IT PROJECT
DOCUMENTATION

Iperf3 Network Performance Test – Bandwidth & Latency Analysis

MORGAN BURERA

Google IT Support Professional Certificate

JUNE 2025

Ubuntu, Terminal

Table of Contents

This document is part of a personal project portfolio developed during the Google IT Support Certification. All simulations and analyses were performed in a controlled lab environment. These projects serve as a complement to the course and provide an initial hands-on experience applying its concepts to real-world scenarios

I. Project Overview

This project aimed to simulate a real-world diagnostic scenario using **iperf3**, a powerful network performance measurement tool. The goal was to:

- Measure **bandwidth** and **latency** between a client and server VM
- Generate and analyze traffic under normal and degraded conditions
- Use **Wireshark** to observe packet-level behavior

II. Environment Setup

Component	Details
OS	Ubuntu 22.04 (Virtual Machines)
Network Type	Internal Network (enp0s3 / enp0s8)
Test setup	Client ↔ Server (bidirectional test)
Capture Tools	iperf3, Wireshark, tshark

III. Test Setup and Process

Normal Condition Test

- Started the iperf3 server:

```
iperf3 -s
```

- From the client VM:

```
iperf3 -c [Server_IP] -t 10
```

- Observed results (bandwidth in Mbps, latency via RTT)
- Captured traffic using Wireshark for later analysis.

Degraded Network Simulation

- Introduced network delay and loss:

```
sudo tc qdisc add dev enp0s8 root netem delay 200ms loss 5%
```

- Reran iperf3 test from client
- Captured the degraded traffic using:

```
sudo tshark -i enp0s3 -w degraded_iperf3.pcapng
```

- Restored the interface state:

```
sudo tc qdisc del dev enp0s8 root
```

---

4. Packet Capture and Analysis

Wireshark Filters Used:

`tcp.port == 5201`

Key Findings:

- Normal test showed consistent bandwidth (e.g., ~90 Mbps) and stable TCP handshake.
- Degraded test exhibited increased latency and some TCP retransmissions.
- Packet flow confirmed impact of `tc` netem delay and loss parameters.

V/ Resolution / Outcome

The objective was not to solve a problem but to observe **performance differences** with and without network degradation.

This test environment can be reused for diagnosing n**etwork bottlenecks** or validating **QoS improvements**.

VI/ Key Takeaways & Skills Demonstrated

- Setup and usage of **iperf3** for bandwidth/latency testing
- **Network interface manipulation** using `tc`
- - Real-time and offline **packet analysis** with tshark & Wireshark
- Interpretation of **TCP behavior** under degraded conditions
- Creating reproducible test environments in Linux

VIII/ Optional Enhancements, Reflection

Initially, adding 200ms latency and 5% packet loss using tc did not significantly affect tools like iperf3 -c or curl -I, which led to some confusion. After investigation, I realized:

- ‘iperf3' measures bandwidth, not latency by default — the impact is more visible with UDP and low bandwidth constraints.
- ‘curl -I’ requests are too lightweight and optimized to reflect small network degradations.
- tc must be applied to the correct network interface (e.g., the one used for internet routing) — otherwise, it has no effect.

To better assess the impact of network impairments, I adjusted the tools and parameters used (e.g., added curl -w for detailed timing and used ping for latency visibility). This helped me later to verify the tc configuration effectively.

Video Recording Link

https://youtu.be/XpY0JDUqqZI