# **Related Work Survey and Project Planning Report (A4b)**

### **1. Scope of Document**

* Briefly introduce the scope of the report, detailing that it covers the **Related Work Survey** and **Project Planning** for the Telemetry-Driven Load Balancer for Software-Defined Networks.

The purpose of this report is to concisely detail the existing work related to our proposed project and to provide a clear timeline of our project implementation. The *related work survey* portion of this report will specifically look at existing projects that have attempted a similar approach to load balancing; that is, a non-commercial, non-proprietary prototype load balancer that focuses on balancing strengths of traffic paths. Further, this portion will distinguish our proposed project from the existing work. The *project planning* portion will detail our implementation timeline by choosing achievable and sensible milestones for building our telemetry-driven load balancer, along with the relative time frame they will be achieved in.

### **2. Overview of Project**

* **Objective**: This project’s objective is to successfully design a dynamic, cost-effective load balancer using real-time network telemetry for SDN environments.
* **Functionality**: The main function of the proposed telemetry-driven load balancer is to distribute traffic evenly across network paths based on real-time conditions.
* **Goal**: Our project goal is to demonstrate the feasibility of a resource-efficient, scalable load balancer leveraging existing cloud-based infrastructure.

### **Part 1: Related Work Survey**

### **3. Introduction to Related Work**

* Introduce the **related work survey** by describing the general importance of load balancing in SDN, telemetry systems, and how real-time traffic management has evolved.

Load balancing is a crucial element in the performance, reliability, and efficiency of any well-designed Software-Defined Network (SDN). Its primary purpose is to optimize network resources by distributing traffic evenly, preventing certain data paths or devices from becoming overloaded while others remain underutilized. As networks have grown more complex, the methods of traffic management have evolved dramatically. Early approaches were largely static, manual, and reactive; once traffic routes were configured, they remained unchanged unless manually adjusted by administrators. However, as network demands increased, dynamic routing protocols were introduced, allowing traffic to be automatically rerouted based on network conditions without the need for human intervention.   
Today, real-time traffic management has become far more sophisticated. Modern systems often rely on AI-driven algorithms that make intelligent, real-time decisions based on current traffic patterns and network health data. This evolution towards automated, adaptive traffic management aligns with the goals of our project: rather than developing a traditional load balancer, we are proposing a telemetry-driven prototype that adjusts traffic paths dynamically based on real-time flow statistics collected from Open vSwitch (OVS) switches. By leveraging these real-time data streams, our system aims to ensure optimal load distribution across the network.

In the context of modern SDN environments, several key technologies and algorithms play a critical role in enabling effective load balancing and real-time traffic management. OpenFlow (an early and widely-adopted protocol for SDN), allows the SDN controller to communicate directly with the forwarding plane of network devices. It provides the foundation for centralized traffic control, enabling flexible routing decisions based on real-time conditions.   
Tools like GoFlow2 and Kafka are essential for collecting, processing, and analyzing network telemetry data. GoFlow2 is used to extract detailed flow statistics from Open vSwitch (OVS) instances, providing a granular view of traffic patterns across the network. This real-time data is then streamed into Kafka, a distributed streaming platform that processes and aggregates the telemetry information, making it accessible for decision-making in load balancing.

As we build upon this architecture, our project aims to leverage these established tools to enhance real-time telemetry-driven load balancing. By integrating this telemetry data into our system, we seek to improve traffic distribution efficiency in complex network environments. However, to fully understand the landscape, it's important to review existing work in this space and differentiate our approach from related studies.

### **4. Review of Related Works**

* **Research Item 1**
  + Write a paragraph on this source’s approach, significant results, limitations, and possible improvements. Cite the source within this paragraph.
* **Research Item 2**
  + Write a paragraph on this source’s approach, significant results, limitations, and possible improvements. Cite the source within this paragraph.
* **Research Item 3**
  + Write a paragraph on this source’s approach, significant results, limitations, and possible improvements. Cite the source within this paragraph.

### **5. Comparison and Conclusion**

* Summarize how the **proposed project** improves upon existing work.
* Highlight how **real-time telemetry** and a focus on **scalability** address gaps in current solutions.
* State how the project will provide a **cost-effective** and **resource-efficient** alternative, improving traffic management for resource-constrained environments.

### **Part 2: Project Planning**

### **6. Work Packets (WP)**

* **WP-1: Project Management**: Assign project lead and timeline monitoring responsibilities.
* **WP-2: GUI Development**: Develop and test the front-end interface for managing traffic flows and telemetry data visualization.
* **WP-3: Application Software Development**: Write load balancer algorithms using Go, integrate Kafka, and configure OVS with OpenFlow.
* **WP-4: Hardware Integration**: Test load balancer with SDN components in a simulated network environment.
* **WP-5: Testing and Validation**: Implement testing scenarios using **Mininet** to validate traffic flow efficiency and performance.

### **7. Gantt Chart / Timeline**

* Break down tasks, deliverables, and milestone (the following was copied from the presentation, this can be changed as needed):
  + **Milestone 1**: Literature Review & Technology Selection.
  + **Milestone 2**: System Architecture Design.
  + **Milestone 3**: Telemetry Data Collection & Processing.
  + **Milestone 4**: Load Balancing Algorithm Development.
  + **Milestone 5**: Integration & Testing.
  + **Milestone 6**: Performance Evaluation.
  + **Milestone 7**: Refinement & Optimization.
  + **Milestone 8**: Final Prototype Demonstration.

### **8. Team Member Responsibilities**

* Define the **contribution** of each team member to the Work Packets (WP).
  + Assign roles based on expertise: software development, testing, front-end, system integration, and management.

### **Conclusion**

* Recap the **related work** that the project builds upon, highlighting key differences.
* Summarize the **project plan**, emphasizing clear roles, deliverables, and a structured timeline.