

Performance Evaluation in Ethernet and WiFi Scenarios

Andrea Botticella^{*†}

andrea.botticella@studenti.polito.it

Renato Mignone^{*†}

renato.mignone@studenti.polito.it

Elia Innocenti^{*†}

elia.innocenti@studenti.polito.it

Simone Romano^{*†}

simone.romano2@studenti.polito.it

ABSTRACT

This report examines the performance of wireless and device-to-device communication by comparing theoretical predictions with experimental results in three scenarios: WiFi-only, Ethernet-only, and a mixed configuration. Using iperf3 and Wireshark, we measured goodput and analyzed its variability under different conditions. The experimental data were contrasted with theoretical estimates based on protocol efficiencies and network overheads. Our findings underscore Ethernet's stability and highlight the challenges of WiFi's shared medium and half-duplex constraints.

1 BACKGROUND AND OBJECTIVES

- 1 This laboratory evaluates and compares the performance of wired
2 and wireless communication in a local area network by setting
3 up three scenarios: both devices on WiFi, both on Ethernet, and a
4 mixed configuration with one on each. The main objectives of the
5 lab are to:
- 6 • Measure performances using iperf3 and Wireshark [7, 8].
 - 7 • Analyze the variability and stability of the connection in each
8 scenario by collecting data over multiple test runs.
 - 9 • Compare the experimental results against theoretical predictions
10 based on protocol efficiencies and network overhead.
 - 11 • Investigate potential sources of performance degradation in wire-
12 less communication, such as interference, half-duplex operation,
13 and shared medium limitations.
 - 14 These experiments provide practical insights into the strengths and
15 limitations of both Ethernet and WiFi, crucial for optimizing mixed
16 network performance.

2 METHODOLOGY AND CONCEPTS

- 17 This section outlines the experimental setup, the tools employed for
18 the measurements, and the theoretical basis for estimating goodput.

19 2.1 Selected Tools

- 20 To evaluate the performance of both Ethernet and WiFi connections,
21 we utilized several specialized tools:
- 22 • **iperf3**: Used to generate traffic and measure goodput in both
23 TCP and UDP modes. By executing repeated tests, iperf3 provides
24 key metrics such as minimum, maximum, average, and standard
25 deviation of the throughput.
 - 26 • **Wireshark**: Used to capture and analyze network traffic, Wire-
27 shark helped inspect data flows, identify frames, and validate
28 results. It also generated useful charts for analyzing TCP streams.

^{*}The authors collaborated closely in developing this project.

[†]All the authors are students at Politecnico di Torino, Turin, Italy.

- **Automation Script**: A Python script was developed to automate the entire measurement process. This script manages both server and client modes of iperf3, logs output, and computes summary statistics. The script accepts several command-line flags, as detailed in the Appendix A.

2.2 Goodput Estimation

Goodput represents the rate at which useful data is delivered to the application layer, excluding protocol overheads and retransmitted packets. The theoretical estimation of goodput is based on the efficiency of the protocol and the capacity of the network link:

$$G \leq \eta_{\text{protocol}} \times C,$$

where C is the capacity of the bottleneck link and η_{protocol} is the protocol efficiency.

1. For **Ethernet** [1], the efficiency for TCP is computed as:

$$\eta_{TCP}^{Eth} = \frac{MSS}{MSS + \text{TCP headers} + \text{IP headers} + \text{Eth. overhead}},$$

with the Maximum Segment Size (MSS) defined as the MTU minus the headers. For a standard MTU of 1500 bytes, we obtain:

- $MSS \approx 1460$ bytes (after subtracting 20 bytes for the IP header and 20 bytes for the TCP header),
- An additional Ethernet overhead of approximately 38 bytes.

Thus, the efficiency for TCP over Ethernet is approximately:

$$\eta_{TCP}^{Eth} \approx \frac{1460}{1460 + 20 + 20 + 38} \approx 94.9\%.$$

Similarly, the efficiency for UDP is computed as follows. Since UDP has an 8-byte header, its MSS is given by:

- $MSS \approx 1472$ bytes (after subtracting 20 bytes for the IP header and 8 bytes for the UDP header).

Thus, the efficiency for UDP over Ethernet is given by:

$$\eta_{UDP}^{Eth} \approx \frac{1472}{1472 + 20 + 8 + 38} \approx 95.7\%.$$

2. For **WiFi** [2], additional factors must be considered due to its half-duplex nature and the inherent overhead of the 802.11 protocol (e.g., control frames, retransmissions, and channel contention). In the case of **TCP over WiFi**, the effective efficiency is typically around 80% under optimal conditions. This lower efficiency arises from the extra overhead associated with TCP's connection-oriented features—such as congestion control, flow control, and the guarantee of in-order delivery—which require additional control packets and retransmissions. In contrast, **UDP over WiFi** generally attains an efficiency of approximately 85–90% by avoiding these mechanisms, leading to a simpler and faster data transmission process.

$$\eta_{TCP}^{WiFi} (\approx 80\%) \quad \text{and} \quad \eta_{UDP}^{WiFi} (\approx 85\%-90\%).$$

These theoretical estimates set an upper bound on the achievable goodput, against which our experimental results are compared. Discrepancies between theory and practice are primarily due to dynamic environmental factors, such as interference, channel variability, and the inherent limitations of wireless communication.

3 EXPERIMENTAL SETUP AND TEST CASES

3.1 Equipment and Configuration

In this section, we describe the hardware and software configuration used to perform our network performance measurements. Table 1 summarizes the main devices, their interfaces, and relevant specifications.

Device	Key Specifications
PC1	Victus 16-s1005nl Notebook <i>Operating System:</i> Ubuntu 24.04.2 LTS [11] <i>Ethernet Interface:</i> Realtek RTL8111/8168/8211/8411 [3] <i>Wireless Interface:</i> Realtek RTL8852BE (802.11ax) 2x2 [4]
PC2	Microsoft Surface Laptop Go 3 <i>Operating System:</i> Ubuntu 24.10 [11] <i>Ethernet Interface:</i> via Anker PowerExpand+ USB-C Hub [10] <i>Wireless Interface:</i> Intel Alder Lake-P CNVi (802.11ax) 2x2 [5]
Router	Vodafone Power Station Wi-Fi 6 <i>Ethernet Ports:</i> 4 × 1 Gbps <i>Wi-Fi:</i> Dual-band 802.11ax (2.4 GHz 2x2, 5 GHz 4x4) [12] <i>Default gateway ip:</i> 192.168.1.1
Cables	CAT.5E (up to 1 Gbps)

Table 1: Summary of Hardware and Network Configuration

Connection	Key Specifications
Ethernet	<i>Cabling:</i> CAT.5E <i>Nominal Speed:</i> 1 Gbps <i>Client ip:</i> 192.168.1.12
Wi-Fi	<i>Standard:</i> 802.11ax <i>Nominal Speed:</i> 1200 Mbps <i>Bandwidth:</i> 80 MHz <i>Client ip:</i> 192.168.1.8 <i>Third host ip:</i> 192.168.1.13 (shared capacity scenario)

Table 2: Ethernet and Wi-Fi Connection Specifications

This hardware setup allows us to compare Ethernet versus Wi-Fi performance under a consistent router and cabling environment. In the next section, we detail the evaluation scenarios and the measurement methodology.

3.2 Evaluation Scenarios

We considered three distinct network configurations to assess the performance differences between wired and wireless communications. For each scenario, the theoretical goodput is computed based on the nominal link capacity and protocol efficiency.

1. Both Ethernet:

In this scenario, both PC1 and PC2 are connected to the router via CAT.5E cables, providing a nominal link capacity of 1 Gbps. The efficiency for Ethernet is calculated as follows:

$$\eta_{TCP}^{Eth} \approx \frac{1460}{1460 + 20 + 20 + 38} \approx 94.9\%,$$

$$\eta_{UDP}^{Eth} \approx \frac{1472}{1472 + 20 + 8 + 38} \approx 95.7\%.$$

Thus, the expected goodput is:

$$G_{TCP}^{Eth} \leq 0.949 \times 1000 \text{ Mbps} \approx 949 \text{ Mbps},$$

$$G_{UDP}^{Eth} \leq 0.957 \times 1000 \text{ Mbps} \approx 957 \text{ Mbps}.$$

2. Both Wi-Fi:

For this configuration, both devices use their wireless interfaces (802.11ax) to connect to the router. Although the nominal Wi-Fi link speed is assumed to be approximately 1.2 GbEbps, the half-duplex nature of Wi-Fi effectively halves the throughput available for data transfer. Assuming a Wi-Fi efficiency factor of about 80%, the expected goodput for TCP is:

$$G_{TCP}^{WiFi} \leq 0.80 \times 1.2 \text{ Gbps} \times \frac{1}{2} \approx 480 \text{ Mbps},$$

and similarly for UDP, with a different efficiency factor:

$$G_{UDP}^{WiFi} \leq 0.85 \times 1.2 \text{ Gbps} \times \frac{1}{2} \approx 510 \text{ Mbps}.$$

3. Mixed Scenario:

In this configuration, one device (PC1) is connected via Ethernet while the other (PC2) uses its Wi-Fi interface. Since only one side is on Wi-Fi, we average the Wi-Fi portion and the Ethernet portion rather than halving for two Wi-Fi paths. Hence, the expected goodput is:

$$G_{TCP}^{Mixed} \leq \frac{(0.80 \times 1.2 \text{ Gbps}) + (0.949 \times 1.0 \text{ Gbps})}{2} \approx 955 \text{ Mbps},$$

$$G_{UDP}^{Mixed} \leq \frac{(0.85 \times 1.2 \text{ Gbps}) + (0.957 \times 1.0 \text{ Gbps})}{2} \approx 989 \text{ Mbps}.$$

These calculations provide the theoretical upper bounds for goodput in each scenario. The experimental results, obtained via automated measurements using the provided Python script, are compared against these predictions to evaluate real-world performance.

4 ANALYSIS AND FINDINGS

4.1 TCP Performance

The performance tests using TCP reveal several noteworthy trends. The analysis for TCP performance in different scenarios is organized as follows:

Test	TCP: Goodput per flow (Mbps)				
	Prediction	Average	Min	Max	Std
Both WiFi	480	434.7	396.3	461.96	22.5
Both Ethernet	949	939.6	938.2	942.7	1.5
Mixed	955	663.7	619.2	698.86	26.6

Table 3: TCP Results (Client → Server)

113 **1. Both Ethernet:**

114 Figure 1 shows the TCP throughput measured in the Ethernet scenario. The graph reveals a rapid ramp-up in throughput during
 115 the first few seconds, followed by a stable transmission rate that approaches the theoretical value. The **Maximum Segment**
 116 **Size (MSS)** reaches and remains stable at 1500 bytes, as defined by the TCP protocol. Additionally, the **bandwidth** is stable at
 117 **950 Mbps**, as indicated by the results and the low standard deviation.

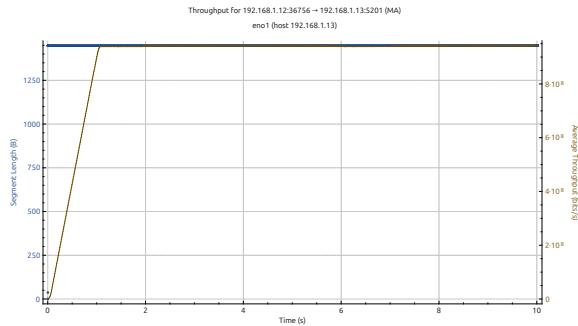


Figure 1: TCP Throughput in the Ethernet Scenario.

122 Figure 2 illustrates the round-trip time (RTT), which remains
 123 very low (typically within 1-3 milliseconds), highlighting the
 124 minimal latency in wired connections.

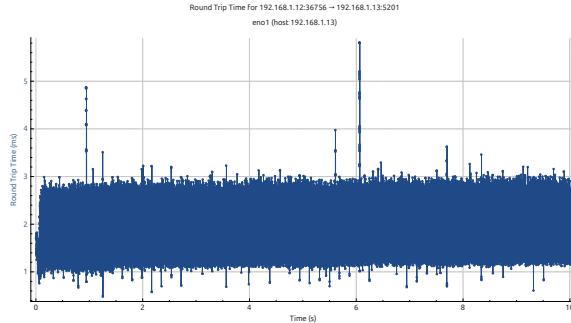


Figure 2: TCP Round Trip Time in the Ethernet Scenario.

125 Overall, the Ethernet scenario demonstrates a near-ideal performance with high throughput and minimal latency, closely
 126 matching the theoretical predictions.

127 **2. Both WiFi:**

128 In the WiFi scenario, the throughput graph (Fig. 3) shows an
 129 initial ramp-up phase during the first 2 seconds, after which the
 130 throughput fluctuates around an average value of 434.7 Mbps.
 131 These fluctuations suggest that protocol overhead, wireless
 132 interference, and the half-duplex nature of WiFi adversely affect
 133 performance.

134 The round-trip time (RTT) measurements (Fig. 4) reveal RTT
 135 values ranging from about 20-50 ms, indicating intermittent
 136 delays likely due to congestion and contention in the wireless
 137 medium.

138 Overall, while the theoretical capacity for TCP over WiFi is
 139 estimated to be around 480 Mbps, the experimental data indicate
 140 that real-world factors does not reduce that much the effective

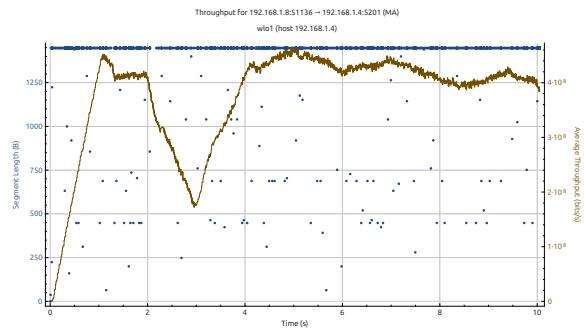


Figure 3: TCP Throughput in the WiFi Scenario.

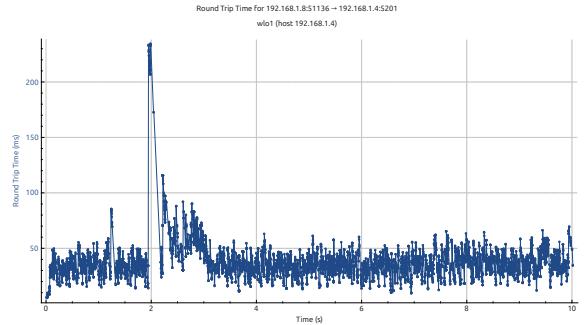


Figure 4: TCP Round Trip Time in the WiFi Scenario.

142 throughput of the WiFi network. This due to the fact that the
 143 network was in an ideal condition, where only the client and
 144 the server were connected to the access point, and no other
 145 devices were generating traffic.

146 **3. Mixed:**

147 Figure 5 displays the TCP throughput for the mixed configuration.
 148 The graph shows that the throughput reaches a stable level
 149 after an initial ramp-up phase, although it remains below the
 150 Ethernet scenario and is consistent with the expected reduction
 151 due to the reliance on the wireless link.

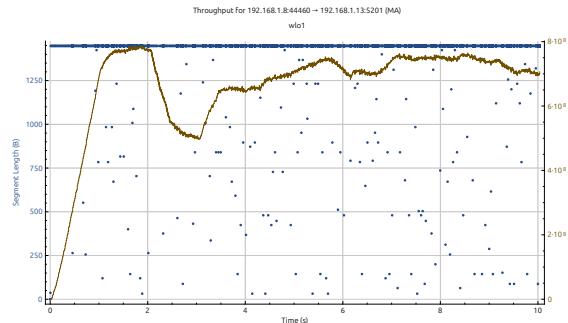


Figure 5: TCP Throughput in the Mixed Ethernet/WiFi Scenario.

152 The round-trip time (RTT) measurements, presented in Figure 6, indicate moderate latency, with RTT values generally remaining
 153 within a lower range compared to the pure WiFi scenario. This suggests that the wired segment helps in reducing overall
 154 latency.

142 throughput of the WiFi network. This due to the fact that the
 143 network was in an ideal condition, where only the client and
 144 the server were connected to the access point, and no other
 145 devices were generating traffic.

146 **3. Mixed:**
 147 Figure 5 displays the TCP throughput for the mixed configuration.
 148 The graph shows that the throughput reaches a stable level
 149 after an initial ramp-up phase, although it remains below the
 150 Ethernet scenario and is consistent with the expected reduction
 151 due to the reliance on the wireless link.

152 The round-trip time (RTT) measurements, presented in Figure 6, indicate moderate latency, with RTT values generally remaining
 153 within a lower range compared to the pure WiFi scenario. This suggests that the wired segment helps in reducing overall
 154 latency.

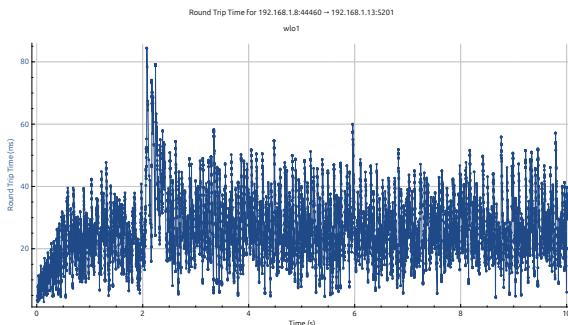


Figure 6: TCP Round Trip Time in the Mixed Ethernet/WiFi Scenario.

3a. Shared Capacity:

In this scenario, a third host connected to the same access point was concurrently downloading the film "Natale a Rio" [14] (directed by Neri Parenti), which introduced significant interference during the tests. This additional traffic compromised the available network capacity, leading to degraded performance. As we can see in (Fig. 7), the other host starts the download from the second test (~25 seconds). In fact the value of the throughput, goes from the one of the standard Mixed Scenario (~670Mbps), to a lower one (~500Mbps). This behavior is due to the fact that the bandwidth is shared between the two hosts, and the download of the movie is consuming useful bandwidth.

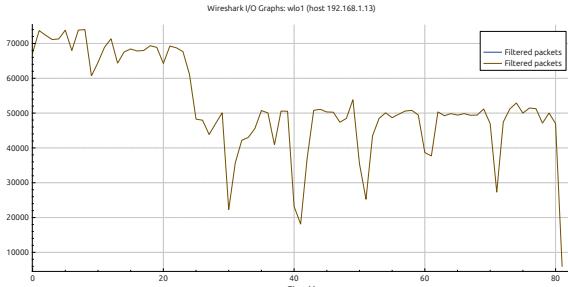


Figure 7: Wireshark I-O Graph for TCP in the Shared Capacity Scenario.

4.2 UDP Performance

The UDP tests offer an insightful comparison to the TCP results by eliminating congestion control and acknowledgment overhead. The analysis for UDP performance across different scenarios is structured as follows:

Test	UDP: Goodput per flow (Mbps)				
	Prediction	Average	Min	Max	Std
Both WiFi	510	487.8	453.1	499.9	15.8
Both Ethernet	957	952.8	948.3	954.6	1.73
Mixed	989	674.9	636.6	717.8	28

Table 4: UDP Results (Client → Server)

1. Both Ethernet:

The throughput achieved is very close to the theoretical prediction, confirming that the wired setup reliably supports high-speed data transfer. The absence of retransmission or congestion

control overhead in UDP further contributes to this consistent performance. In summary, the wired (Ethernet) tests demonstrate that both TCP and UDP protocols achieve performance levels very close to their theoretical capacity. This confirms that, in a controlled wired environment, network performance is minimally impacted by protocol overhead or environmental factors.

2. Both WiFi:

In the WiFi scenario, UDP shows a steadier flow than TCP. Although its reduced overhead allows for marginally higher instantaneous throughput, the overall performance still falls short of the theoretical maximum due to interference and channel contention. This comparison highlights how wireless limitations, such as interference and protocol overhead, impact real-world throughput.

3. Mixed:

In the mixed scenario, UDP transmissions exhibit a generally consistent flow, benefiting from the absence of congestion control overhead. The presence of a wired link on one end reduces overall latency and provides more stable performance compared to a fully wireless setup. Nonetheless, the wireless segment remains the principal bottleneck, limiting the achievable throughput. Overall, both TCP and UDP in the mixed scenario approximate their respective theoretical predictions, illustrating how partial reliance on Ethernet can mitigate some of the challenges inherent to WiFi.

3a. Shared Capacity:

The UDP tests under the shared capacity scenario also reveal the negative impact of the additional download traffic. Despite the inherent resilience of UDP to retransmission delays, the interference from the third host causes a noticeable reduction in performance. Overall, the shared capacity scenario clearly demonstrates that when a third host generates significant traffic (as in the case of streaming a movie), the available network resources are further divided, leading to performance degradation for both TCP and UDP protocols. This scenario highlights the importance of considering real-world usage patterns and interference when designing and evaluating network performance.

5 CONCLUSION

This project compared wired (Ethernet) and wireless (WiFi) network performance using iperf3 measurements and Wireshark analysis. Ethernet tests approached theoretical goodput with minimal latency, confirming the efficiency of controlled wired environments. In contrast, WiFi exhibited lower throughput and higher latency due to interference, contention, and half-duplex constraints. The mixed scenario showed that while a wired link can reduce latency, the wireless segment remains the bottleneck. Finally, adding a third host performing heavy traffic (movie streaming) further degraded both TCP and UDP results, highlighting how real-world congestion significantly impacts performance. These observations underscore the need to consider environmental and traffic factors in network design. While theoretical models provide upper bounds, practical limitations such as interference and shared medium contention ultimately determine real-world performance.

A APPENDIX

231 Server Mode Initialization

```
232     def run_server():
233         """
234             Run iperf3 server with clean output handling.
235             """
236             server_logger.info("Starting iperf3 server...")
237
238             proc = subprocess.Popen(
239                 ["iperf3", "-s", "-J"],
240                 stdout=subprocess.PIPE,
241                 stderr=subprocess.PIPE,
242                 text=True,
243                 bufsize=1,
244             )
245             # Handle server output and errors in a separate
246             # thread...
```

Listing 1: Excerpt for server mode initialization.

247 Client Mode Execution and Reporting

```
248     def run_client(server_ip, udp=False, bitrate="1M",
249                     iterations=10):
250         """
251             Run iperf3 client tests and generate reports.
252             """
253             for i in range(iterations):
254                 cmd = ["iperf3", "-c", server_ip, "-J", "-t",
255                         "10", "-i", "1"]
256                 if udp:
257                     cmd.extend(["-u", "-b", bitrate])
258
259                 result = subprocess.run(
260                     cmd,
261                     capture_output=True,
262                     text=True,
263                     check=True
264                 )
265
266                 data = json.loads(result.stdout)
267                 # Extract test data, compute statistics, and
268                 log results...
```

Listing 2: Excerpt for client mode execution.

269 Logging and Output Management

```
270     def setup_logger(log_file, name):
271         logger = logging.getLogger(name)
272         logger.setLevel(logging.DEBUG)
273         handler = logging.FileHandler(log_file)
274         formatter = logging.Formatter(
275             '%(asctime)s - %(levelname)s - %(message)s'
276         )
277         handler.setFormatter(formatter)
278         logger.addHandler(handler)
279         return logger
```

Listing 3: Excerpt for logging setup.

280 These snippets illustrate how the script handles server mode ini-
281 tialization, client tests (both TCP and UDP), and logging. Error
282 handling, multithreaded stderr management, and CSV report gen-
283 eration are also included in the complete script.

All the code for the report and lab material is available on the repository [13]: <https://github.com/WDCSecure/LabWiFi.git> 284
285

REFERENCES

- [1] 2018. IEEE Standard for Ethernet. (2018). https://standards.ieee.org/standard/802_3-2018.html 286
287
- [2] 2021. IEEE Standard for Information technology—Telecommunications and information exchange between systems Local and metropolitan area networks—Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 5: Enhancements for High Efficiency WLAN. (2021). https://standards.ieee.org/standard/802_11ax-2021.html 288
289
- [3] Realtek Semiconductor Corp. 2023. *Realtek RTL8111/8168/8211/8411: PCI Express Gigabit Ethernet controllers*. <https://www.realtek.com> 290
291
- [4] Realtek Semiconductor Corp. 2023. *Realtek RTL8852BE: 802.11ax Wi-Fi 6 PCIe network interface card*. <https://www.realtek.com> 292
293
- [5] Intel Corporation. 2023. *Intel Alder Lake-P CNVi: Integrated Wi-Fi 6E solution*. <https://www.intel.com/> 294
295
- [6] Python Software Foundation. 2023. *Python: A programming language that lets you work quickly and integrate systems more effectively*. <https://www.python.org/> 296
297
- [7] The Wireshark Foundation. 2023. *Wireshark: The world's foremost network protocol analyzer*. <https://www.wireshark.org/> 298
299
- [8] The iperf3 Development Team. 2023. *iperf3: A TCP, UDP, and SCTP network bandwidth measurement tool*. <https://iperf.fr/> 300
301
- [9] Jangeun Jun, P. Peddabachagari, and M. Sichitiu. 2003. Theoretical maximum throughput of IEEE 802.11 and its applications. In *Second IEEE International Symposium on Network Computing and Applications, 2003. NCA 2003*. 249–256. <https://doi.org/10.1109/NCA.2003.1201163> 302
303
- [10] Anker Innovations Limited. 2023. *Anker PowerExpand+ USB-C Hub: A USB-C hub with Ethernet and other interfaces*. <https://www.anker.com/> 304
305
- [11] Canonical Ltd. 2023. *Ubuntu: An open-source software platform that runs everywhere from the PC to the server and the cloud*. <https://ubuntu.com/> 306
307
- [12] Vodafone Group Plc. 2023. *Vodafone Power Station Wi-Fi 6: A dual-band Wi-Fi 6 router*. <https://www.vodafone.it/privati/area-supporto/assistenza-dispositivi/vodafone-station/vodafone-power-station-wifi6.html> 308
309
- [13] WDCSecure. 2025. LabWiFi: repository containing the material and the documentation for the WiFi Lab. (2025). <https://github.com/WDCSecure/LabWiFi.git> 310
311
- [14] Wikipedia. 2008. Natale a Rio. (2008). https://en.wikipedia.org/wiki/Natale_a_Rio 312
313