SUCCESS-6G: VERIFY

WP5 Deliverable E14

Project Title:	SUCCESS-6G: VERIFY
Title of Deliverable:	Final testing and validation of service KPIs
Status-Version:	v1.0
Delivery Date:	30/04/2025
Contributors:	Allen Abishek, Ricard Vilalta, Raul Muñoz (CTTC), Miguel Fornell, Francisco Paredes (Idneo), Angelos Antonopoulos (Nearby Computing)
Lead editor:	CTTC
Reviewers:	Charalampos Kalalas (CTTC), Francisco Paredes (Idneo)
Keywords:	Real-time location awareness; end-to-end latency; closest node selection

Document revision history

Version	Date	Description of change
v0.1	20/02/25	Table of Contents (ToC)
v0.2	28/02/25	Content added
v0.3	24/03/25	Additional inputs
v0.4	16/04/25	Final inputs and revision
v1.0	30/04/25	Final version uploaded to the website

Acknowledgment

The research conducted by SUCCESS-6G - TSI-063000-2021-39/40/41 receives funding from the Ministerio de Asuntos Económicos y Transformación Digital and the European Union-NextGenerationEU under the framework of the "Plan de Recuperación, Transformación y Resiliencia" and the "Mecanismo de Recuperación y Resiliencia".

[1] http://creativecommons.org/licenses/by-nc-nd/3.0/deed.en_US

Executive Summary

The computational overhead of over-the-air (OTA) software updates presents challenges in optimizing resource utilization and minimizing network congestion. SUCCESS-6G-VERIFY addresses these challenges by leveraging MEC-based workload distribution, AI-driven computation management, and predictive resource allocation. By intelligently offloading processing tasks to edge nodes and optimizing data caching strategies, the system significantly reduces update latency and improves overall network efficiency. Experimental validations highlight improvements in processing time, bandwidth allocation efficiency, and energy consumption, demonstrating that the SUCCESS-6G-VERIFY framework is well-equipped to support scalable and computationally optimized OTA software update mechanisms in connected vehicle ecosystems.

Table of Contents

List of Figures

1 Introduction

The increasing reliance on software-driven functionalities in modern vehicles necessitates a robust and efficient method for delivering over-the-air (OTA) software updates. As connected and autonomous vehicle ecosystems continue to evolve, manufacturers and service providers must ensure that software updates are timely, computationally optimized, and minimally disruptive to vehicular operations. Vehicle-to-Everything (V2X) communication is a key enabler of this transformation, facilitating seamless and reliable OTA updates while incorporating advanced computing mechanisms to enhance performance. However, reducing computational overhead, optimizing data transfer, and ensuring efficient network resource utilization remain critical challenges. The SUCCESS-6G-VERIFY project aims to address these challenges through the integration of Software-Defined Networking (SDN), Multi-access Edge Computing (MEC), and AI-driven computation management.

A fundamental requirement for efficient OTA software updates is leveraging distributed computing resources to offload processing from centralized cloud systems. Cellular V2X (C-V2X) technology, enabled by 5G and MEC, enhances the computational efficiency of OTA update delivery by optimizing data caching, bandwidth allocation, and real-time processing at edge nodes. The SUCCESS-6G-VERIFY framework integrates AI-driven workload distribution and dynamic resource allocation to manage the computational complexity of large-scale software updates. Additionally, containerized deployment models and microservices-based architectures ensure seamless update provisioning with minimal latency. Beyond connectivity, computational efficiency is a primary concern for OTA software updates. The high frequency of updates, combined with varying network conditions, necessitates intelligent workload balancing and predictive resource allocation. SUCCESS-6G-VERIFY employs orchestration mechanisms that dynamically adjust computational resources based on real-time vehicular demand and network load. Through federated learning and distributed intelligence, the system reduces processing delays while maintaining service reliability and scalability.

Efficient computation management also plays a key role in reducing network congestion and improving overall vehicular performance. MEC-based resource scheduling minimizes data redundancy and accelerates update deployment by executing processing tasks closer to end-users. SUCCESS-6G-VERIFY integrates intelligent caching strategies and traffic-aware optimization to enhance update success rates while minimizing system downtime. These mechanisms ensure that vehicles receive software updates with optimal efficiency, reducing unnecessary data transfers and computational strain on network infrastructure.

This deliverable presents results on the implementation and validation of efficient OTA software updates within a V2X connectivity framework. Experimental evaluations conducted on the ADRENALINE testbed demonstrate significant improvements in computational efficiency, network utilization, and latency reduction. The integration of SDN, MEC, and AI-driven workload management has resulted in a scalable and adaptive solution capable of addressing the evolving computational demands of connected vehicle ecosystems. The findings from this research highlight the potential of SUCCESS-6G-VERIFY in revolutionizing vehicular software update methodologies. By leveraging cutting-edge computing and network optimization techniques, the proposed framework ensures that vehicles remain updated with minimal resource consumption and operational disruptions. As the automotive industry continues to transition towards fully connected and autonomous systems, the implementation of computationally efficient OTA update mechanisms will be instrumental in enhancing vehicle performance, service reliability, and regulatory compliance.

The subsequent sections present the specific methodologies employed, experimental setup, and detailed performance evaluations of the proposed efficient OTA update system, providing a comprehensive analysis of its benefits and potential industry applications.

2 Use case 2: Automated software updates for vehicles

2.1 General description and overall objectives

Over-the-air software updates are delivered remotely from a cloud-based server, through a cellular connection, to the connected vehicle with the aim of providing new features and updates to the vehicle's software systems. Such software updates may include changes to any software that controls the vehicle's physical parts or electronic signal processing system. In practice, the updates often tend to apply more to user interfaces like infotainment screens and navigation (i.e., vehicle maps). The update procedure, when performed over the air, enables a vehicle's performance and features to be continuously up-to-date and improved. The integration of advanced data analytics, automated and remote service delivery eliminates the need for visiting repair/service centres, while technological advancements in these updates give vehicle manufacturers the freedom to constantly "freshen up" finished products remotely. C-V2X technology plays a crucial role in the update process, enabling efficient, scalable, and seamless wireless communication between vehicles and software management platforms. Figure 1 illustrates the implementation phases for this use case.

Figure 1 Implementation phases for the automated software updates

The overall **objectives** of this use case can be summarized as follows:

- Safer and more entertaining driving experience.
- Hardware and software components maintained and updated regularly during a vehicle's lifespan, implying a slower rate of depreciation.
- Prevention of cyberattacks targeting outdated software.
- Compliance to new rules and standards.
- Lower repair costs and elimination of labour charges.
- Lower warranty costs for manufacturers and lower downtime for customers

The key **stakeholders** involved in the use case are:

- The ***Mobile Network Operator (MNO)***, providing wireless connectivity between the vehicle, the edge computing infrastructure, and the vehicular software management system. The MNO is interested in optimizing the network operation by enhancing its energy efficiency and coverage, while offering novel services to accommodate more users.
- The ***edge infrastructure provider***, offering and managing computational resources at the edge and supporting real-time services as well as virtualized network functions and AI-empowered algorithms for advanced computational tasks.
- The ***equipment provider,*** providing in-vehicle embedded devices, e.g., hardware components and sensor devices, that can be remotely reconfigured and updated.
- The ***vehicular software management system,*** operated by the equipment provider or vehicle manufacturer, is responsible for issuing periodically new software updates.
- The ***software developers***, devising and applying data-processing modules for automated update of vehicular components' software.
- The ***cloud providers*** can optionally be involved, offering additional computational resources to host the service.

Note that, without loss of generality, some stakeholders may assume multiple roles or, equally, some roles may be assumed by multiple stakeholders. For instance, the MNO could also be the owner of the edge infrastructure, or an equipment provider may also be responsible for the operation of the vehicular software management system or outsource it to a third party.

2.2 User story 2.3: Over-the-air vehicular software updates with efficient computation

Over-the-air software updates would substantially benefit from the realization of computationally efficient mechanisms to dramatically reduce data processing times. By leveraging edge-processing capabilities in conjunction with distributed learning techniques, the associated computational overhead is expected to be relieved. Additionally, zero-touch orchestration of container-based solutions will ensure the automatic reconfiguration of vehicular on-board units with an efficient usage of computational resources, while adapting to V2X network topology changes. This would require agile management of the edge-cloud continuum to guarantee seamless service delivery.

2.3 Overall UC2 architecture and network deployments

The elaboration of Figure 2 details a system architecture specifically designed for Over-the-air (OTA) Software Updates, integral to the SUCCESS-6G framework. This architecture addresses the complex requirements of Use Case 2. Figure 2 Proposed overall UC2 system architectureprovides a high-level system architecture for OTA vehicular software updates within a robust V2X connectivity framework, leveraging ETSI TeraFlowSDN for network automation and control. The figure illustrates the key components enabling software update dissemination to connected vehicles via 5G mobile edge computing (MEC) nodes.

At the core of this system is the ETSI TeraFlowSDN Controller, which manages the network infrastructure, including the gNBs (5G base stations) and Transport Network. The NFV Orchestrator (NFV-O) enables dynamic deployment and scaling of virtualized network functions, such as Distributed User Plane Functions (D-UPF) within MEC nodes.

Each edge node (Edge Node 1 & Edge Node 2) hosts a Software Update Server, responsible for caching and distributing updates to C-V2X On-Board Units (OBU) in connected vehicles. These updates are delivered via the 5G network, passing through the transport network, controlled by the TeraFlowSDN controller.

To ensure security and integrity, the system integrates a Security-as-a-Service module, providing firewall protection and secure communications for software updates. The updates originate from local cloud infrastructure, which includes 5G Core Control Plane components such as SMF (Session Management Function), AMF (Access and Mobility Management Function), and UPF (User Plane Function).

The software update client within the vehicle's C-V2X OBU interacts with the Software Update Servers over the network, ensuring efficient and timely delivery of critical updates for vehicle applications.

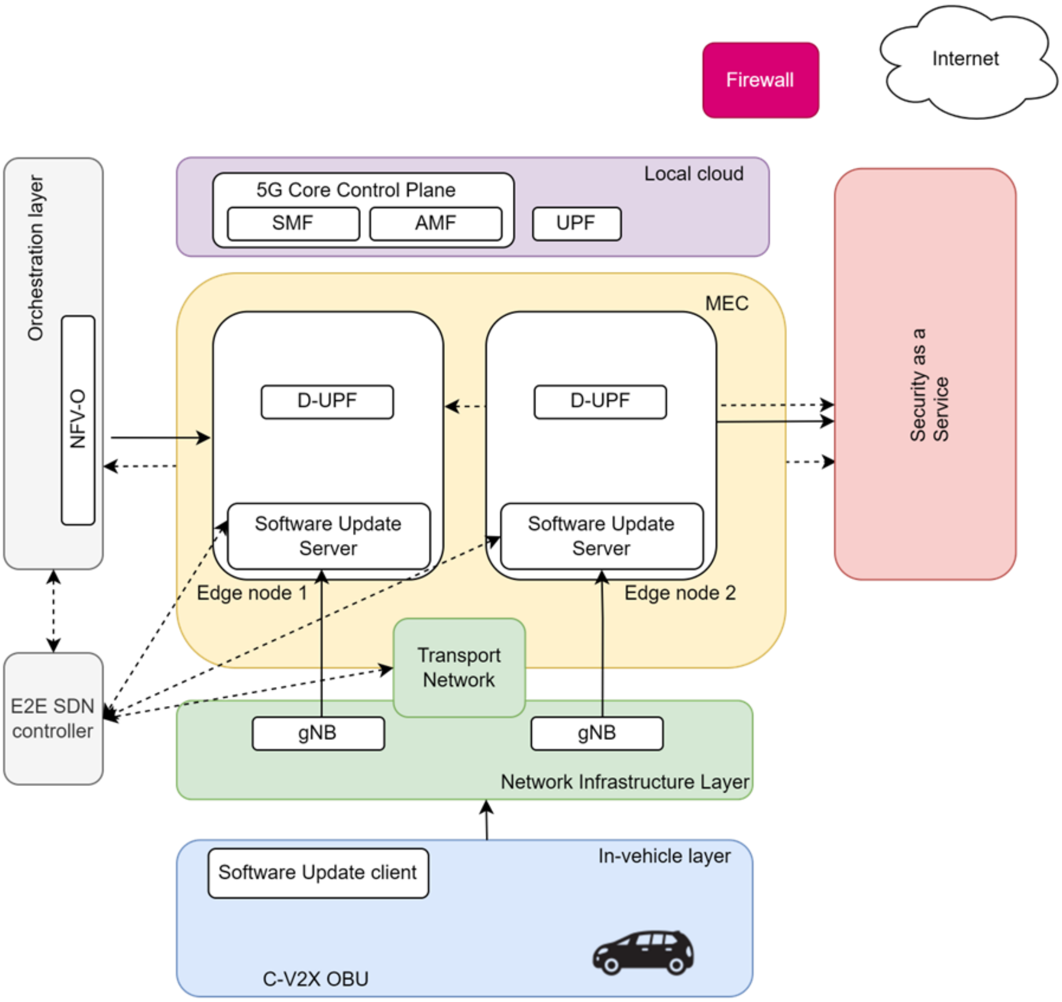

Figure 2 Proposed overall UC2 system architecture

This architecture highlights the interplay between 5G, MEC, SDN, and V2X technologies to facilitate secure and efficient OTA software updates, enabling reliable vehicle connectivity and automation.

2.4 Facilities for Use Case 2: ADRENALINE Testbed

The ADRENALINE testbed® is an open and disaggregated SDN/NFV-enabled packet/optical transport network and edge/core cloud infrastructure for 6G, IoT/V2X and AI/ML services, constantly evolving since its creation in 2002, and reproducing operators' networks from an End to End (E2E) perspective and Data Centre Interconnect (DCI). The figure below summarizes the networking scenario of the ADRENALINE testbed, to be used for the execution of SUCCESS-6G.

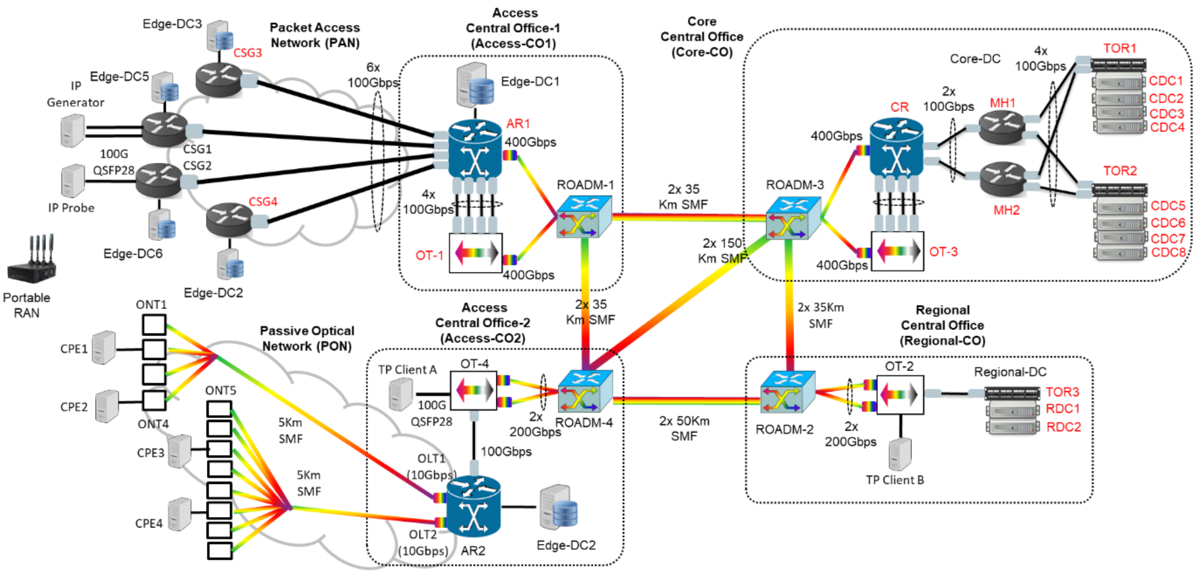

Figure 3 ADRENALINE testbed to be used for Use Case 2

ADRENALINE spans the access, aggregation-metro and core segments, and includes distributed Data Centres (DCs) geographically disperse and located at the edge or in central locations. As depicted in the figure, the key elements are: (1) an SDN-controlled optical network (flexi-grid DWDM photonic mesh), with 4 ROADM nodes and over 600km of amplified DWDM links. Currently, all the links of the mesh are based on amplified C-band transmission, but one of them also supports amplified flexible L-band transmission; (2) packet-optical nodes with optical pluggable transceivers, providing aggregated 400G data rates (muxponders) for transporting traffic flows between the access networks and the core central offices or data centers; (3) programmable SDN-enabled S-BVTs able to transmit multiple flows at variable data rate/reach up to 1 Tb/s; (4) a Packet Access Network (PAN) connected to the metro infrastructure with IP Cell Site Gateways (CSGs); (5) a PON tree formed by disaggregated Optical Network Terminals (ONTs), offering connectivity to several Customer Premises Equipment (CPEs). ADRENALINE also includes a Portable 5G RAN platform for testing and validation of 5G and beyond use cases. The different access networks (i.e., PON) and the photonic mesh are managed by dedicated orchestrators and controllers (e.g., CTTC FlexOpt Optical Controller) to automatically handle the connectivity services entailing the de-/allocation of heterogeneous network resources (i.e., packet and optical devices). The *domain-specific* controllers and orchestrators are coordinated hierarchically by the ETSI TeraFlowSDN controller, which exposes a North Bound Interface to allow interaction of resources to request network connectivity services. This service platform orchestrates the transport (optical/packet) and computing:

i) Multi-VIM (virtualized infrastructure managers) combining OpenStack and K8s controllers for virtual machines and containers;

ii) TeraFlowSDN controller for E2E connectivity among virtual machines, containers, and end-points. The service platform is also in charge of managing the life-cycle of network services and network slices: i) a network service is composed of chained NFs;

iii) a network slice is composed of one or several concatenated network services that deploy a set of NFs.

3 Over-the-air vehicular software updates with efficient computation: Implementation at the ADRENALINE testbed

3.1 MEC Bandwidth Management

In the field of vehicular technology, the integration of Multi-Access Edge Computing (MEC) with advanced network control and management, such as TeraFlowSDN (TFS), presents a formidable solution for OTA Software Updates. MEC offers ultra-low latency and high bandwidth, along with real-time access to data and radio network information, which are crucial in swiftly and efficiently managing OTA updates.

The MEC BandWidth Management (BWM) service plays a pivotal role in allocating and adjusting bandwidth resources, including bandwidth size and priority, specifically tailored for MEC applications. This feature enables MEC applications to specify bandwidth requirements, essential for the smooth transmission of OTA updates.

This demonstration examines the synergistic relationship between the BWM service and TeraFlowSDN (TFS) in optimizing resource allocation for OTA software updates in vehicular networks. The BWM service empowers applications to designate specific bandwidth allocations and other quality of service constraints, such as latency, crucial for the timely and effective delivery of OTA updates. Concurrently, TFS provides sophisticated orchestration of traffic flow management and control, ensuring that OTA update traffic is given priority.

Exploring the benefits of MEC BWM and TFS within the context of OTA updates reveals several advantages. Improved update efficiency, reduced network congestion, and enhanced scalability are prominent among these. The prioritization of OTA update traffic, facilitated by BWM and TFS, is instrumental in reducing latency, thereby streamlining the update process. Allocating dedicated bandwidth to OTA updates also alleviates network congestion, ensuring a more efficient update process for all connected vehicles.

Moreover, the scalability offered by MEC BWM and TFS is critical in adapting to the rapidly evolving field of vehicular technology and the increasing volume of OTA updates. As vehicles become more connected and reliant on software, the need for robust solutions that can adapt to growing demands becomes paramount.

In summary, the integration of MEC BWM and TFS emerges as a significant enabler for network operators aiming to deliver seamless and efficient OTA software updates to vehicles. This demonstration unveils the dynamics and potential of this integration, highlighting its capacity to transform the landscape of vehicular network management.

3.2 Requirements and KPI

As the automotive industry progresses toward increasingly connected and software-driven vehicles, robust mechanisms for efficient bandwidth management and traffic orchestration become paramount. OTA updates must be delivered swiftly, reliably, and securely to ensure that vehicles remain functional and protected against emerging threats. This section establishes a series of requirements designed to optimize bandwidth usage, integrate advanced networking technologies, and ensure the scalability needed to accommodate growing demands. From leveraging MEC for ultra-low latency to integrating TeraFlowSDN (TFS) for intelligent traffic control, each requirement addresses a critical facet of managing and distributing OTA updates effectively. By fulfilling these requirements, automotive networks can maintain high performance even under increased load, ensuring seamless, real-time updates for a rapidly expanding fleet of connected vehicles.

3.2.1 Requirements

3.2.1.1 MEC Bandwidth Management

The system should integrate MEC for efficient bandwidth management. This involves allocating and adjusting bandwidth resources dynamically to ensure that OTA software updates are transmitted smoothly and efficiently. MEC provides ultra-low latency and high-bandwidth access to vehicles, optimizing resource utilization in vehicular networks.

The rationale for MEC Bandwidth Management is to ensure seamless and timely software updates in connected vehicles. As vehicles increasingly rely on software for performance and security features, the ability to provide updates efficiently with minimal network congestion becomes critical. MEC reduces latency and enhances update reliability by processing data closer to end-users, reducing the load on centralized data centers.

3.2.1.2 TeraFlowSDN (TFS) Integration

The system should leverage TFS to orchestrate traffic flow and prioritize OTA updates. TFS enables dynamic resource allocation and traffic engineering, ensuring that software updates receive preferential treatment within the network while maintaining overall network stability.

The rationale for integrating TFS is to provide a sophisticated network control and management layer that optimizes network performance. By prioritizing OTA traffic and dynamically adjusting bandwidth allocations, TFS prevents network congestion and guarantees high-speed, reliable software distribution to connected vehicles.

3.2.1.3 Bandwidth Reservation for OTA Updates

The system must enable OTA update applications to request specific bandwidth allocations and quality of service constraints. This ensures that update transmissions are not interrupted due to competing network traffic, particularly in high-load scenarios.

The rationale for bandwidth reservation is to ensure the availability of dedicated network resources for critical OTA updates. Without bandwidth allocation, software updates might experience delays, affecting vehicle security and performance. By implementing reservation mechanisms, the system enhances reliability and predictability of updates.

3.2.1.4 Scalability of OTA Update Distribution

The system must support the growing number of connected vehicles by dynamically adapting to increasing demands. This includes optimizing network and computational resources at MEC nodes to handle simultaneous OTA update requests efficiently.

The rationale for scalability is to accommodate the rapid expansion of vehicular networks. As more vehicles require frequent software updates, network operators must ensure that OTA systems can scale to meet the rising demand while maintaining service quality and minimizing latency.

3.2.1.5 Network Congestion Control

The system should implement congestion management mechanisms to prevent excessive network load, which could disrupt update delivery. MEC and TFS should dynamically adjust traffic flow to prevent bottlenecks.

The rationale for congestion control is to maintain overall network stability. Without proper traffic engineering, excessive OTA update traffic can degrade network performance for other critical applications. Intelligent congestion management ensures that all services operate efficiently.

3.2.2 Key Performance Indicators (KPIs)

3.2.2.1 Bandwidth Allocation Efficiency (%)

This KPI measures the efficiency of bandwidth usage in delivering OTA updates, ensuring optimal resource utilization within MEC networks.

Efficient bandwidth allocation ensures that network resources are optimally utilized while preventing excessive congestion. Higher efficiency translates to smoother OTA updates and better network performance.

Target Value: The system should maintain at least 90% bandwidth allocation efficiency for OTA updates.

3.2.2.2 End-to-End Update Latency (ms)

This KPI measures the total time taken for an OTA update to reach its destination from the MEC server to the vehicle.

Minimizing latency is crucial to ensuring timely updates, particularly for security patches. Delayed updates can leave vehicles vulnerable to cybersecurity threats or outdated functionalities.

Target Value: The system should deliver OTA updates within 300 ms in optimal network conditions.

3.2.2.3 Network Congestion Avoidance Rate (%)

This KPI evaluates the effectiveness of congestion control mechanisms in preventing excessive traffic buildup during OTA update distribution.

Avoiding network congestion ensures that critical vehicular communication is not disrupted. A higher avoidance rate indicates an efficient traffic management system that prioritizes essential updates while maintaining network stability.

Target Value: The system should achieve at least a 95% congestion avoidance rate.

3.2.2.4 Update Success Rate (%)

This KPI measures the percentage of successfully delivered and installed OTA updates without failures or retransmissions.

A high success rate ensures that vehicles receive updates without disruptions. Failures in OTA updates can compromise vehicle performance and require costly manual interventions.

Target Value: The system should maintain an OTA update success rate of at least 99%.

3.3 UC2 Architecture and network deployment

Figure 4 illustrates a multi-layered architecture for enabling secure and efficient software updates in a 5G-enabled Connected Vehicle-to-Everything (C-V2X) environment. The system integrates Multi-access Edge Computing (MEC), Software-Defined Networking (SDN), and Network Function Virtualization (NFV) to optimize software updates and network resource management. Various components interact across different network layers to facilitate low-latency, secure, and efficient software updates for in-vehicle systems.

The E2E SDN Controller, located in the orchestration layer, is responsible for managing the end-to-end transport network and traffic routing. This component ensures dynamic and programmable network management, allowing efficient routing of software updates from Software Update Servers to the in-vehicle system. Through SDN principles, this controller provides centralized visibility and control, optimizing network paths and prioritizing update traffic based on real-time network conditions and Quality of Service (QoS) policies. By interacting with the NFV Orchestrator (NFV-O), it ensures network resources are allocated dynamically, improving network efficiency and scalability.

Two Software Update Servers, represented in dark blue, are deployed at edge nodes within the MEC infrastructure. These servers play a critical role in reducing software update latency by caching and distributing updates closer to vehicles, rather than relying on centralized cloud servers. The use of Distributed User Plane Functions (D-UPFs) in each edge node enables localized data processing, minimizing backhaul traffic to the 5G Core Network.

These servers handle OTA software updates for vehicles, ensuring real-time updates for safety-critical systems such as autonomous driving algorithms, infotainment software, and cybersecurity patches. The interaction between D-UPFs and Software Update Servers optimizes traffic flow, ensuring fast and secure update delivery without overloading the core network.

The Transport Network, centrally placed in the figure and highlighted in dark blue, serves as the data transmission backbone for distributing software updates. It connects the edge nodes, 5G base stations (gNBs), and in-vehicle systems, ensuring seamless data transfer. The E2E SDN Controller dynamically manages this network, optimizing bandwidth allocation and ensuring updates are delivered efficiently. This transport layer is crucial for maintaining low-latency connectivity between software update servers and C-V2X onboard units (OBUs).

The Software Update Client, highlighted in dark blue, resides in the In-Vehicle Layer and represents the OBU in the connected vehicle. This client is responsible for receiving, validating, and installing software updates sent through the transport network from the Software Update Servers. The client ensures updates are properly authenticated and installed without disrupting the vehicle's real-time operations. By leveraging secure communication protocols and MEC-based caching, the update process is optimized to be fast, reliable, and resilient to network disruptions.

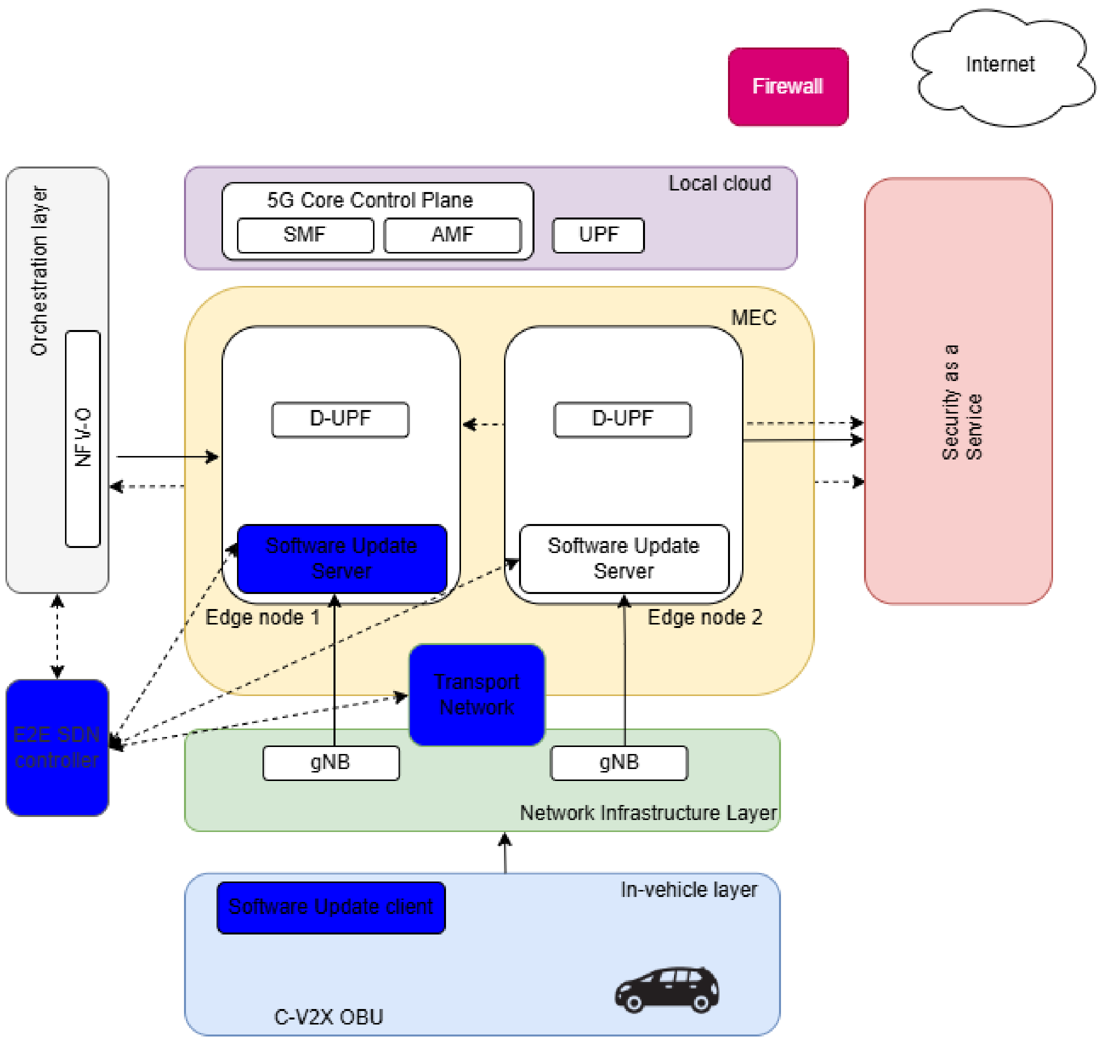

Figure 4 Instantiation of SUCCESS-6G architecture for OTA vehicular software updates with efficient computation

3.4 Exposed interfaces

The ETSI GS MEC 015 Bandwidth Management API[2] is a standardized API specification designed for Multi-access Edge Computing (MEC) environments. It defines a structured mechanism for managing bandwidth allocations, ensuring efficient data traffic handling across MEC applications. The API is built using OpenAPI 3.1.0, providing a well-documented, machine-readable format for service interoperability. The latest version of the API is 2.2.1, and it follows the BSD-3-Clause license, making it accessible for open-source and commercial implementations.

The API supports interactions over the server endpoint https://localhost/bwm/v1, allowing applications to allocate, retrieve, update, and remove bandwidth resources dynamically. It provides a RESTful interface for MEC applications to request bandwidth for specific sessions or application instances, ensuring optimized network resource utilization. The API documentation is complemented by ETSI GS MEC 015 V2.2.1 Traffic Management APIs, which provide additional details on its operational framework.

The API exposes two key resource endpoints for bandwidth allocation management:

- /bw_allocations (Collection Resource)
  - GET: Retrieves a list of bandwidth allocation resources, allowing applications to fetch configured bandwidth reservations.
  - POST: Creates a new bandwidth allocation resource, enabling MEC applications to request dedicated bandwidth.
- /bw_allocations/{allocationId} (Instance Resource)
  - GET: Fetches details of a specific bandwidth allocation instance.
  - PUT: Updates the entire resource, replacing the previous configuration.
  - PATCH: Partially modifies an existing allocation by applying deltas (incremental updates).
  - DELETE: Removes an existing bandwidth allocation, unregistering it from the MEC system.

Each request method follows the RESTful principles, ensuring standardized request-response interactions with HTTP status codes for error handling and success confirmation.

The core data model revolves around the BwInfo schema, which defines how bandwidth allocation requests and responses are structured. The schema includes fields such as:

- allocationId: Unique identifier for the bandwidth allocation instance.
- appName: Name of the application making the request.
- allocationDirection: Defines whether the bandwidth is downlink, uplink, or symmetrical.
- appInsId: Identifies the specific application instance requesting bandwidth.
- fixedAllocation: Specifies the bandwidth allocation in bits per second (bps).
- fixedBWPriority: Sets allocation priority when multiple applications or sessions compete for bandwidth.
- requestType: Distinguishes between application-specific and session-specific bandwidth allocations.

Additionally, the API provides filtering options through query parameters like:

- app_instance_id: Filters allocations for a specific application instance.
- app_name: Retrieves allocations based on application name.

[2] https://forge.etsi.org/rep/mec/gs015-bandwith-mgmt-api/-/raw/master/BwManagementApi.yaml?ref_type=heads

- session_id: Fetches allocations related to a given session.

For updates via PATCH, the BwInfoDeltas schema is used, containing only the fields that need modification, ensuring minimal impact on existing configurations.

3.5 Workflow

Figure 5 offers a sophisticated and scholarly representation of the orchestrated network operations within an End-to-End (E2E) Software-Defined Network (SDN) environment. This diagram serves not merely as a visual aid but as an academic illustration of the intricate interplay among various controllers, infrastructure components, and end-user interactions. The processes depicted in this figure are emblematic of the advanced capabilities and integrations within modern network architectures.

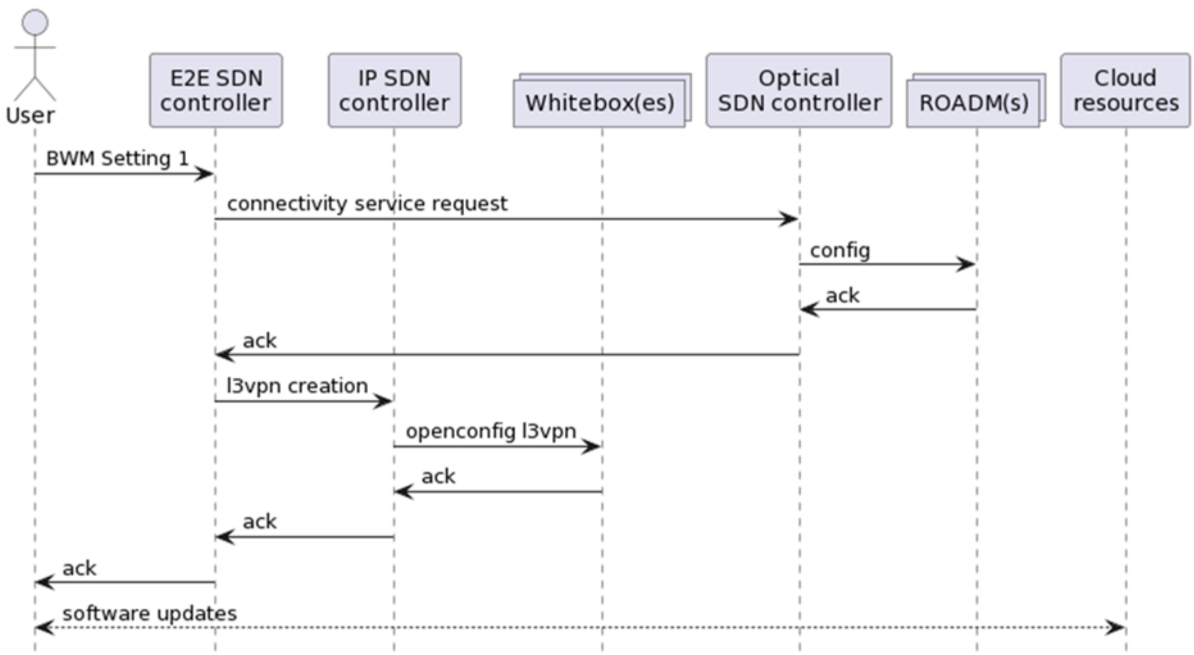

Figure 5 Sequence diagram for OTA vehicular software updates with efficient computation

The diagram's narrative commences with the user, denoted as "edge1", initiating a bandwidth setting, referred to as BWM Setting 1. This action triggers a critical communication sequence with the E2E SDN controller, a pivotal entity within the network framework. The E2E SDN controller, embodying advanced computational capabilities, processes this request, engaging in a nuanced dialogue with the Optical SDN controller. This interaction is vital for the fulfillment of a connectivity service request, an operation that is foundational to network performance and efficiency.

The request encompasses the configuration of pivotal optical network infrastructure components, namely Whiteboxes and Reconfigurable Optical Add-Drop Multiplexers (ROADMs). These components play an instrumental role in the optical networking realm, facilitating efficient data transmission and network flexibility. The sequence diagram meticulously illustrates the exchange of acknowledgments between the Optical SDN controller and the ROADMs, a process that guarantees the successful configuration and establishment of connectivity. Subsequently, the Optical SDN controller communicates the acknowledgment back to the E2E SDN controller, completing this segment of the orchestration.

In parallel, a sophisticated integration process unfolds involving the E2E SDN controller and the IP SDN controller. The E2E SDN controller, in this context, initiates the creation of a Layer 3 Virtual Private

Network (L3VPN). This is achieved through a communicative interface with the IP SDN controller. The IP SDN controller, operating under the OpenConfig standard, interacts with Whiteboxes to configure the L3VPN. This interaction highlights the seamless integration of different network layers and the versatility of the controllers in managing diverse network functions.

The diagram further delineates the exchange of acknowledgments between the Whiteboxes and the IP SDN controller, an essential step in confirming the successful execution of the L3VPN creation. This successful interaction is communicated back to the E2E SDN controller, which, in an emblematic demonstration of the integrated network architecture, acknowledges the User's initial request.

The concluding segment of the sequence diagram emphasizes the dynamic interaction between the User and Cloud resources. This interaction is particularly significant in the context of software maintenance and updates. The User engages in a bidirectional dialogue with the Cloud, reflecting the ongoing need for software updates and the dynamic nature of cloud-based resource management. This interaction underscores the importance of cloud resources in contemporary network environments, particularly in the realm of software deployment and updates.

3.6 Preliminary experimental validation of the functionalities

The integration of the ETSI GS MEC 015 Bandwidth Management API and ETSI TeraFlowSDN (TFS) Controller presents a promising approach to enhancing network resource allocation and optimizing software-defined traffic orchestration in vehicular networks. These technologies aim to ensure efficient OTA software update delivery by dynamically managing bandwidth and intelligently directing network traffic, thereby improving reliability, reducing latency, and preventing congestion.

The ETSI GS MEC 015 Bandwidth Management API, as depicted in Figure 6, provides a RESTful interface that allows for dynamic bandwidth allocation and resource management. The API supports essential operations such as retrieving existing bandwidth allocations (GET), creating new allocation requests (POST), updating bandwidth configurations (PUT/PATCH), and removing outdated allocations (DELETE). By leveraging MEC, the system can allocate network resources close to end-users, optimizing the transmission of OTA updates and reducing the strain on centralized data centers. This dynamic bandwidth provisioning ensures that software updates are delivered smoothly and efficiently, preventing potential delays caused by competing network traffic.

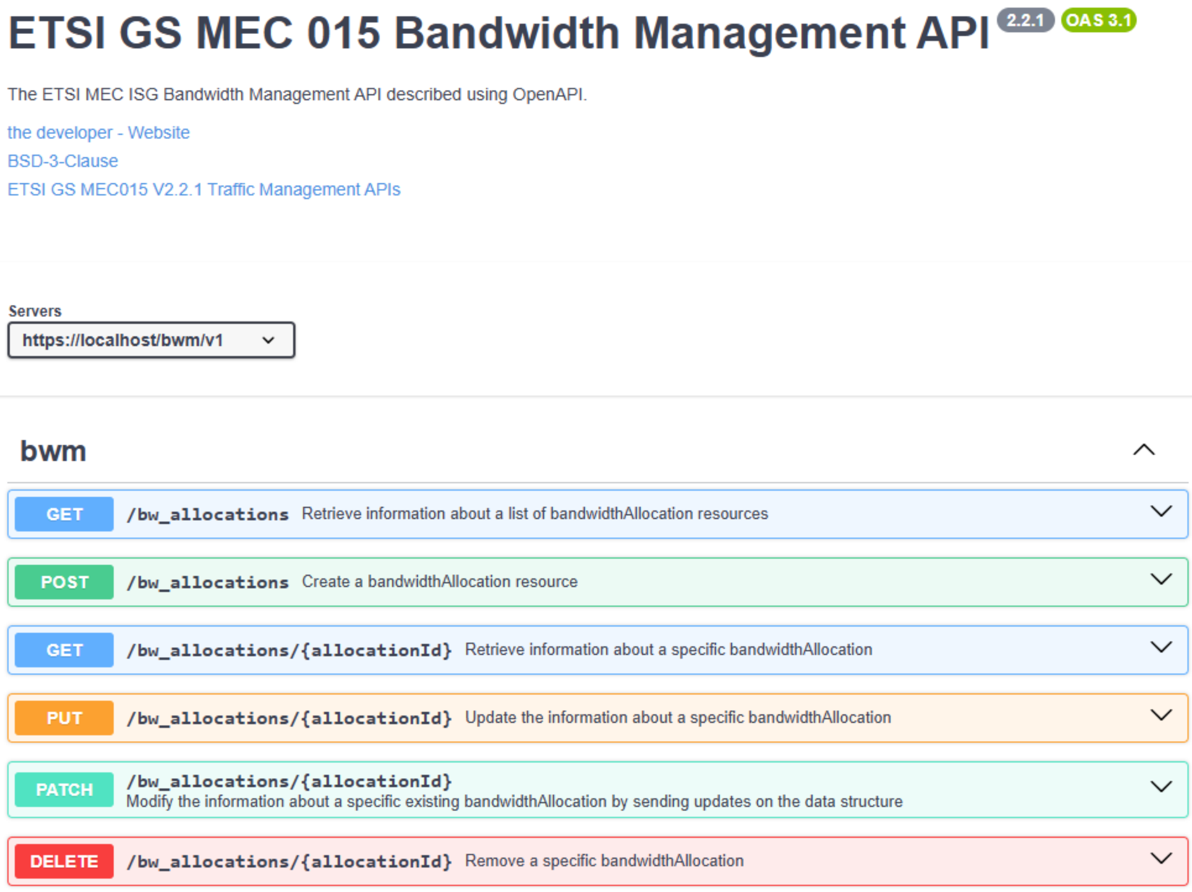

Figure 6 MEC 015 Bandwidth Management API

The ETSI TeraFlowSDN Controller, as shown in Figure 7, facilitates software-defined networking (SDN)-based traffic control and network orchestration. The interface allows users to select network topologies, upload JSON-based network configuration descriptors, and visualize managed network resources. By implementing SDN principles, TeraFlowSDN enables dynamic path selection, congestion avoidance, and real-time traffic engineering. This capability is particularly valuable for vehicular networks, where OTA updates must be transmitted over complex, heterogeneous infrastructures. The ability to reconfigure network paths dynamically ensures minimal disruptions and enhances the reliability of the update distribution process.

The preliminary findings from these two technologies indicate a strong potential for seamless network automation in vehicular ecosystems. The MEC Bandwidth Management API provides a scalable solution for prioritizing OTA updates while preventing excessive congestion. Simultaneously, TeraFlowSDN introduces intelligent traffic control mechanisms that dynamically optimize network routes based on real-time conditions. The combination of these technologies demonstrates the feasibility of a more adaptive, efficient, and secure approach to software updates in connected vehicles.

Overall, integrating MEC-based bandwidth management with SDN-powered network orchestration has the potential to significantly enhance the performance of vehicular networks. This approach ensures that software updates are delivered reliably while maintaining network stability and scalability. Future evaluations will focus on measuring key performance indicators (KPIs) such as end-to-end update latency, bandwidth allocation efficiency, and congestion avoidance rates to validate the effectiveness of this integrated solution.

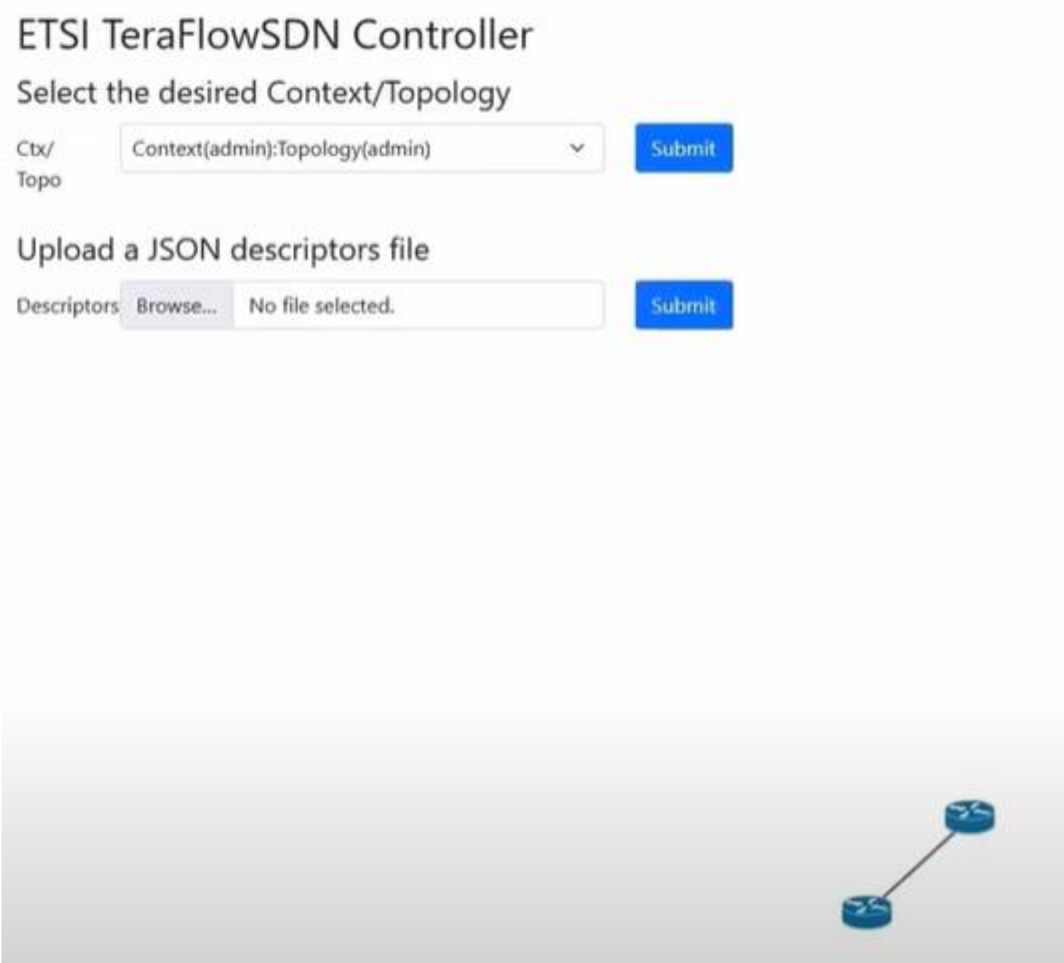

Figure 7 Integration with TeraFlowSDN

3.7 Final testing and validation

The integration of TeraFlow SDN with MEC and optical networking for vehicular communication and V2X updates demonstrates a highly scalable and efficient solution for managing OTA updates and data traffic in connected vehicle environments. The results showcase a seamless orchestration between bandwidth management APIs, software-defined networking, and real-time traffic optimization, ensuring low-latency, high-reliability, and dynamically adaptable network performance.

Figure 8 depicts an advanced vehicular network architecture within a TeraFlow SDN framework, where a connected vehicle initiates a network communication session routed through a series of networking switches. These switches are interfaced with optical transport nodes, depicted as green boxes, which manage light-wave-based data transmission for high-speed, low-latency connectivity. The optical backbone serves as the primary medium for transmitting vehicle-to-cloud data, ensuring robust, high-bandwidth connectivity. The data flow, shown as a red path, traces the vehicle's communication journey from the edge network to the cloud infrastructure, where processing, storage, and service provisioning occur. The cloud, depicted as a cluster of servers within a cloud icon, represents the final destination for OTA updates, security patches, and other vehicular applications that require remote access.

This architecture exemplifies a modern, software-defined approach to network management, enabling dynamic data path optimization, real-time reconfiguration, and intelligent routing based on network conditions. The solid lines in the figure illustrate physical wired connections, whereas dashed lines suggest wireless transmission or indirect virtualized pathways, contributing to flexible and adaptive network behavior.

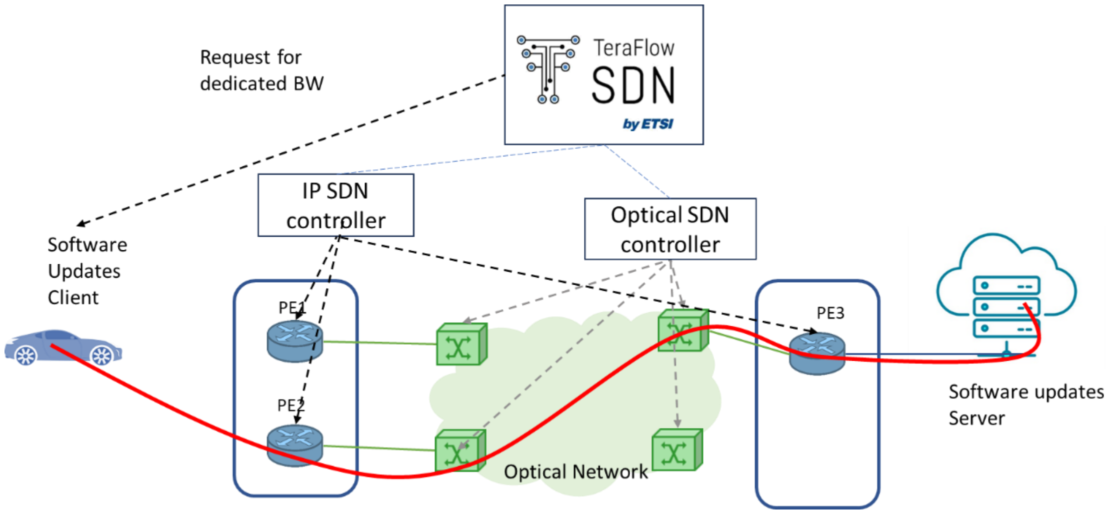

Figure 8 Proposed architecture for OTA vehicular software updates with efficient computation

Figure 9 and Figure 10 showcase the TeraFlow SDN Controller's device and service management capabilities. The device management interface (Figure 9) lists packet routers identified by UUIDs, showing their current operational status, driver configurations (OPENCONFIG), and connection endpoints. Each device is actively managed through configurable rules, ensuring that software-defined policies can dynamically adjust traffic based on demand.

Figure 10 highlights the service provisioning aspect, displaying an active network service that links two routers (CSG-1 and CSG-2) via the Layer 3 Network Model (L3NM). This confirms that TeraFlow SDN efficiently orchestrates service configurations, enabling scalable and policy-driven traffic management across the network. These findings indicate that TeraFlow SDN is fully capable of handling real-time networking requirements for V2X communications, efficiently routing traffic, and dynamically adjusting services to maintain optimal network performance.

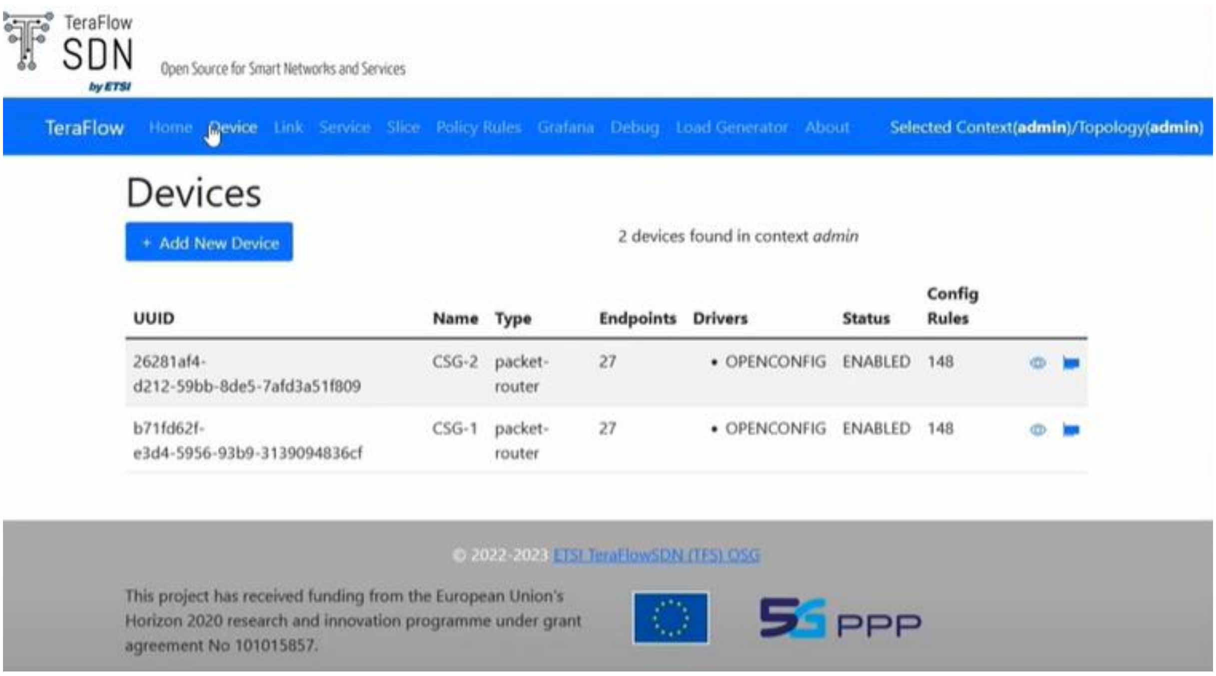

Figure 9 TeraFlowSDN managed devices

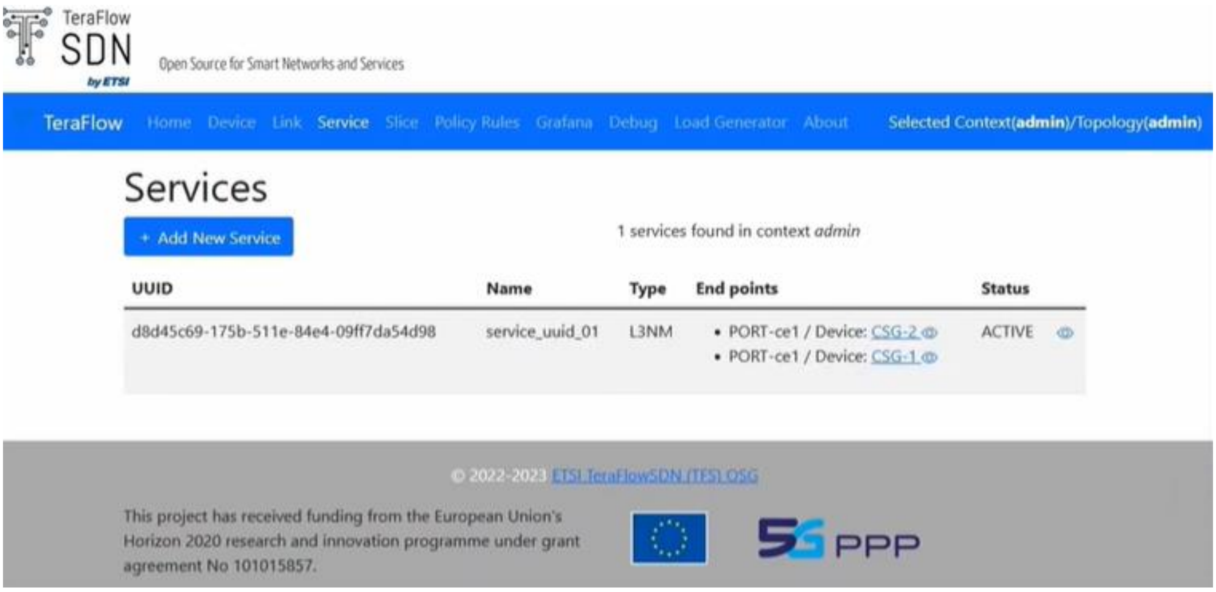

Figure 10 TeraFlowSDN Services

Figure 11 provides a packet capture (PCAP) analysis of a V2X update client interacting with TeraFlow SDN via the MEC Bandwidth Management API. The captured exchange illustrates HTTP-based communication, with a POST request from the V2X client to the /bw_allocations endpoint, initiating a bandwidth allocation request. The response (HTTP 200 OK) confirms successful bandwidth allocation, demonstrating that MEC and SDN integration can dynamically provision network resources based on demand.

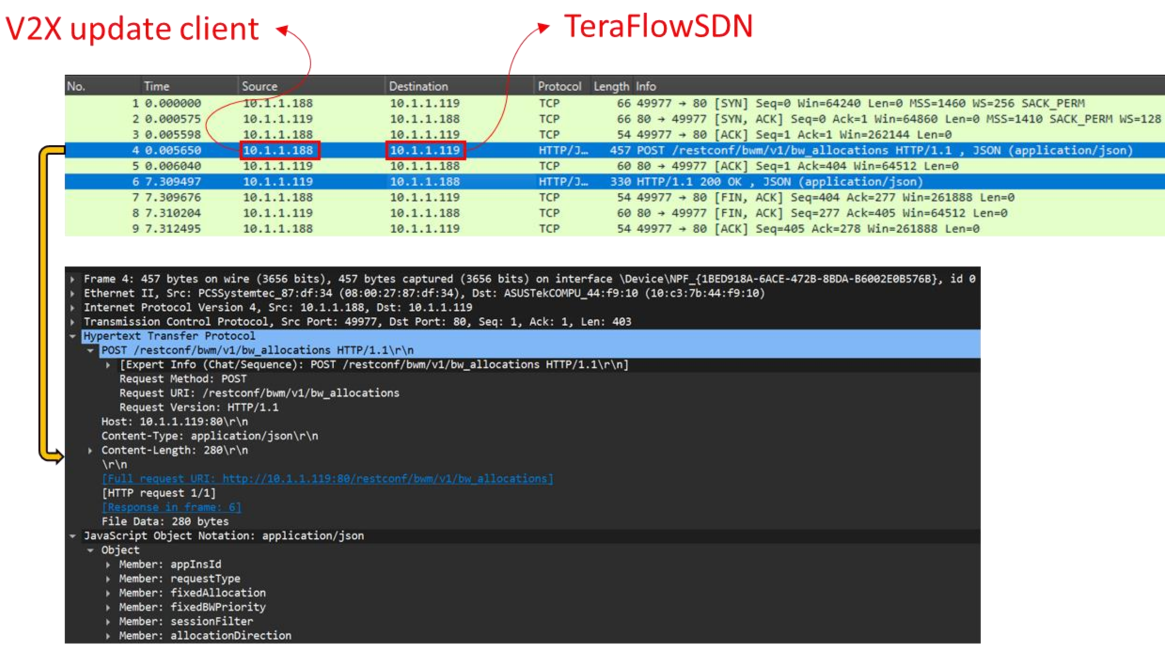

Figure 11 Wireshark capture of the integration

This interaction highlights several key aspects:

- Real-Time Bandwidth Allocation: The MEC API effectively provisions dedicated bandwidth for V2X updates, ensuring minimal transmission delays.
- Seamless API Integration: The standardized RESTful API allows applications to request and manage network resources dynamically, facilitating interoperability with third-party platforms.

- Traffic Engineering & Optimization: By coordinating with SDN-based traffic routing, MEC bandwidth allocation prevents network congestion, ensuring that critical software updates are delivered without performance degradation.

The integration of MEC, SDN, and Optical Networking in this experiment demonstrates several significant advantages for connected vehicular ecosystems:

- Ultra-Low Latency: The optical backbone and SDN-based traffic management significantly reduce end-to-end update latency, ensuring timely software updates for connected vehicles.
- Dynamic Bandwidth Allocation: The MEC API seamlessly provisions bandwidth on demand, preventing network congestion and ensuring high-priority services are delivered first.
- Scalability & Future-Proofing: The SDN-based service orchestration supports future network expansions, allowing more vehicles and traffic loads without significant architectural overhauls.
- Reliability & Resilience: The combination of MEC for edge computing, SDN for intelligent routing, and optical networks for high-capacity transport ensures fail-safe operations even under high-demand scenarios.

These results confirm that TeraFlow SDN, when integrated with MEC and optical networking, provides a scalable, secure, and highly efficient solution for managing V2X communication and OTA software updates. The dynamic interaction between bandwidth management APIs and SDN-based orchestration ensures optimal resource utilization, making it a compelling approach for next-generation connected vehicle infrastructures. Future work will focus on further performance benchmarking, security enhancements, and real-world deployment scenarios to validate its operational viability in production automotive networks.

4 Use case 2 proof-of-concept (PoC)

For the proof-of-concept of OTA vehicular software updates at the Circuit Parcmotor Castellolí, all software, both server-side and on-board unit (OBU) software, has been migrated to their final locations: the Castellolí server and the IDNEO-developed OBU installed in the test vehicle (Figure 12).

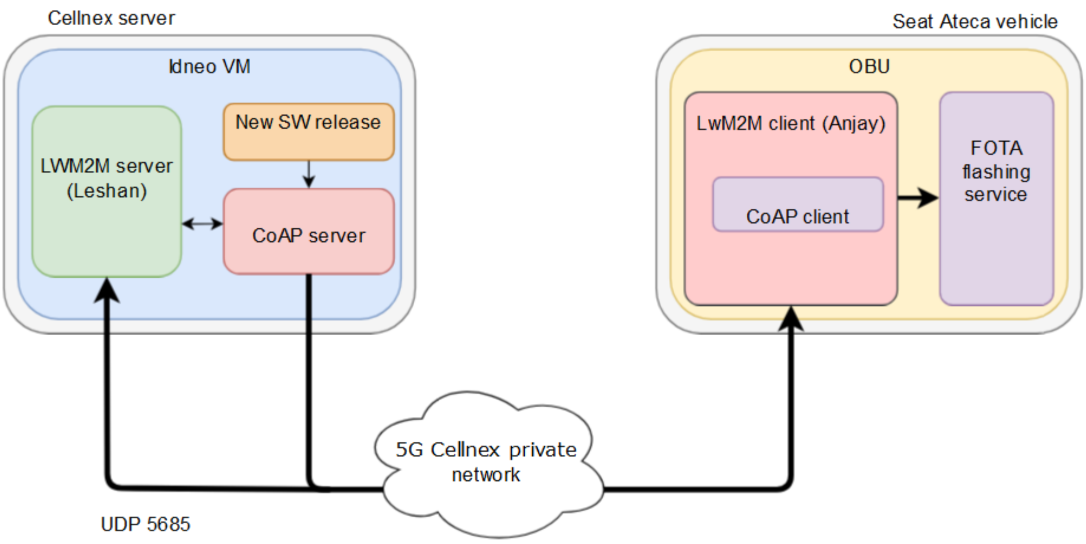

Figure 12 PoC architecture for OTA vehicular software updates at Castelloli

During the third round of tests, the focus was on evaluating network connectivity and performance in communications between the OBU and the Firmware Over-the-Air (FOTA) server. The assessment included analyzing persistent connections, reconnection processes, failure scenarios, download times, and network events that could potentially impact communication between the OBU application and the FOTA server. Additionally, the use case involving software updates from the management application was tested to ensure its functionality and reliability.

Below is the procedure followed to evaluate the use case. Figure 13 shows the dashboard used to manage the devices. There were no devices registered on the dashboard initially. This dashboard is generated by an application that, in addition to having a communications port, also has a web management port. This application was containerized and deployed at the edge of the network.

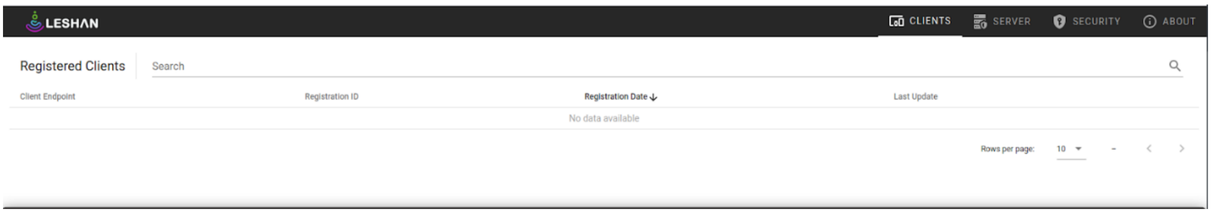

Figure 13 LWM2M server deployed on the virtual machine waiting for connections

Once the OBU was powered on, the device registration process with the server was observed, displaying the assigned endpoint name on the dashboard. Additionally, real-time logs from the client application running on the device were monitored during the test. Simultaneously, network communication packets were captured to facilitate further analysis of system performance and connectivity.

Figure 14 Top: Client connected to Server, Bottom: TCU console showing server logs

Within the management platform, relevant data can be obtained from the OBU. This information is communicated between the OBU and the server via COAP communication encrypted with DTLS. A relevant piece of information, which we can see in Figure 15, is the version of the release deployed on the OBU.

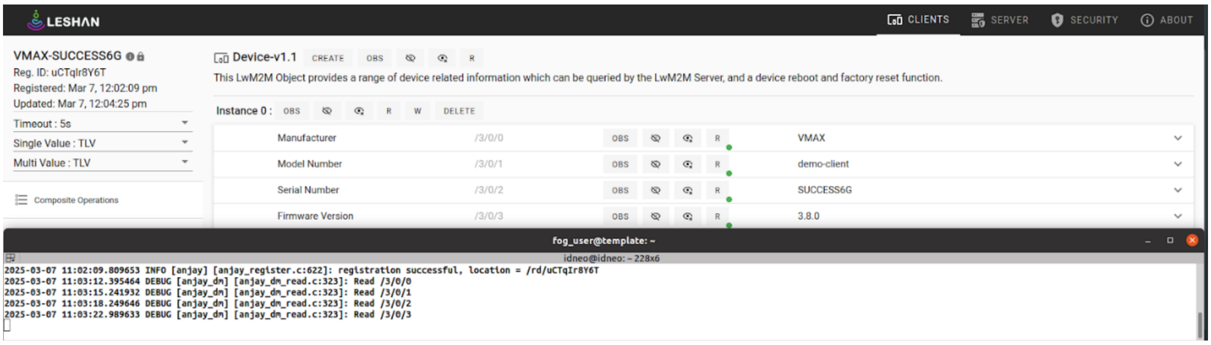

Figure 15 TCU and Server interaction, device data observation

Next, the server is instructed on the location of the file containing the new software version to be deployed. For the server, this file is considered a resource that must be transmitted as a configurable item. Prior to this step, the file was uploaded to a repository accessible via HTTPS, ensuring secure and efficient retrieval during the update process.

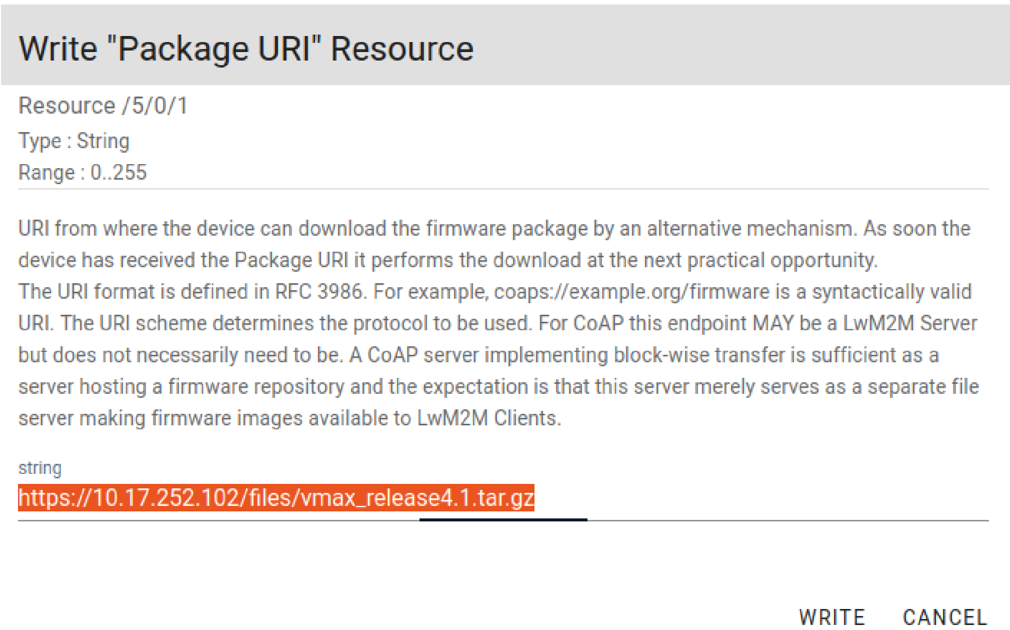

Figure 16 Top: Software Release configuration for update, Bottom: Software Releases in HTTPS repository

The file is deployed to the OBU, which downloads it once it has been instructed where to retrieve it. Figure 17 shows the *"firmware.fota"* file, which contains the updated software and is stored on the OBU's internal SD card. At this point, the Update button will be used to begin deploying the new release, since in a real-world environment, this new release will be deployed when the system is rebooted during an engine shutdown.

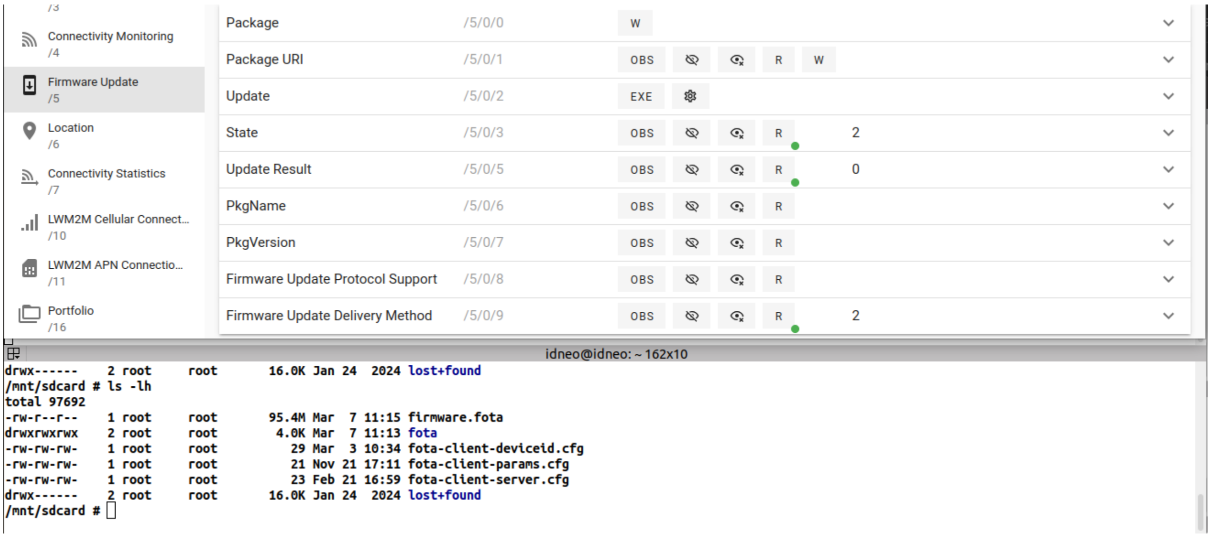

Figure 17 Top: Software Update User Interface, Bottom: Release Download to the TCU File System

Upon system reboot, the OBU executes a startup script that verifies whether a firmware update is required. If an update is necessary, the script initiates the update process and ensures its completion. Once the update is successfully applied, the OBU is expected to automatically reconnect to the edge server, as illustrated in Figure 18.

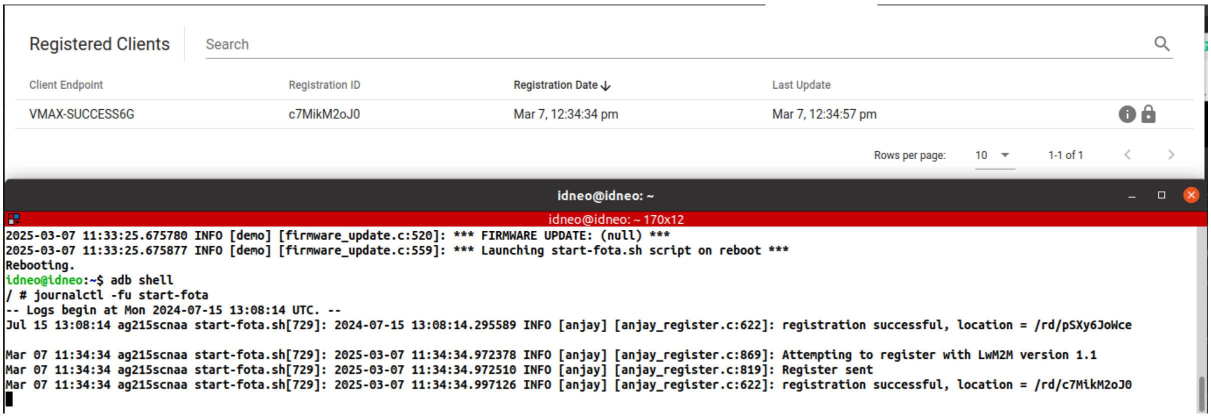

Figure 18 Reboot and automatic reconnection to the LWM2M server

A further analysis is performed based on the captured network traffic. As illustrated in Figure 19, the OBU, assigned the IP address 10.17.201.240, receives a reset command from the server at IP address 10.17.252.102. Following this, the OBU initiates a DTLS handshake to re-register with the network. This event marks the completion of the update process.

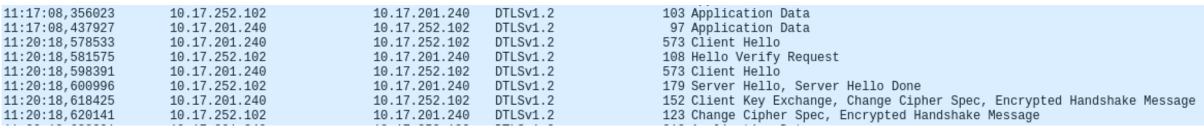

Figure 19 TCU offline while performing reboot and update = 3 min 10 s

5 Conclusions

Efficient computation management for OTA vehicular software updates is crucial in reducing latency, optimizing network resources, and ensuring seamless service delivery. The SUCCESS-6G-VERIFY project demonstrates how MEC-based workload distribution and AI-driven predictive resource allocation can improve the efficiency of update deployment while minimizing network congestion. The experimental results validate the advantages of intelligent caching and traffic-aware optimization in enhancing computational performance. Future research should focus on refining AI-driven orchestration techniques and expanding edge computing capabilities to support the growing complexity of vehicular software updates in increasingly connected and autonomous transportation systems.

A significant challenge in computation-efficient OTA updates is balancing the workload distribution between cloud and edge computing resources. The use of federated learning models and distributed processing architectures ensures that updates are processed closer to vehicles, minimizing reliance on centralized servers and reducing latency. This approach not only accelerates update dissemination but also enhances overall system resilience.

Furthermore, the integration of AI-enhanced resource scheduling optimizes bandwidth utilization and computational efficiency. By dynamically prioritizing critical updates and allocating processing resources based on real-time demand, the system prevents network congestion and improves update success rates. Continued advancements in edge AI and MEC infrastructure will be instrumental in achieving even greater levels of efficiency and reliability in vehicular software update ecosystems.

6 References

Abishek A, Vilalta R, Gifre L, Alemany P, Manso C, Casellas R, Martínez R, Muñoz R. Network Extensions to Support Robust Secured and Efficient Connectivity Services for V2X Scenario. In2024 24th International Conference on Transparent Optical Networks (ICTON) 2024 Jul 14 (pp. 1-4). IEEE.

Vilalta R, Vía S, Mira F, Casellas R, Muñoz R, Alonso-Zarate J, Kousaridas A, Dillinger M. Control and management of a connected car using sdn/nfv, fog computing and yang data models. In2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft) 2018 Jun 25 (pp. 378-383). IEEE.

Muñoz R, Vilalta R, Yoshikane N, Casellas R, Martínez R, Tsuritani T, Morita I. Integration of IoT, transport SDN, and edge/cloud computing for dynamic distribution of IoT analytics and efficient use of network resources. Journal of Lightwave Technology. 2018 Apr 1;36(7):1420-8.

Fallgren M, Dillinger M, Alonso-Zarate J, Boban M, Abbas T, Manolakis K, Mahmoodi T, Svensson T, Laya A, Vilalta R. Fifth-generation technologies for the connected car: Capable systems for vehicle-to-anything communications. IEEE vehicular technology magazine. 2018 Jul 17;13(3):28-38.

Garg S, Kaur K, Kaddoum G, Ahmed SH, Jayakody DN. SDN-based secure and privacy-preserving scheme for vehicular networks: A 5G perspective. IEEE Transactions on Vehicular Technology. 2019 May 20;68(9):8421-34.

Varma IM, Kumar N. A comprehensive survey on SDN and blockchain-based secure vehicular networks. Vehicular Communications. 2023 Aug 22:100663.

Meyer P, Hackel T, Langer F, Stahlbock L, Decker J, Eckhardt SA, Korf F, Schmidt TC, Schüppel F. A security infrastructure for vehicular information using sdn, intrusion detection, and a defense center in the cloud. In2020 IEEE Vehicular Networking Conference (VNC) 2020 Dec 16 (pp. 1-2). IEEE.