Performance Evaluation in Ethernet and WiFi Scenarios

Andrea Botticella* †
andrea.botticella@studenti.polito.it

Elia Innocenti* †
elia.innocenti@studenti.polito.it

Renato Mignone* †
renato.mignone@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

This laboratory evaluates and compares the performance of wired and wireless communication in a local area network by setting up three scenarios: both devices on WiFi, both on Ethernet, and a mixed configuration with one on each. The main objectives of the lab are to:

- Measure performances using iperf3 and Wireshark [7, 8].
- Analyze the variability and stability of the connection in each scenario by collecting data over multiple test runs.
- Compare the experimental results against theoretical predictions based on protocol efficiencies and network overhead.
- Investigate potential sources of performance degradation in wireless communication, such as interference, half-duplex operation, and shared medium limitations.

These experiments provide practical insights into the strengths and limitations of both Ethernet and WiFi, crucial for optimizing mixed network performance.

2 METHODOLOGY AND CONCEPTS

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

2.1 Selected Tools

To evaluate the performance of both Ethernet and WiFi connections, we utilized several specialized tools:

- **iperf3**: Used to generate traffic and measure goodput in both TCP and UDP modes. By executing repeated tests, iperf3 provides key metrics such as minimum, maximum, average, and standard deviation of the throughput.
- **Wireshark**: Used to capture and analyze network traffic, Wireshark helped inspect data flows, identify frames, and validate results. It also generated useful charts for analyzing TCP streams.
- **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 ≈ 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 ≈ 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.

---
*The authors collaborated closely in developing this project.
†All the authors are students at Politecnico di Torino, Turin, Italy.

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

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

Table 1: Summary of Hardware and Network Configuration

Connection	Key Specifications	
Ethernet	*Cabling:* CAT.5E	
	Nominal Speed: 1 Gbps	*Protocol:* Ethernet II
	Client ip: 192.168.1.12	***Server ip:*** 192.168.1.13
Wi-Fi	*Standard:* 802.11ax	*Security Protocol:* WPA2-AES
	Nominal Speed: 1200 Mbps	*Frequency:* 5 GHz
	Bandwidth: 80 MHz	*Channel:* 100
	Client ip: 192.168.1.8	***Server ip:*** 192.168.1.4
	Third host ip: 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. 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)

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

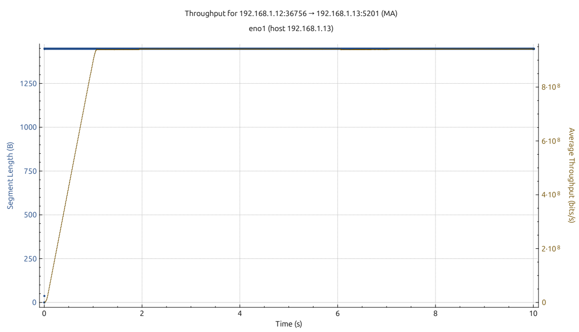

Figure 1: TCP Throughput in the Ethernet Scenario.

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

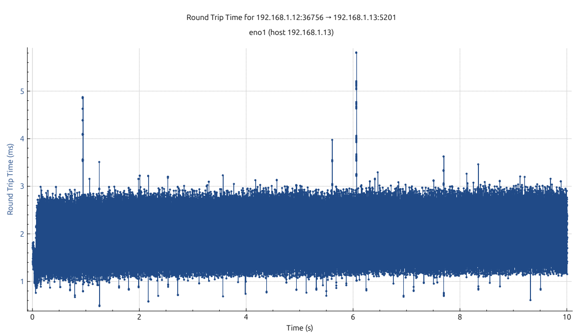

Figure 2: TCP Round Trip Time in the Ethernet Scenario.

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

2. **Both WiFi:**
   In the WiFi scenario, the throughput graph (Fig. 3) shows an initial ramp-up phase during the first 2 seconds, after which the throughput fluctuates around an average value of 434.7 Mbps. These fluctuations suggest that protocol overhead, wireless interference, and the half-duplex nature of WiFi adversely affect performance.
   The round-trip time (RTT) measurements (Fig. 4) reveal RTT values ranging from about 20-50 ms, indicating intermittent delays likely due to congestion and contention in the wireless medium.
   Overall, while the theoretical capacity for TCP over WiFi is estimated to be around 480 Mbps, the experimental data indicate that real-world factors does not reduce that much the effective throughput of the WiFi network. This due to the fact that the network was in an ideal condition, where only the client and the server were connected to the access point, and no other devices were generating traffic.

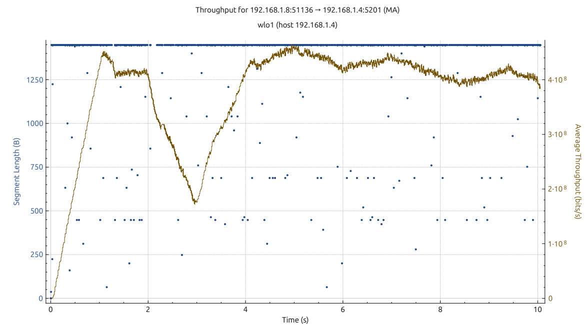

Figure 3: TCP Throughput in the WiFi Scenario.

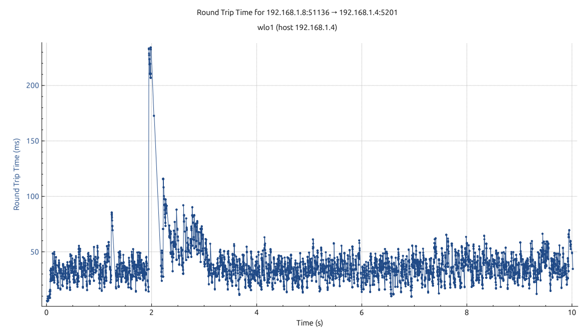

Figure 4: TCP Round Trip Time in the WiFi Scenario.

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

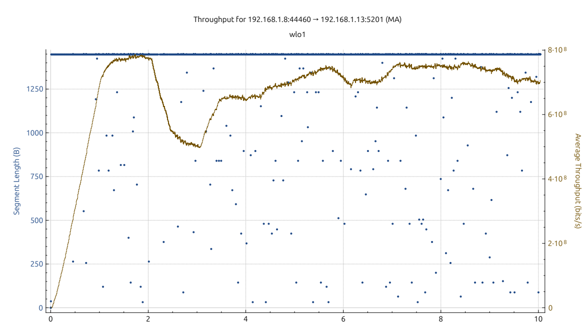

Figure 5: TCP Throughput in the Mixed Ethernet/WiFi Scenario.

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

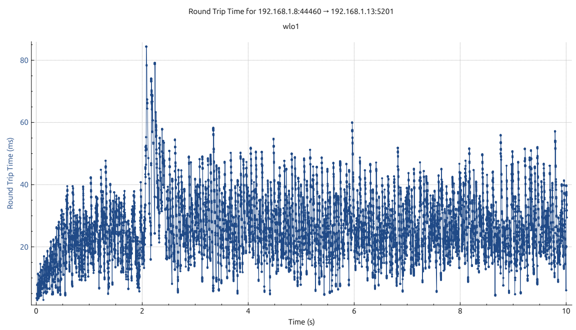

Figure 6: TCP Round Trip Time in the Mixed Ethernet/WiFi Scenario.

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 throughtput, 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 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)

1. **Both Ethernet:**
In the Ethernet scenario, UDP achieves a near-theoretical throughput of **952.8 Mbps** (vs. TCP's 939.6 Mbps), with minimal standard deviation (**1.73 Mbps**). The absence of retransmissions or congestion control allows UDP to utilize the full wired capacity.
While TCP's latency remains marginally lower (1–3 ms RTT) due to acknowledgment-based stability, UDP's lack of overhead enables slightly higher throughput.

2. **Both WiFi:**
UDP averages **487.8 Mbps** (vs. TCP's 434.7 Mbps) in WiFi, achieving a **~12% throughput advantage** by avoiding TCP's congestion control. Despite outperforming TCP, UDP falls short of the **510 Mbps theoretical maximum** due to WiFi interference and contention. Moreover, it shows lower variability (std. dev. **15.8 Mbps** vs. TCP's **22.5 Mbps**), indicating smoother performance. This makes UDP preferable for real-time applications prioritizing speed over error correction.

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

3a. **Shared Capacity:**
In the shared capacity test, UDP throughput drops significantly. The third host's download introduces contention, reducing available bandwidth. UDP's lack of congestion control leads to aggressive transmission attempts, but packet drops result in lower effective throughput. Unlike TCP, which degrades predictably due to its back-off mechanism, UDP's performance becomes more volatile. This highlights TCP's adaptability in shared environments, where fairness and resource allocation are critical, while UDP's rigidity makes it less suitable for contested networks.

5 CONCLUSION

This study evaluated TCP and UDP performance across wired, wireless, and hybrid networks. Results confirm that **Ethernet environments** maximize protocol efficiency, enabling near-theoretical throughput with minimal latency. In contrast, **WiFi introduces variability** due to interference and contention, impacting both protocols despite controlled conditions.
TCP prioritizes reliability, making it robust for file transfers and web traffic, though its congestion control limits performance in dynamic environments. **UDP**, while faster in ideal conditions, struggles with packet loss and shared resources, rendering it less suitable for congested networks.
Hybrid **Ethernet-WiFi setups** highlight the wireless segment as the primary bottleneck, with both protocols constrained by WiFi's instability. Shared capacity scenarios further degrade performance, emphasizing the need for adaptive protocols in contested networks. These findings underscore the importance of aligning protocol choice with environmental constraints. While theoretical models provide benchmarks, real-world performance hinges on interference, medium contention, and protocol design. Network planning must balance throughput, latency, and resilience to dynamic conditions.

A APPENDIX

Server Mode Initialization

```
def run_server():
  """
  Run iperf3 server with clean output handling.
  """
  server_logger.info("Starting␣iperf3␣server...")

  proc = subprocess.Popen(
      ["iperf3", "-s", "-J"],
      stdout=subprocess.PIPE,
      stderr=subprocess.PIPE,
      text=True,
      bufsize=1,
  )
  # Handle server output and errors in a separate
       thread...
```

Listing 1: Excerpt for server mode initialization.

Client Mode Execution and Reporting

```
def run_client(server_ip, udp=False, bitrate="1M",
     iterations=10):
  """
  Run iperf3 client tests and generate reports.
  """
  for i in range(iterations):
      cmd = ["iperf3", "-c", server_ip, "-J", "-t",
           "10", "-i", "1"]
      if udp:
          cmd.extend(["-u", "-b", bitrate])

      result = subprocess.run(
          cmd,
          capture_output=True,
          text=True,
          check=True
      )

      data = json.loads(result.stdout)
      # Extract test data, compute statistics, and
           log results...
```

Listing 2: Excerpt for client mode execution.

Logging and Output Management

```
def setup_logger(log_file, name):
  logger = logging.getLogger(name)
  logger.setLevel(logging.DEBUG)
  handler = logging.FileHandler(log_file)
  formatter = logging.Formatter(
      '%(asctime)s␣-␣%(levelname)s␣-␣%(message)s'
  )
  handler.setFormatter(formatter)
  logger.addHandler(handler)
  return logger
```

Listing 3: Excerpt for logging setup.

These snippets illustrate how the script handles server mode initialization, client tests (both TCP and UDP), and logging. Error handling, multithreaded `stderr` management, and CSV report generation 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

REFERENCES

[1] 2018. IEEE Standard for Ethernet. (2018). https://standards.ieee.org/standard/802_3-2018.html

[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

[3] Realtek Semiconductor Corp. 2023. *Realtek RTL8111/8168/8211/8411: PCI Express Gigabit Ethernet controllers.* https://www.realtek.com

[4] Realtek Semiconductor Corp. 2023. *Realtek RTL8852BE: 802.11ax Wi-Fi 6 PCIe network interface card.* https://www.realtek.com

[5] Intel Corporation. 2023. *Intel Alder Lake-P CNVi: Integrated Wi-Fi 6E solution.* https://www.intel.com/

[6] Python Software Foundation. 2023. *Python: A programming language that lets you work quickly and integrate systems more effectively.* https://www.python.org/

[7] The Wireshark Foundation. 2023. *Wireshark: The world's foremost network protocol analyzer.* https://www.wireshark.org/

[8] The iperf3 Development Team. 2023. *iperf3: A TCP, UDP, and SCTP network bandwidth measurement tool.* https://iperf.fr/

[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

[10] Anker Innovations Limited. 2023. *Anker PowerExpand+ USB-C Hub: A USB-C hub with Ethernet and other interfaces.* https://www.anker.com/

[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/

[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

[13] WDCSecure. 2025. LabWiFi: repository containing the material and the documentation for the WiFi Lab. (2025). https://github.com/WDCSecure/LabWiFi.git

[14] Wikipedia. 2008. Natale a Rio. (2008). https://en.wikipedia.org/wiki/Natale_a_Rio