchmarking tools and four metrics are reported: (1) TCP latency, (2) UDP latency, (3) TCP throughput, and (4) HTTP latency to retrieve files. At the level of the network technology to connect the nodes, four possibilities were considered: (1) a virtual network, (2) WiFi in the 2.4 GHz band, (3) WiFi in the 5 GHz band, and (4) Ethernet.

In most assessments, the ODROID-N2+ surpassed the Raspberry Pi 4 Model B (RPi 4B). However, regarding the user experience and community support, the RPi 4B significantly excels the ODROID-N2+. For example, the author did not face major issues in booting the RPi 4B from different mediums (e.g., MicroSD cards, thumb drives, SSDs). At the level of the ODROID-N2+, the author had to update Petitboot (the operating system bootloader chosen by Hardkernel), and needed to change some parameters in the command line for booting. Moreover, with the massive community behind the SBCs of the Raspberry Pi Foundation, all the problems faced during this research with the RPi 4B were solved by searching specialized forums. The experience with the ODROID-N2+ was different, and the author did not find solutions to some of the issues by searching the web. Hence, the author had to spend numerous hours to overcome them, which significantly delayed this work.

The rest of the paper is structured as follows. Section 2 discusses a number of peer-reviewed literature works conducted within this research area. The specifications of the SBCs used in this research are given in Section 3. Section 4 introduces the testbeds. The performance tests are presented, done, and discussed in Sections 5, 6, 7, 8, and 9. Finally, Section 10 concludes the paper and discusses future avenues for further research work within the area of study.

## 2 RELATED WORK

In the specialized literature, hypervisors have been assessed but mainly for servers based on Intel processors. For example, Shirinbab, Lundberg, and Ilie [38] made a performance comparison of KVM, VMware ESXi, and Citrix XenServer in two HP ProLiant DL380 G6x86 servers equipped with two Intel Xeon quad-core CPUs with hyper-threading. They reported results such as CPU utilization, disk utilization, and response time during live migrations. In [24], the authors assessed Microsoft Hyper-V, KVM, VMware vSphere, and Xen in a server that had an Intel Xeon 5160 quad-core processor using several benchmarking tools (e.g., `RAMspeed`, `Bonnie++`, `Filebench`, and `Netperf`). Kumar and Singh [27] benchmarked VMware ESXi and Citrix XenServer. They reported parameters such as CPU usage, memory performance, and disk performance. For their experiments, the authors used a server based on an Intel Xeon quad-core processor. Algarni, Ikbal, Alroobaea, Ghiduk, and Nadeem [6] compared the performance of KVM, Citrix XenServer, and Proxmox VE on a Fujitsu Primergy RX2540 M2 server with two Intel Xeon E5-2660 v4 processors. Their study included CPU, memory, cache, and filesystem performance with famous benchmarking tools such as `John the Ripper`, `RAMspeed`, `CacheBench`, and `IOzone`. Djordjevic, Timcenko, Kraljevic, and Macek [12] proposed a mathematical model of the filesystem performance in virtual environments when using type-1 hypervisors. The model was validated based on results obtained from experiments done with VMware ESXi, KVM, Microsoft Hyper-V, and Xen, running on an HPE ProLiant DL180 G6, equipped with Intel Xeon E5520 quad-core processors. The author of this paper found other related works [13, 14, 19, 26, 28–30], but all focused on Intel-based processors.

In the area of performance evaluation of hypervisors running on ARM-based SBCs, very few works have been done so far. The author could find two of them [16, 39], primarily focused on CPU, memory, and filesystem performance. In their study, Toumassian, Werner, and Sikora [39] considered Xen and Jailhouse [3] running on an SBC with an Allwinner A20 SoC, which is based on a dual-core ARM Cortex-A7 CPU and an ARM Mali400 MP2 GPU. They initially ran a heavy-load application on the SBC (native execution) to get a baseline. Then, they executed the same application in VMs created with the hypervisors, so that they could measure the overhead of the virtualization. The authors of [16] assessed the performance of KVM running on two SBCs (RPi 4B and ODROID-N2+). They reported results such as reading and writing throughput in different types of storage mediums, processing power assessment, memory performance, timed compilation of open-source software, and performance of encryption algorithms. Therefore, to the author's best knowledge, it seems that this research work is the first to evaluate the network performance that involves at least one VM running on an ARM-based SBC.

**Table 1: Specification of the RPi 4B and ODROID-N2+**

| | | |
|---|---|---|
| SoC Type | RPi 4B | Broadcom BCM2711 |
| | OD-N2+ | Amlogic S922X |
| Core Type | RPi 4B | Quad-core ARM Cortex-A72 @ 1.8 GHz |
| | OD-N2+ | Quad-core ARM Cortex-A73 @ 2.4 GHz<br>Dual-core ARM Cortex-A53 @ 2.0 GHz |
| RAM | RPi 4B | 1, 2, 4, or 8 GB LPDDR4-3200 SDRAM |
| | OD-N2+ | 2 or 4 GB DDR4 |
| USB Ports | RPi 4B | 2 x USB 2.0 & 2 x USB 3.0 |
| | OD-N2+ | 4 x USB 3.0 & 1 x Micro USB 2.0 OTG |
| HDMI Ports | RPi 4B | 2 x Micro HDMI |
| | OD-N2+ | 1 x Full-size HDMI |
| Data Storage | RPi 4B | MicroSD |
| | OD-N2+ | MicroSD/eMMC |
| Ethernet | RPi 4B | 10/100/1000 Mbps |
| | OD-N2+ | 10/100/1000 Mbps |
| WiFi | RPi 4B | IEEE 802.11b/g/n/ac |
| | OD-N2+ | No |
| Price | RPi 4B | US$35, US$45, US$55, or US$75 |
| | OD-N2+ | US$66 or US$83 |

## 3 SPECIFICATIONS OF THE SINGLE-BOARD COMPUTERS UNDER TEST

SBCs are typically used for industrial applications such as data acquisition, robotics, and process control. They are becoming increasingly popular due to their small form factor, versatility, low energy consumption, decreasing cost, and increasing processing power. Nowadays, many manufacturers are proposing SBCs, including the Raspberry Pi Foundation, Hardkernel, BeagleBoard.org, Nvidia, and UDOO. This study focuses on the SBCs developed by the Raspberry Pi Foundation and Hardkernel (the ODROID family). They were chosen since they seem to be the favorite SBCs used by hobbyists and professionals in the area.

The Raspberry Pi Foundation is currently manufacturing several models of Pis (e.g., Pi Zero, Pi Zero W, Pi Zero 2 W, Pi 3 Model A+, Pi 3 Model B, Pi 3 Model B+, Pi 4 Model B, Pi 400), all of them with ARM-based processors. The Raspberry Pi 400 [34] is a Raspberry Pi 4 Model B [33] built into a compact keyboard; therefore, more suitable to be used as a desktop computer. To run a hypervisor on an SBC, [16] recommended having at least 2 GB of RAM, narrowing down the selection to the RPi 4B, which is the unique model of the Raspberry Pi Foundation that is offered with different sizes of RAM (1, 2, 4, and 8 GB). On the other hand, Hardkernel develops SBCs based on the x86 (e.g., ODROID-H3, ODROID-H3+) and ARM (e.g., ODROID-XU4, ODROID-M1, ODROID-N2L, ODROID-N2+, ODROID-C4) architectures. Several of these ARM-based SBCs were appropriate for this study, but this paper focuses on the ODROID-N2+ [21, 22] since it seems to be one of the best-selling SBCs of Hardkernel [21, 22], and is offered with 2 or 4 GB of RAM. Table 1 shows the specification of the two Devices Under Test (DUTs).

As pointed out in Table 1, the ODROID-N2+ does not have WiFi support. Therefore, the author searched for a USB WiFi adaptor. Several adapters (e.g., Linksys WUSB6300, Netgear A6210) were tested but were not directly supported by the OS. The source code

for compiling the required module is available on the Internet. However, it is generally written for the x86 architecture. Depending on the dongle, it may also be compiled and used for the SBCs of the Raspberry Pi Foundation. None of the source codes consulted by the author mentions the ODROID SBCs, and despite all the efforts of the author, porting it to the ODROID family was unsuccessful. In some cases, the compilation did not work at all. In the cases of a successful compilation, the module did not operate properly for one reason or another. After numerous trials and errors, the ASUS USB-AC53 Nano USB 2.0 WiFi adapter [8] was selected. It is a dual-band WiFi adapter that supports IEEE 802.11b/g/n in the 2.4 GHz band and IEEE 802.11a/n/ac in the 5 GHz band.

In reference to the storage of the SBCs, several alternatives are available. The RPi 4B can have its filesystem installed in a MicroSD card, or in a thumb drive or SSD connected to a USB port. Additionally to the previous three storage mediums, the ODROID-N2+ can also have its filesystem on an eMMC card. As shown in [16], the performance of the SSD is much better, therefore, more suitable for installing a VMM. Also, in general, it offers a larger capacity, allowing the creation of more VMs, with larger virtual disks. For these reasons, a 1 TB SanDisk Ultra 3D SSD (SDSSDH3-1T00-G25) was used as the storage of the SBCs. According to the manufacturer, it can achieve reading and writing speeds up to 560 MB/s and 530 MB/s, respectively. The SSDs were connected to the SBCs through a USB 3.0 port.

The chips of the SBCs may overheat, forcing CPU throttling to decrease the electrical energy consumed and reduce the heat generation. To tackle this problem, Hardkernel sells the ODROID-N2+ [21, 22] with a heat sink in its bottom part. However, the default configuration of the RPi 4B [33] does not have any overheating control mechanism. Hence, the two RPi 4B used in this work were put inside cases that had small fans.

## 4 DESCRIPTION OF THE TESTBEDS

The following equipment was used for the testbeds: two RPi 4B with 8 GB of RAM, two ODROID-N2+ with 4 GB of RAM, and one wireless router. The specifications of the SBCs were given in Section 3. For the wireless router, a NETGEAR AC1750 smart WiFi router R6400v2 was utilized. It had the following characteristics: a 1 GHz Broadcom BCM4708A0 processor with two cores, 128 MB of flash, 256 MB of RAM, WiFi in two radio bands (IEEE 802.11b/g/n in the 2.4 GHz band and IEEE 802.11a/n/ac in the 5 GHz band), four 10/100/1000 Mbps LAN Ethernet ports, and one 10/100/1000 Mbps WAN Ethernet port. In the 2.4 GHz band, the bandwidth was configured to a maximum of 450 Mbps (options for the maximum bandwidth are 54, 217, or 450 Mbps). At the 5 GHz band level, the bandwidth was configured to a maximum of 1300 Mbps (options for the maximum bandwidth are 289, 600, and 1300 Mbps).

For the RPis 4B, the last version of the Raspberry Pi OS [35] (64-bit version) was installed. It was released on September 2022, by the Raspberry Pi Foundation. It is worth mentioning that this OS is available in several versions: (1) Raspberry Pi OS Lite, (2) Raspberry Pi OS with Desktop, and (3) Raspberry Pi OS with Desktop and Recommended Software. The "Lite" option was selected since it is a minimal image without an X window server, which is more suitable for running a hypervisor. On the other hand, for the ODROID family, Hardkernel recommends Ubuntu. After having a hard time with the hypervisor on the ODROID-N2+, the author opted for the CLI version of Armbian that was released on November 2022 [7].

Even if several hypervisors have been developed (e.g., KVM [11, 25], Xen [10, 20], Proxmox VE [18, 32], VMware ESXi [15], etc), they are primarily focused on the x86 technologies (e.g., Intel and AMD). For the ARM-based architecture, some efforts are on the way for Proxmox VE (with Pimox [4]) and VMware ESXi, but KVM seems to be the project that has put much effort into porting its hypervisor to this emerging architecture. KVM [11, 25] (Kernel-based Virtual Machine) is a full open-source virtualization solution, consisting of some kernel modules, libraries (e.g., `libvirt`), and user-space utilities to create and manage virtual machines (e.g., `virsh`, `virt-manager`). With KVM, users can create different types of virtual networks (e.g., Isolated, Routed, NAT). To do so, KVM introduces the notion of "virtual switch". It is a software switch where the VMs will be plugged in. Each time that a virtual switch is created, a virtual interface will be built at the level of the virtualization server with a connection to the virtual switch. With all the possible types of networks, the VMs connected to the same virtual switch can communicate with each other, and with the virtualization host. Below is some specific information on the frequently used types of networks:

- **Isolated mode:** The packets generated by the VMs connected to the virtual switch will not pass outside of the virtualization host, nor can they receive traffic from outside the virtualization host.
- **Routed mode:** In this mode, the virtualization host acts as a router between the VMs connected to the virtual switch and the external world. The VMs may send traffic outside the virtualization host and external traffic may reach the VMs, if additional routing entries are appropriately configured.
- **NAT mode:** In this case, KVM provides a DHCP and a NAT servers (through `dnsmasq` [37]). The idea is that the VMs connected to the virtual switch should get their network configuration through the DHCP server, and communicate with external devices through the NAT server. The VMs should use the VMM physical machine IP address for communication with external networks. By default, without additional configuration, the communication between a VM and an external device shall be initiated by the VM.

The testbed of Figure 1 consisted of two SBCs of the same type (two RPi 4B or two ODROID-N2+) connected through the NETGEAR WiFi router. Using KVM, two VMs were created in each SBC (VM1 and VM2 in the first SBC, and VM3 and VM4 in the second SBC). The virtual switches were configured in "routed mode", so network traffic could flow seamlessly from any VM to any VM (e.g., without a translation due to NAT). `VM1`, `VM2`, and the `virbr1` interface of the first SBC were assigned IP addresses in the network `10.0.1.0/24`. Similarly, `VM3`, `VM4`, and the `virbr1` interface of the second SBC were assigned IP addresses in the network `10.0.2.0/24`. The WiFi interfaces (`wlan0`) of both SBCs were configured with IP addresses of the network `192.168.1.0/24`.

The four VMs had the same characteristics as specified next: 2 vCPUs, 1024 MB of RAM, and a virtual disk of 5 GB. In each VM, a minimal version of Debian Buster (Debian 10.13) was installed. That

is, to keep them light, the VMs had a basic operating system and no X Window capabilities. The only service that was installed and run in the VMs was an SSH server, so that they could be accessed remotely.

To assess the network performance, qperf [5] was selected. qperf is an open-source benchmarking tool to measure network parameters between two nodes. It can work over TCP/IP as well as the RDMA transports. qperf offers several tests such as `tcp_lat` (TCP latency), `udp_lat` (UDP latency), `tcp_bw` (TCP bandwidth/throughput), and `udp_bw` (UDP bandwidth/throughput). They are all based on the client/server model, so one instance of qperf is run in a node as a server while the other is run in another node as a client. For the TCP latency test (`tcp_lat`), a TCP message of a specific length is sent back and forth between the client and the server. Then, the total time for the full exchange is divided by the number of round trips, and 2, to get the latency (also known as own-way delay). For the TCP bandwidth/throughput test (`tcp_bw`), the client floods the server by sending as many TCP messages of a specific length, as possible. At the end of the experiment, the average TCP throughput received by the server is reported. The UDP tests (`udp_lat` and `udp_bw`) are similar to their TCP counterparts.

Figure 2 depicts the steps that the author followed to install the last version of qperf in the SBCs and the VMs. The line numbers were added to ease referencing the lines. Line 01 cloned the source code of the last version of qperf from GitHub. Lines 03-05 prepared/configured the code for compilation. The compilation and installation were done with Line 06 and Line 07, respectively. Line 08 shows how to run an instance of the server. Lines 09-12 display the commands used to run the client for the `tcp_lat`, `udp_lat`, `tcp_bw`, and `udp_bw` assessments, respectively. Option `-t` specified the duration of the experiment in seconds, and for all the experiments of this work, it was set to 50 seconds. Option `-m` was used to set the TCP/UDP message payload length. Four values were chosen: 1024, 4096, 8192, and 16384 bytes for all the experiments.

Each experiment in this paper was executed several times, and the reported results are an average of the repeated experimental runs. By repeating and averaging, the consistency of the empirical findings is ensured.

## 5 TCP/UDP ASSESSMENT WITHIN THE SAME HYPERVISOR

In this section, TCP and UDP were evaluated between two nodes (VM/native) running in the same SBC (i.e., connected through the same virtual switch). Figure 3 shows the TCP latency. The figure has four groups of six bars. The groups correspond to a TCP payload of 1024, 4096, 8192, and 16384 bytes, respectively. For each group, the six bars are: (1) VM-Native for RPi 4B, (2) Native-VM for RPi 4B, (3) VM-VM for RPi 4B, (4) VM-Native for ODROID-N2+, (5) Native-VM for ODROID-N2+, and (6) VM-VM for ODROID-N2+. The label "VM-Native" indicates that the client of qperf was run in a VM, while the server of qperf was run natively in the SBC (e.g., the client in VM1 and the server natively in the first SBC). The label "Native-VM" means the opposite situation (e.g., the client natively in the first SBC and the server in VM1). The label "VM-VM" specifies that both the client and server were run in VMs (e.g., the client in VM1 and the server in VM2). Figure 3 indicates that the latency increases with a growing TCP payload, due to the higher number of bytes to be transmitted. The results for "VM-Native" and "Native-VM" are almost identical, which is logical since the total time required to bounce back and forth a TCP message several times will not change if the exchange is started by a VM or natively. The latency of the ODROID-N2+ is better than the one of the RPi 4B.

Figure 4 represents the UDP latency for a UDP payload of 1024, 4096, 8192, and 16384 bytes, respectively. Similarly to Figure 3, it shows that the latency increases with the growing UDP payload, and that the results for "VM-Native" and "Native-VM" are almost identical for the same previously mentioned reasons. The UDP latency (see Figure 4) is slightly inferior to the TCP latency (see Figure 3), and confirms that the connectionless UDP protocol is lighter than the connection-oriented TCP protocol.

Figure 5 illustrates the maximum TCP throughput for a TCP payload of 1024, 4096, 8192, and 16384 bytes, respectively. As can be seen, the throughput increases with the growing TCP payload. The ODROID-N2+ outperformed the RPi 4B. The results are high (a little more than 8 Gbps for the peak performance), since the communication between the two nodes is done virtually, mostly through some memory copies of the TCP message. In general, the performance of the RPi 4B was better for `VM-Native`, while the ODROID-N2+ has a greater performance for `Native-VM`.

The author also tried to get results for the UDP throughput. However, they were not reported in this work since they significantly varied from test-to-test, and would be questionable. The difference between UDP and TCP is that TCP has a congestion control mechanism (congestion window). That is, in TCP, the sender can not flood the receiver with an overwhelming number of segments. However, it is the case in UDP which is a connectionless protocol, and where the receiver does not give any feedback to the sender, eventually leading to many losses. This unstable behavior was constant in all the UDP throughput experiments (e.g., not only the ones in this section), so the author decided not to include them at all in this paper.

## 6 TCP/UDP ASSESSMENT WHEN USING WIFI IN THE 2.4 GHZ BAND

In this section, TCP and UDP were evaluated between two nodes (VM/Native) running in different SBCs. As shown in Figure 1, the network connection was made through the NETGEAR R6400v2 WiFi router, using the 2.4 GHz band. It is worth remembering that the ODROID-N2+ does not have a WiFi adapter, and an ASUS USB-AC53 Nano USB 2.0 WiFi adapter [8] was connected to one of its USB 3.0 ports. The label "VM-Native" indicates that the client of qperf was run in a VM in the first SBC, while the server of qperf was run natively in the second SBC (e.g., the client in VM1 and the server natively in the second SBC). The label "Native-VM" means the opposite situation (e.g., the client natively in the first SBC and the server in VM3). The label "VM-VM" specifies that both the client and server were run in VMs created in different SBCs (e.g., the client in VM1 and the server in VM3).

Figures 6 and 7 represent the TCP and UDP latencies, respectively. They indicate that the latency increases with a growing payload. Withing a manufacturer, the results are very similar for

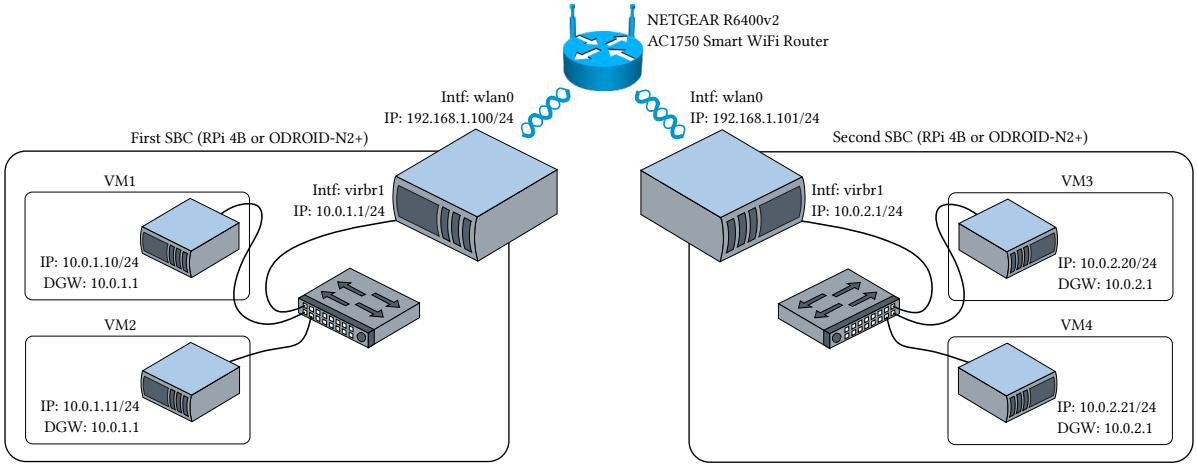


**Figure 1: Testbed for the Experiments**

```
01: git clone https://github.com/linux-rdma/qperf.git
02: cd qperf
03: ./cleanup
04: ./autogen.sh
05: ./configure
06: make
07: make install
```

```
08: qperf
```

```
09: qperf <IPAddressServer> -vv -t <durationInSec> -m <msgSize> tcp_lat
```

```
10: qperf <IPAddressServer> -vv -t <durationInSec> -m <msgSize> udp_lat
```

```
11: qperf <IPAddressServer> -vv -t <durationInSec> -m <msgSize> -ub tcp_bw
```

```
12: qperf <IPAddressServer> -vv -t <durationInSec> -m <msgSize> -ub udp_bw
```

**Figure 2: Compilation and Usage of qperf**

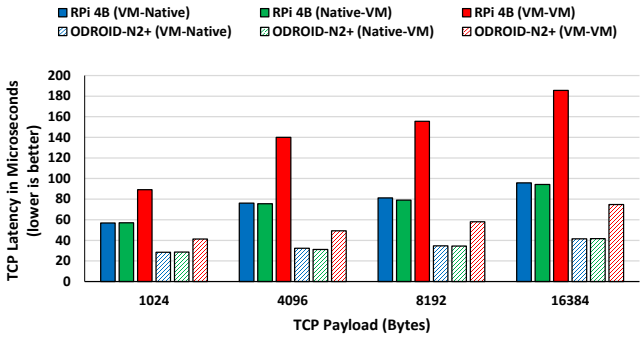


**Figure 3: TCP Latency Between Nodes Connected to the Same Virtual Switch**

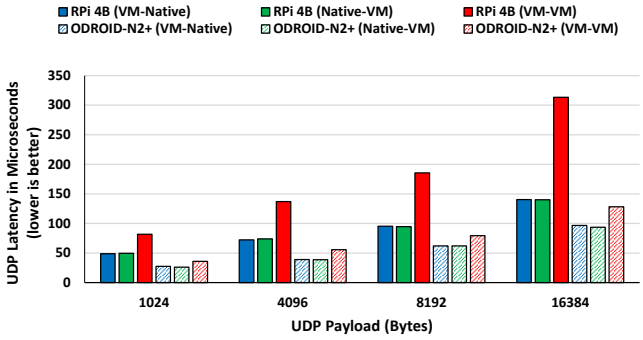


**Figure 4: UDP Latency Between Nodes Connected to the Same Virtual Switch**

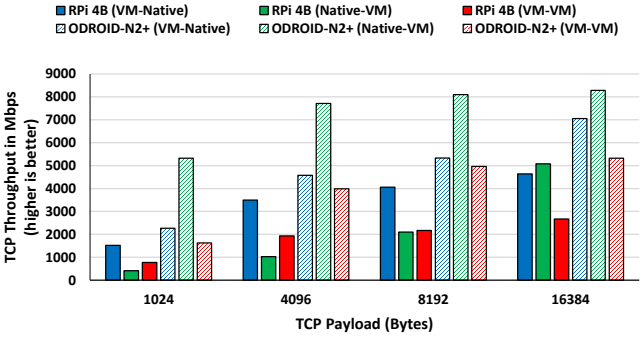


**Figure 5: TCP Throughput Between Nodes Connected to the Same Virtual Switch**

the three cases ("VM-Native", "Native-VM", and "VM-VM"). This can be explained by the physical network (WiFi in the 2.4 GHz band), which significantly restricts the speed of the exchange, limiting the possibility of noticing the differences between virtual and native executions. Similarly to the experiments of Section 5, the UDP latency is slightly lower than the TCP latency. The RPi 4B did a little better for a payload of 1024 bytes, while the ODROID-N2+ has lower latencies for the other payloads (i.e., 4096, 8192, and 16384 bytes).

Figure 8 depicts the maximum TCP throughput for a TCP payload of 1024, 4096, 8192, and 16384 bytes, respectively. It marginally increases with the growing TCP payload. The RPi 4B did slightly better for the `VM-Native` and `Native-VM` experiments. For the `Native-VM` assessment, the ODROID-N2+ did much better.

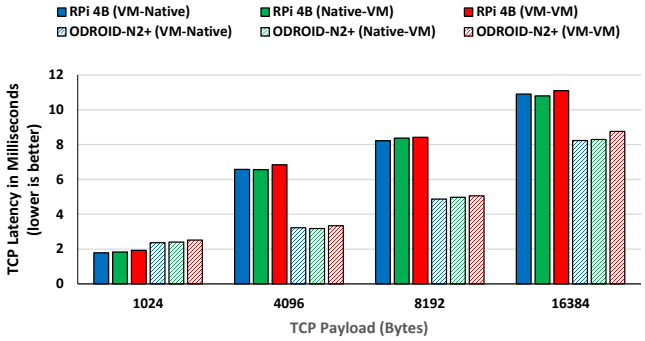


**Figure 6: TCP Latency Between Nodes Running in Different SBCs when Using WiFi in the 2.4 GHz Band**

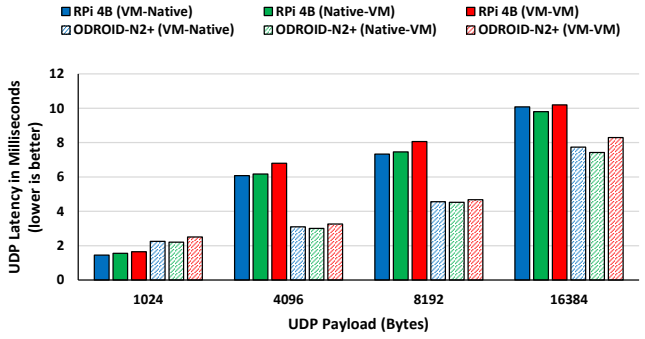


**Figure 7: UDP Latency Between Nodes Running in Different SBCs when Using WiFi in the 2.4 GHz Band**

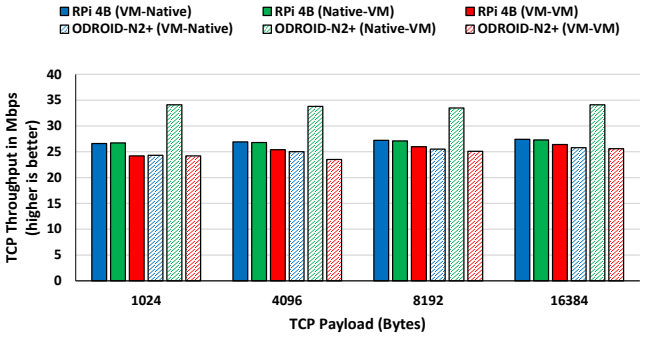


**Figure 8: TCP Throughput Between Nodes Running in Different SBCs when Using WiFi in the 2.4 GHz Band**

## 7 TCP/UDP ASSESSMENT WHEN USING WIFI IN THE 5 GHZ BAND

In this section, TCP and UDP were evaluated between two nodes (VM/Native) running in different SBCs. As shown in Figure 1, the network connection was made through the NETGEAR R6400v2 WiFi router, when using the 5 GHz band.

Figures 9 and 10 show the TCP and UDP latencies, respectively. Except for the much lower values, in general, the behavior of the latency in the 5 GHz band is similar to the one of the 2.4 GHz band (see Section 6).

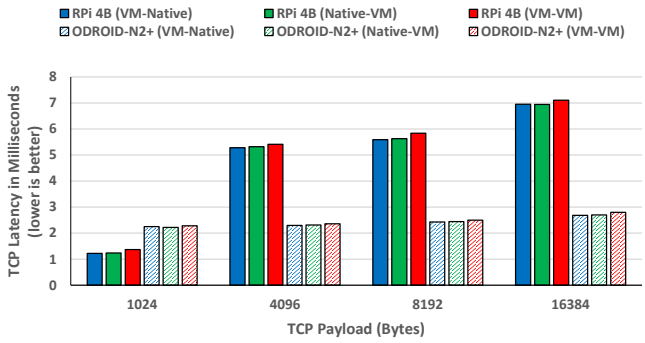


**Figure 9: TCP Latency Between Nodes Running in Different SBCs when Using WiFi in the 5 GHz Band**

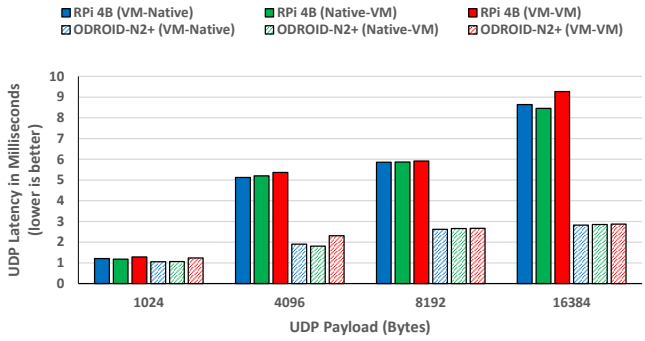


**Figure 10: UDP Latency Between Nodes Running in Different SBCs when Using WiFi in the 5 GHz Band**

Figure 11 depicts the maximum TCP throughput for a TCP payload of 1024, 4096, 8192, and 16384 bytes, respectively. It slightly increases with the growing payload. The ODROID-N2+ outperformed the RPi 4B. As expected, going from the 2.4 GHz band (see Figure 8) to the 5 GHz band (see Figure 11) did improve the results. The maximum TCP throughput for the RPi 4B went from an approximate value of 25 Mbps to an approximate value of 80 Mbps. In the case of the ODROID-N2+, it went from an approximate value of 25/35 Mbps to an approximate value of 100 Mbps.

## 8 TCP/UDP ASSESSMENT WHEN USING ETHERNET

In this section, TCP and UDP were evaluated between two nodes (VM/Native) running in different SBCs, this time, connected through Ethernet. That is, rather than using WiFi as in Figure 1, the two SBCs were connected to the LAN ports of the WiFi router using UTP cables. WiFi (the `wlan0` interface) was disabled at the level of the two SBCs.

Figures 12 and 13 illustrate the TCP and UDP latencies, respectively. They suggest that the latency increase with a growing payload. As expected, they are much lower than the latencies of the

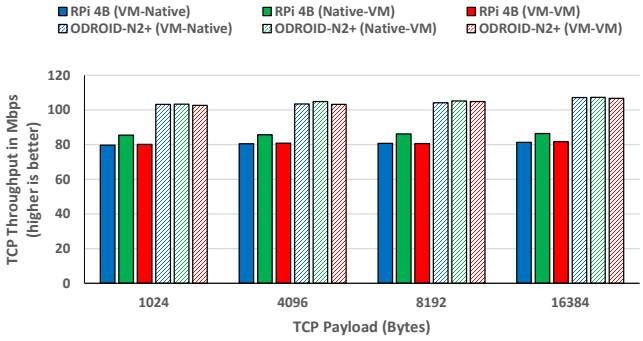


**Figure 11: TCP Throughput Between Nodes Running in Different SBCs when Using WiFi in the 5 GHz Band**

WiFi experiments (see Sections 6 and 7), but higher than the latencies obtained within the same SBC (see Section 5). Similarly to the previous experiments, the UDP latency is slightly lower than the TCP latency. However, this time, the RPi 4B outperformed the ODROID-N2+.

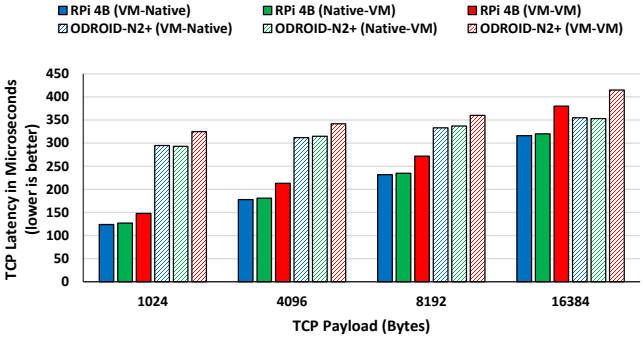


**Figure 12: TCP Latency Between Nodes Running in Different SBCs when Using Ethernet**

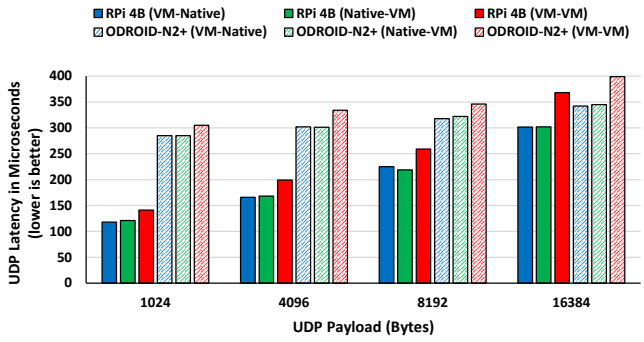


**Figure 13: UDP Latency Between Nodes Running in Different SBCs when Using Ethernet**

Figure 14 depicts the maximum TCP throughput for a TCP payload of 1024, 4096, 8192, and 16384 bytes, respectively. It increases with the growing payload, which is more noticeable for the RPi 4B. The ODROID-N2+ outperformed the RPi 4B. As expected, going from WiFi (see Figure 11) to Ethernet (see Figure 14) did improve the results. For example, the maximum TCP throughput for the ODROID-N2+ went from an approximate value of 100 Mbps for WiFi in the 5 GHz band to an approximate value of 940 Mbps for Ethernet.

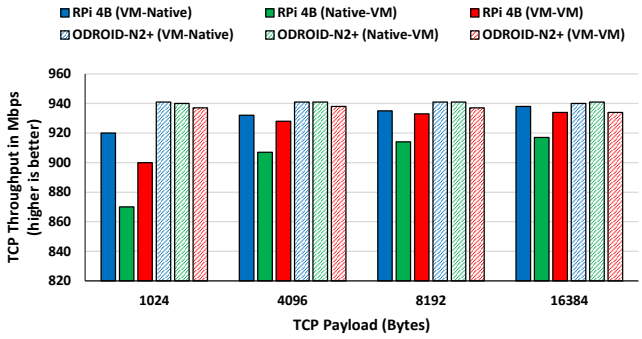


**Figure 14: TCP Throughput Between Nodes Running in Different SBCs when Using Ethernet**

## 9 HTTP ASSESSMENTS

The HTTP protocol is one of the most successful protocols of the application layer. It is the core of the World Wide Web (WWW). In this section, the transfer of files between a web server (Apache) and a web client (Bombardier) was evaluated. Bombardier [1] is a cross-platform HTTP(S) benchmarking tool. It is written in the Go programming language and uses `fasthttp` [2] instead of the Go's default HTTP library, because of its reliability and lightning-fast performance.

Figure 15 shows the steps followed to install and use the last version of Apache [23, 36] and Bombardier [1] in the SBCs and the VMs. Line 01 fetched a fresh list of available packages from the repositories. Line 02 installed some dependencies to compile Apache. The last versions of the source code of Apache, APR, and APR-Util were obtained, uncompressed, and unpacked in Lines 03-08. Line 12 was used for the configuration, Line 13 did the compilation, while Line 14 installed Apache. The default configuration of the web server was updated in Line 17. Lines 18 and 19 were executed to start and stop Apache, respectively. In Line 21, the last version of Bombardier for the ARM-based architecture was downloaded. The execution of the benchmarking tool is shown in Line 23, where options `-d` and `-c` specify the duration of the test and the maximum number of concurrent connections. For all the results reported in this paper, the duration was set to 50 seconds, while up to 100 concurrent connections were allowed. At the level of the HTML file to be transferred from the server to the client, four files were provided with different sizes (1024, 4096, 8192, and 16384 bytes). It is worth mentioning that when executed, Bombardier sends concurrent requests to the web server, for the duration of the experiment. At the end of the experiment, it reports several metrics: (1) the average HTTP throughput, (2) the average number of requests per second, and (3) the average latency for a request. The latency is defined as the total time elapsed between sending a request and

receiving all the bytes of the requested HTML file. All these metrics are correlated, and the author opted to report the latency.

```
01: apt-get update
02: apt-get install libpcre3-dev libexpat1-dev
03: wget https://dlcdn.apache.org/httpd/httpd-2.4.54.tar.bz2
04: wget http://archive.apache.org/dist/apr/apr-1.7.0.tar.bz2
05: wget http://archive.apache.org/dist/apr/apr-util-1.6.1.tar.bz2
06: tar -jxvf httpd-2.4.54.tar.bz2
07: tar -jxvf apr-1.7.0.tar.bz2
08: tar -jxvf apr-util-1.6.1.tar.bz2
09: mv apr-1.7.0 httpd-2.4.54/srclib/apr
10: mv apr-util-1.6.1 httpd-2.4.54/srclib/apr-util
11: cd httpd-2.4.54
12: ./configure --with-included-apr --prefix=${HOME}/httpd
13: make
14: make install
15: cd ..
16: rm -rf httpd-2.4.54
17: vi ${HOME}/httpd/conf/httpd.conf
18: ${HOME}/httpd/bin/apachectl -k start -f ${HOME}/httpd/conf/httpd.conf
19: ${HOME}/httpd/bin/apachectl -k stop
20: rm -f ${HOME}/httpd/logs/*
21: wget https://github.com/codesenberg/bombardier/releases/download/
        v1.2.5/bombardier-linux-arm64
22: chmod a+x bombardier-linux-arm64
23: ./bombardier-linux-arm64 -d <duration> -c <maxConnections> <url>
```

**Figure 15: Compilation and Usage of Apache/Bombardier**

Figure 16 depicts the results obtained between two nodes (VM/native) running in the same SBC (i.e., connected through the same virtual switch). The figure has four groups of six bars. The groups correspond to the transfer of a file of 1024, 4096, 8192, and 16384 bytes, respectively. The label "VM-Native" indicates that Bombardier was run in a VM, while Apache was run natively in the SBC (e.g., Bombardier in VM1 and Apache natively in the first SBC). The label "Native-VM" means the opposite situation (e.g., Bombardier natively in the first SBC and Apache in VM1). The label "VM-VM" specifies that both Bombardier and Apache were run in VMs (e.g., Bombardier in VM1 and Apache in VM2). The figure indicates that the "VM-VM" experiments were slower, especially for the RPi 4B, and that the latency increased with the file size. The ODROID-N2+ outperformed the RPi 4B.

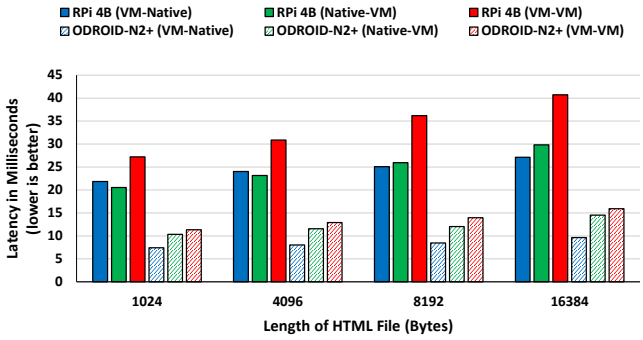


**Figure 16: Latency to Fetch HTML Files Between Nodes Connected to the Same Virtual Switch**

Figure 17 shows the results obtained between two nodes (VM/Native) running in different SBCs, when using WiFi in the 2.4 GHz band. As can be seen, the results are very similar in the three cases (VM-Native, Native-VM, and VM-VM), and between the two architectures (RPi 4B and ODROID-N2+). The results are much higher than the ones in Figure 16, since there are real transmissions through the WiFi network, compared to the previous virtual transmission.

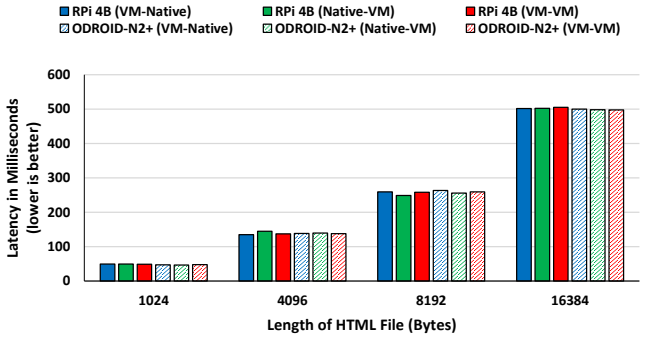


**Figure 17: Latency to Fetch HTML Files Between Nodes Running in Different SBCs when Using WiFi in the 2.4 GHz Band**

Figure 18 represents the results obtained between two nodes (VM/Native) running in different SBCs, when using WiFi in the 5 GHz band. There is not much difference between the three cases (VM-Native, Native-VM, and VM-VM). As expected, going from the 2.4 GHz band (see Figure 17) to the 5 GHz band (see Figure 18) of WiFi did improve the latency. The ODROID-N2+ outperformed the RPi 4B.

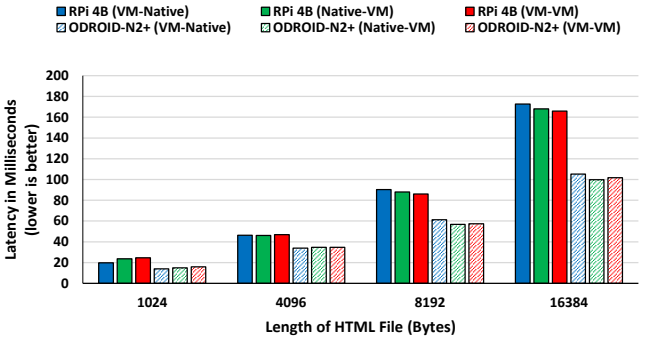


**Figure 18: Latency to Fetch HTML Files Between Nodes Running in Different SBCs when Using WiFi in the 5 GHz Band**

Figure 19 illustrates the results obtained between two nodes (VM/Native) running in different SBCs, when using Ethernet. As can be seen, the latency grows with the increasing size of the HTML file transferred. As expected, going from WiFi in the 5 GHz band (see Figure 18) to Ethernet (see Figure 19) did improve the latency. In this case also, the ODROID-N2+ did better than the RPi 4B.

## 10 CONCLUSIONS AND FUTURE WORK

In this work, the network performance between nodes (VM/Native) that run in the same SBC, or in different SBCs was analyzed. By nodes, the author wants to specify two possibilities: (1) the node

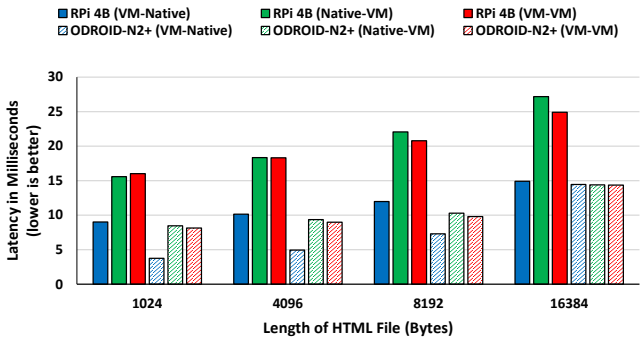


**Figure 19: Latency to Fetch HTML Files Between Nodes Running in Different SBCs when Using Ethernet**

can be a VM and (2) the execution is done natively in the SBC. KVM was selected as the VMM, because it seems to be the project that has made more efforts to port its hypervisor to the emerging ARM-based architecture.

This work is intended to guide users in the selection of an SBC. In general, the ODROID-N2+ outperformed the RPi 4B at the level of the network performance. However, it is worth remembering that the RPi 4B is a cheaper solution, and is also distributed with WiFi natively. In the case of the ODROID-N2+, a USB WiFi adapter should be purchased if the project is aimed at using wireless communication. Moreover, the author faced the experience of the community support. The SBCs of the Raspberry Pi Foundation have a large community, and when in need of support, most of the issues can be solved by searching specialized forums. On the other hand, the ODROID family has less success at the commercial level, resulting in a much smaller community behind it. Hence, a significant amount of efforts should be made into solving issues. For example, the author tried several USB WiFi adapters before finding one that finally worked for the selected OS (Armbian). Hence, choosing between one or the other SBC to host a VMM cannot only be based on the needed processing power, budget constraint, WiFi requirement, and network performance. Two other deciding factors should also be considered: (1) the software to be installed in the SBC and (2) the additional hardware that the project is planning to connect to the SBC.

As future research avenues, the author plans to assess the network performance of virtualization vs. container technologies (Docker, LXC, Podman, rkt, OpenVZ, etc), and to incorporate additional SBCs in the study. Another line of research is to benchmark the IPv4 and IPv6 network performance, when using virtualization and container technologies.

## ACKNOWLEDGMENTS

We are grateful to "Faculty Commons" and the "College of Science & Mathematics" at Jacksonville State University for partially funding this project.

## REFERENCES

[1] [n. d.]. *Bombardier: A HTTP(S) Benchmarking Tool.* https://github.com/codesenberg/bombardier
[2] [n. d.]. *Fastest and Reliable HTTP Implementation in Go.* https://github.com/valyala/fasthttp
[3] [n. d.]. *Jailhouse - Linux-based Partitioning Hypervisor.* https://github.com/siemens/jailhouse
[4] [n. d.]. *Pimox - Proxmox v7 for the Raspberry Pi.* https://github.com/pimox/pimox7
[5] [n. d.]. *qperf.* https://github.com/linux-rdma/qperf
[6] Sultan Abdullah Algarni, Mohammad Rafi Ikbal, Roobaea Alroobaea, Ahmed S Ghiduk, and Farrukh Nadeem. 2018. Performance Evaluation of Xen, KVM, and Proxmox Hypervisors. *International Journal of Open Source Software and Processes* 9, 2 (June 2018), 39–54.
[7] Armbian. [n. d.]. *ODROID-N2/ODROID-N2+.* https://www.armbian.com/odroid-n2
[8] ASUS. [n. d.]. *USB-AC53 Nano: AC1200 Dual-band USB Wi-Fi Adapter.* https://www.asus.com/us/networking-iot-servers/adapters/all-series/usb-ac53-nano
[9] Edouard Bugnion, Jason Nieh, and Dan Tsafrir. 2017. *Hardware and Software Support for Virtualization* (1st. ed.). Springer. Synthesis Lectures on Computer Architecture.
[10] Prabhakar Chaganti. 2022. *Xen Virtualization: A Fast and Practical Guide to Supporting Multiple Operating Systems with the Xen Hypervisor.* Packt Publishing.
[11] Humble Devassy Chirammal, Prasad Mukhedkar, and Anil Vettathu. 2016. *Mastering KVM Virtualization* (2nd. ed.). Packt Publishing.
[12] Borislav Djordjevic, Valentina Timcenko, Nenad Kraljevic, and Nemanja Macek. 2021. File System Performance Comparison in Full Hardware Virtualization with ESXi, KVM, Hyper-V and Xen Hypervisors. *Advances in Electrical and Computer Engineering* 21, 1 (Sept. 2021), 11–20. https://doi.org/10.4316/AECE.2021.01002
[13] Abdellatief Elsayed and Nashwa Abdelbaki. 2013. Performance Evaluation and Comparison of the Top Market Virtualization Hypervisors. In *2013 8th International Conference on Computer Engineering & Systems (ICCES 2013).* Cairo, Egypt.
[14] Hasan Fayyad-Kazan, Luc Perneel, and Martin Timmerman. 2013. Benchmarking the Performance of Microsoft Hyper-V Server, VMware ESXi and Xen Hypervisors. *Journal of Emerging Trends in Computing and Information Sciences* 4, 12 (Dec. 2013), 922–933.
[15] Thomas Fenton and Patrick Kennedy. 2021. *Running ESXi on a Raspberry Pi: Installing VMware ESXi on Raspberry Pi 4 to Run Linux Virtual Machines.* Apress.
[16] Eric Gamess, Mausam Parajuli, and Syed Shah. 2023. Performance Evaluation of ARM-Based Single-Board Computers as Hypervisors Using Kernel-Based Virtual Machines. *International Journal of Computer Networks & Communications (IJCNC)* (2023). In Press.
[17] Global Market Insights. 2022. *Single-Board Computer Market.* https://www.gminsights.com/industry-analysis/single-board-computer-sbc-market
[18] Rik Goldman. 2016. *Learning Proxmox VE.* Packt Publishing.
[19] Waldemar Graniszewski and Adam Arciszewski. 2016. Performance Analysis of Selected Hypervisors (Virtual Machine Monitors - VMMs). *International Journal of Electronics and Telecommunications* 62, 3 (Sept. 2016), 231–236. https://doi.org/10.1515/eletel-2016-0031
[20] William von Hagen. 2022. *Professional Xen Virtualization.* Wrox.
[21] Hardkernel. [n. d.]. *ODROID-N2+ with 2 GB RAM.* https://www.hardkernel.com/shop/odroid-n2-with-2gbyte-ram-2
[22] Hardkernel. [n. d.]. *ODROID-N2+ with 4 GB RAM.* https://www.hardkernel.com/shop/odroid-n2-with-4gbyte-ram-2
[23] Darren James Harkness. 2022. *Apache Essentials: Install, Configure, Maintain* (2nd. ed.). Apress.
[24] Jinho Hwang, Sai Zeng, Frederick Wu, and Timothy Wood. 2013. A Component-based Performance Comparison of Four Hypervisors. In *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013).* Ghent, Belgium.
[25] Konstantin Ivanov. 2017. *KVM Virtualization Cookbook: Learn How to Use KVM Effectively in Production* (1st. ed.). Packt Publishing.
[26] Samad Salehi Kolahi, Vichar S. Hora, Amrit P. Singh, Sherazia Bhatti, and Susmitha R. Yeeda. 2020. Performance Comparison of Cloud Computing/IoT Virtualization Software, Hyper-V vs vSphere. In *2020 Advances in Science and Engineering Technology International Conferences (ASET 2020).* Dubai, United Arab Emirates. https://doi.org/10.1109/ASET48392.2020.9118185
[27] Anshul Kumar and Raj Singh. 2018. Guest Operating System Based Performance Comparison of VMware and Xen Hypervisor. *International Journal of Research, Science, Technology & Management* 10, II (Feb. 2018), 1–12.
[28] Roberto Morabito, Jimmy Kjällman, and Miika Komu. 2015. Hypervisors vs. Lightweight Virtualization: A Performance Comparison. In *2015 IEEE International Conference on Cloud Engineering (IC2E 2015).* Tempe, Arizona, USA.
[29] Swati Pawar and Sarvesh Singh. 2015. Performance Comparison of VMware and Xen Hypervisor on Guest OS. *International Journal of Innovative Computer Science & Engineering* 2, 3 (Aug. 2015), 56–60.
[30] Shivananda R. Poojara, N. V. Dharwadkar, and Vishal Ghule. 2017. Performance Benchmarking of Hypervisors – A Case Study. *Indian Journal of Science and Technology* 10, 44 (Nov. 2017), 2–11. https://doi.org/10.17485/ijst/2017/v10i44/120579
[31] Matthew Portnoy. 2016. *Virtualization Essentials* (2nd. ed.). Sybex.
[32] Proxmox. [n. d.]. *Proxmox Virtual Environment.* https://www.proxmox.com/en/proxmox-ve

[33] Raspberry Pi Foundation. [n. d.]. *Raspberry Pi 4 Model B.* https://www.raspberrypi.com/products/raspberry-pi-4-model-b

[34] Raspberry Pi Foundation. [n. d.]. *Raspberry Pi 400.* https://www.raspberrypi.com/products/raspberry-pi-400

[35] Raspberry Pi Foundation. [n. d.]. *Raspberry Pi OS.* https://downloads.raspberrypi.org

[36] Antonio Gomes Rodrigues, Bruno Demion, and Philippe Mouawad. 2019. *Master Apache JMeter - From Load Testing to DevOps: Master Performance Testing with JMeter* (2nd. ed.). Packt Publishing.

[37] Carla Schroder. 2021. *Linux Cookbook: Essential Skills for Linux Users and System & Network Administrators* (2nd. ed.). O'Reilly Media.

[38] Sogand Shirinbab, Lars Lundberg, and Dragos Ilie. 2014. Performance Comparison of KVM, VMware and XenServer using a Large Telecommunication Application. In *5th International Conference on Cloud Computing, GRIDs, and Virtualization (Cloud Computing 2014)*. Venice, Italy.

[39] Sebouh Toumassian, Rico Werner, and Axel Sikora. 2016. Performance Measurements for Hypervisors on Embedded ARM Processors. In *2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI 2016)*. Jaipur, India. https://doi.org/10.1109/ICACCI.2016.7732152