

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	<i>Protocol:</i> Ethernet II <i>Server ip:</i> 192.168.1.13
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)	<i>Security Protocol:</i> WPA2-AES <i>Frequency:</i> 5 GHz <i>Channel:</i> 100 <i>Server ip:</i> 192.168.1.4

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. So the expected goodput is determined by the slower link, so we compute the min between the two. Since only one side is on Wi-Fi, the Wi-Fi estimated goodput has not to be halved. Thus, the theoretical goodput is:

$$G_{TCP}^{Mixed} \leq \min\{0.80 \times 1.2 \text{ Gbps}, 0.949 \times 1.0 \text{ Gbps}\} \approx 949 \text{ Mbps},$$

$$G_{UDP}^{Mixed} \leq \min\{0.85 \times 1.2 \text{ Gbps}, 0.957 \times 1.0 \text{ Gbps}\} \approx 957 \text{ Mbps}.$$

We can see that the Eth. link is the bottleneck in this scenario.

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

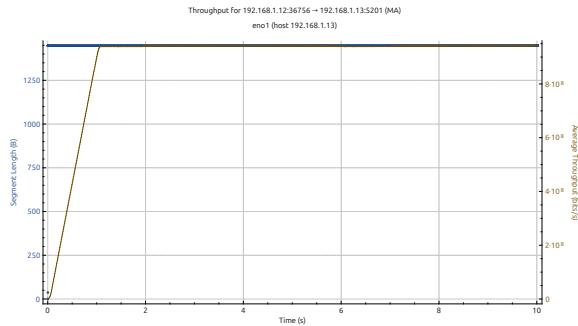
The performance tests using TCP reveal several noteworthy trends. The analysis for TCP performance in different scenarios is organized as follows:

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

Table 3: TCP Results (Client → Server)

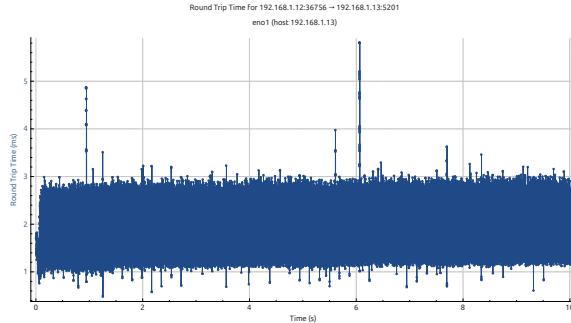
115 **1. Both Ethernet:**

116 Figure 1 shows the TCP throughput measured in the Ethernet scenario. The graph reveals a rapid ramp-up in throughput during
 117 the first few seconds, followed by a stable transmission rate that approaches the theoretical value. The **Maximum Segment**
 118 **Size (MSS)** reaches and remains stable at 1500 bytes, as defined by the TCP protocol. Additionally, the **bandwidth** is stable at
 119 **950 Mbps**, as indicated by the results and the low standard deviation.



120 **Figure 1:** TCP Throughput in the Ethernet Scenario.
 121

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.



125 **Figure 2:** TCP Round Trip Time in the Ethernet Scenario.
 126

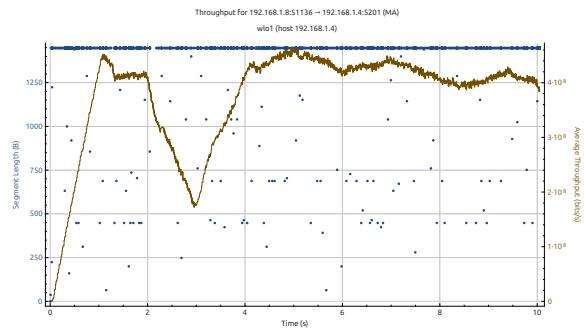
127 Overall, the Ethernet scenario demonstrates a near-ideal per-
 128 formance with high throughput and minimal latency, closely
 129 matching the theoretical predictions.

130 **2. Both WiFi:**

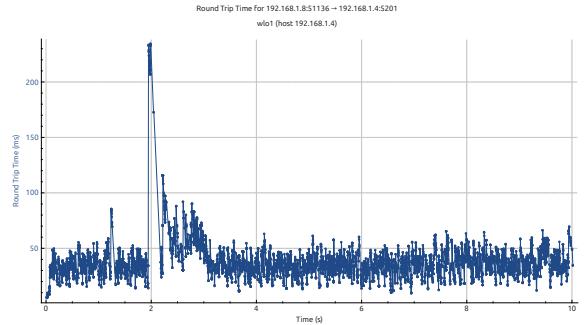
131 In the WiFi scenario, the throughput graph (Fig. 3) shows an
 132 initial ramp-up phase during the first 2 seconds, after which the
 133 throughput fluctuates around an average value of 434.7 Mbps.
 134 These fluctuations suggest that protocol overhead, wireless
 135 interference, and the half-duplex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

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

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



144 **Figure 3:** TCP Throughput in the WiFi Scenario.
 145

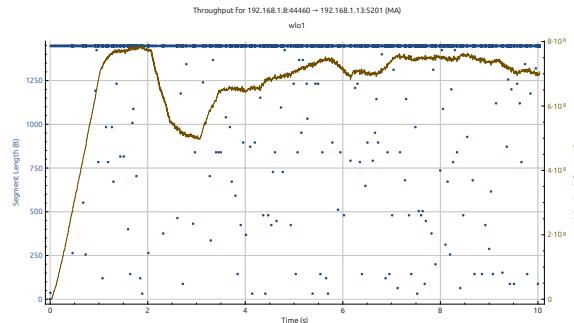


146 **Figure 4:** TCP Round Trip Time in the WiFi Scenario.
 147

148 throughput of the WiFi network. This due to the fact that the
 149 network was in an ideal condition, where only the client and
 150 the server were connected to the access point, and no other
 151 devices were generating traffic.

152 **3. Mixed:**

153 Figure 5 displays the TCP throughput for the mixed configura-
 154 tion. The graph shows that the throughput reaches a stable level
 155 after an initial ramp-up phase, although it remains below the
 156 Ethernet scenario and is consistent with the expected reduction
 157 due to the reliance on the wireless link.



158 **Figure 5:** TCP Throughput in the Mixed Ethernet/WiFi Scenario.
 159

160 The round-trip time (RTT) measurements, presented in Figure 6,
 161 indicate moderate latency, with RTT values generally remain-
 162 ing within a lower range compared to the pure WiFi scenario.
 163 This suggests that the wired segment helps in reducing overall
 164 latency.

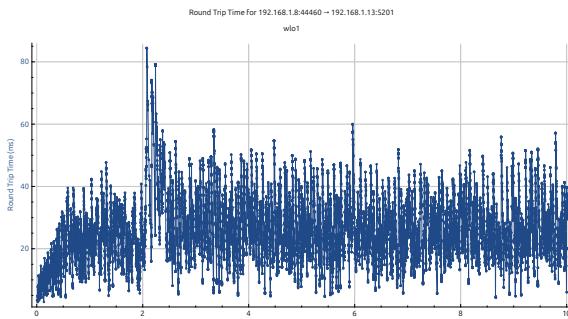


Figure 6: TCP Round Trip Time in the Mixed Ethernet/WiFi Scenario.

159 3a. Shared Capacity:

160 In this scenario, a third host connected to the same access point
 161 was concurrently downloading the film "Natale a Rio" [14] (di-
 162 rected by Neri Parenti), which introduced significant interfer-
 163 ence during the tests. This additional traffic compromised the
 164 available network capacity, leading to degraded performance.
 165 As we can see in (Fig. 7), the other host starts the download
 166 from the second test (~25 seconds). In fact the value of the
 167 throughput, goes from the one of the standard Mixed Scenario
 168 ($\sim 670\text{Mbps}$), to a lower one ($\sim 500\text{Mbps}$). This behavior is due
 169 to the fact that the bandwidth is shared between the two hosts,
 170 and the download of the movie is consuming useful bandwidth.

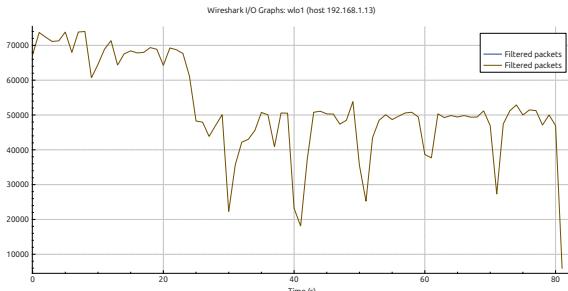


Figure 7: Wireshark I-O Graph for TCP in the Shared Capacity Scenario.

171 4.2 UDP Performance

172 The UDP tests offer an insightful comparison to the TCP results
 173 by eliminating congestion control and acknowledgment overhead.
 174 The analysis for UDP is structured as follows:

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

Table 4: UDP Results (Client → Server)

175 1. Both Ethernet:

176 In the Ethernet scenario, UDP achieves a near-theoretical through-
 177 put of **952.8 Mbps** (vs. TCP's 939.6 Mbps), with minimal stan-
 178 dard deviation (**1.73 Mbps**). The absence of retransmissions or
 179 congestion control allows UDP to utilize the full wired capacity.

While TCP's latency remains marginally lower (1–3 ms RTT)
 180 due to acknowledgment-based stability, UDP's lack of overhead
 181 enables slightly higher throughput.
 182
 183

184 2. Both WiFi:

185 UDP averages **487.8 Mbps** (vs. TCP's 434.7 Mbps) in WiFi,
 186 achieving a **~12% throughput advantage** by avoiding TCP's
 187 congestion control. Despite outperforming TCP, UDP falls short
 188 of the **510 Mbps theoretical maximum** due to WiFi interfe-
 189 rence and contention. Moreover, it shows lower variability (std.
 190 dev. **15.8 Mbps** vs. TCP's **22.5 Mbps**), indicating smoother per-
 191 formance. This makes UDP preferable for real-time applications
 192 prioritizing speed over error correction.
 193

194 3. Mixed:

195 The mixed scenario shows UDP achieving **674.9 Mbps** (vs.
 196 TCP's 663.7 Mbps). The wired segment reduces latency, but the
 197 wireless link remains the bottleneck. UDP's performance aligns
 198 closely with TCP here, as both protocols are constrained by
 199 WiFi's limitations. UDP's throughput is **1.8% higher** than TCP,
 200 reflecting its ability to bypass congestion control.
 201

202 3a. Shared Capacity:

203 In the shared capacity test, UDP throughput drops significantly.
 204 The third host's download introduces contention, reducing
 205 available bandwidth. UDP's lack of congestion control leads
 206 to aggressive transmission attempts, but packet drops result in
 207 lower effective throughput. Unlike TCP, which degrades pre-
 208 dictably due to its back-off mechanism, UDP's performance
 209 becomes more volatile. This highlights TCP's adaptability in
 210 shared environments, where fairness and resource allocation are
 211 critical, while UDP's rigidity makes it less suitable for contested
 212 networks.
 213

214 5 CONCLUSION

215 This study evaluated TCP and UDP performance across wired, wire-
 216 less, and hybrid networks. Results confirm that **Ethernet envi-
 217 ronments** maximize protocol efficiency, enabling near-theoretical
 218 throughput with minimal latency. In contrast, **WiFi introduces
 219 variability** due to interference and contention, impacting both
 220 protocols despite controlled conditions.
 221

222 **TCP** prioritizes reliability, making it robust for file transfers and
 223 web traffic, though its congestion control limits performance in
 224 dynamic environments. **UDP**, while faster in ideal conditions, strug-
 225 gles with packet loss and shared resources, rendering it less suitable
 226 for congested networks.
 227

228 **Hybrid Ethernet-WiFi setups** highlight the wireless segment as
 229 the primary bottleneck, with both protocols constrained by WiFi's
 230 instability. Shared capacity scenarios further degrade performance,
 231 emphasizing the need for adaptive protocols in contested networks.
 232 These findings underscore the importance of aligning protocol
 233 choice with environmental constraints. While theoretical models
 234 provide benchmarks, real-world performance hinges on interfer-
 235 ence, medium contention, and protocol design. Network planning
 236 must balance throughput, latency, and resilience to dynamic con-
 237 ditions.
 238

A APPENDIX

234 Server Mode Initialization

```
235     def run_server():
236         """
237             Run iperf3 server with clean output handling.
238             """
239             server_logger.info("Starting iperf3 server...")
240
241             proc = subprocess.Popen(
242                 ["iperf3", "-s", "-J"],
243                 stdout=subprocess.PIPE,
244                 stderr=subprocess.PIPE,
245                 text=True,
246                 bufsize=1,
247             )
248             # Handle server output and errors in a separate
249             # thread...
```

Listing 1: Excerpt for server mode initialization.

250 Client Mode Execution and Reporting

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

Listing 2: Excerpt for client mode execution.

272 Logging and Output Management

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

Listing 3: Excerpt for logging setup.

283 These snippets illustrate how the script handles server mode ini-
284 tialization, client tests (both TCP and UDP), and logging. Error
285 handling, multithreaded stderr management, and CSV report gen-
286 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> 287
288

REFERENCES

- [1] 2018. IEEE Standard for Ethernet. (2018). https://standards.ieee.org/standard/802_3-2018.html 289
290
- [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 291
292
- [3] Realtek Semiconductor Corp. 2023. *Realtek RTL8111/8168/8211/8411: PCI Express Gigabit Ethernet controllers*. <https://www.realtek.com> 296
297
- [4] Realtek Semiconductor Corp. 2023. *Realtek RTL8852BE: 802.11ax Wi-Fi 6 PCIe network interface card*. <https://www.realtek.com> 298
299
- [5] Intel Corporation. 2023. *Intel Alder Lake-P CNVi: Integrated Wi-Fi 6E solution*. <https://www.intel.com/> 300
301
- [6] Python Software Foundation. 2023. *Python: A programming language that lets you work quickly and integrate systems more effectively*. <https://www.python.org/> 302
303
- [7] The Wireshark Foundation. 2023. *Wireshark: The world's foremost network protocol analyzer*. <https://www.wireshark.org/> 304
305
- [8] The iperf3 Development Team. 2023. *iperf3: A TCP, UDP, and SCTP network bandwidth measurement tool*. <https://iperf.fr/> 306
307
- [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> 308
309
- [10] Anker Innovations Limited. 2023. *Anker PowerExpand+ USB-C Hub: A USB-C hub with Ethernet and other interfaces*. <https://www.anker.com/> 312
313
- [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/> 314
315
- [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> 316
317
- [13] WDCSecure. 2025. LabWiFi: repository containing the material and the documentation for the WiFi Lab. (2025). <https://github.com/WDCSecure/LabWiFi.git> 319
320
- [14] Wikipedia. 2008. Natale a Rio. (2008). https://en.wikipedia.org/wiki/Natale_a_Rio 321
322