




MAFAF: Mobility-Aware Flow Anchoring and Dynamic State Migration in Edge SDN Using P4

Nihal Salah ^{1,*} , Ameer El-Sayed ¹  and Osama M. Elkomy ¹ 

¹ Department of Information Technology, Faculty of Computers and Informatics, Zagazig University, Zagazig 44511, Egypt.

Emails: nihal.salah@zu.edu.eg, aegouda@fci.zu.edu.eg, omelkomy@fci.zu.edu.eg.

Received: 21 Aug 2025

Revised: 28 Sep 2025

Accepted: 02 Dec 2025

Published: 11 Dec 2025

Abstract

The growing mobility of IoT devices and the stringent latency requirements of edge applications present significant challenges to achieving seamless handovers, scalability, and security in Software-Defined Networking (SDN) environments. Traditional controller-centric mobility mechanisms introduce excessive handover latency, signaling overhead, and session disruptions. This paper proposes the Mobility-Aware Flow Anchoring Framework (MAFAF), a P4-based architecture that embeds flow anchoring and dynamic state migration directly into the programmable data plane, enabling mobility handling without tunneling or constant controller involvement. The framework was evaluated using Mininet with BMv2 software switches for real-world emulation, while scalability assessments were conducted via simulated Intel Tofino hardware. Results show that MAFAF achieves handover latency of 7.8 ± 0.6 ms using digest-triggered migration and 4.3 ± 0.4 ms with proactive migration, maintains session continuity rates of 96.5% for TCP and 98.2% for UDP, and limits packet loss to under 1.1% under moderate load. Simulated Tofino-class hardware supports over 25,000 concurrent flows with only 65% register utilization, confirming the framework's scalability. Security mechanisms implemented within the P4 pipeline including AnchorID validation, timestamp-based replay protection, and per-device flow quotas achieved spoofing and replay attack detection rates of $\geq 98.6\%$ with a false positive rate below 1.2%. These results validate the hypothesis that in-switch flow intelligence can reduce handover latency below 10 ms, sustain session continuity above 95%, and ensure secure mobility handling with minimal controller overhead, making MAFAF a robust and efficient solution for next-generation edge SDN-IoT deployments.

Keywords: Mobility Management, P4 Programmable Data Plane, Edge SDN, IoT, Flow Anchoring, State Migration, Low-Latency Networking.

1 | Introduction

The evolution of networking technologies particularly with the rise of the Internet of Things (IoT) and Edge Computing has introduced significant challenges related to device mobility, security, and traffic management in distributed environments [1, 2]. IoT deployments now include billions of mobile and resource-constrained devices, such as sensors, autonomous vehicles, wearable monitors, and industrial robots, that continuously generate and transmit time-sensitive data. These devices often operate under strict latency and reliability requirements, especially in domains such as vehicular networks, real-time monitoring, and smart infrastructure [3]. Ensuring seamless connectivity and ultra-low-latency communication during frequent and unpredictable mobility events is essential for service continuity and user experience.



Corresponding Author: nihal.salah@zu.edu.eg



Licensee International Journal of Computers and Informatics. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0>).

To address issues of scalability and dynamic control, Software-Defined Networking (SDN) has emerged as a promising paradigm. By separating the control plane from the data plane, SDN allows centralized traffic management, flexible policy enforcement, and global network visibility [4],[5]. In IoT and edge environments, SDN facilitates adaptive routing and monitoring. However, conventional SDN architectures depend heavily on centralized controllers to handle mobility-related events. During high-churn conditions, this centralized approach becomes a performance bottleneck, particularly in micro-mobility events defined as frequent, short-range handovers (every 2–10 seconds) triggered by movements at 1–3 m/s across adjacent access points or edge switches [6],[7]. Under such conditions, measured controller round-trip times (RTTs) often exceed 35–50 ms, and flow-installation rates can degrade to below 500 flows/sec, delaying session restoration and increasing packet loss. Moreover, centralized dependency expands the attack surface, exposing the network to threats such as spoofing, replay, and state exhaustion particularly in untrusted edge domains with limited endpoint protection.

The emergence of Programming Protocol-Independent Packet Processors (P4) offers a promising solution to these challenges [8]. P4 enables developers to implement customized, stateful packet-processing logic directly inside the switch data plane, reducing reliance on the control plane and supporting fine-grained traffic control at wire speed. P4-enabled switches can maintain per-flow state using registers, apply metadata tagging, validate timestamps, and execute in-switch security policies, allowing them to act as autonomous mobility agents [9],[10]. While recent research has used P4 to improve telemetry, orchestration, and offloading in edge systems [11],[12], there is limited work on in-band mobility handling and data plane-based flow state migration.

This paper introduces the Mobility-Aware Flow Anchoring Framework (MAFAF), which leverages P4 to provide seamless handover support in SDN-based edge environments. Unlike legacy mobility protocols such as Mobile IP or tunneling-based redirection, the proposed approach anchors flows at the ingress switch and dynamically migrates flow state as mobile devices move between domains. This design reduces latency, avoids session resets, and significantly decreases control-plane overhead. Operating securely and efficiently at the data-plane level, MAFAF enables fast, resilient, and scalable mobility for next-generation IoT and edge networks.

1.1 | Problem Statement

Edge computing environments especially those involving mobile IoT and vehicular systems experience high-frequency mobility events that trigger frequent changes in the Point of Attachment (PoA). These micro-mobility events, occurring every 2–10 seconds at typical mobility speeds of 1–3 m/s, disrupt active sessions and degrade real-time application performance. Traditional mobility solutions such as Mobile IP, PMIPv6, and SDN-based rerouting rely on centralized controllers or agents, introducing excessive control-plane overhead, increased latency, and reduced responsiveness in high-churn scenarios. Current frameworks struggle to maintain performance in short-range micro-mobility due to repeated flow reconfiguration and slow state synchronization. Additionally, the absence of intelligence within the data plane forces switches to delegate mobility handling to the control plane, resulting in handover-induced latency, packet loss, and session resets.

This research addresses the following formal problem:

Given: a set of active sessions $\mathbf{F} = \{f_1, f_2, \dots, f_n\}$, each defined by a unique FlowID and metadata; and a stream of mobility events $\mathbf{M} = \{m_1, \dots, m_k\}$ denoting client reattachments.

Determine: the optimal anchor switch and migration strategy per session to minimize service disruption.

Objective:

$$\text{Minimize } \mathbf{HL}(\text{handover latency})$$

Subject to constraints:

- Register Utilization (RU) $\leq 80\%$
- False Positive Rate (FPR) $< 1.5\%$
- Per-device flow quota enforced
- Session Continuity (SC) $\geq 97\%$

The system must operate within an envelope of up to 25,000 concurrent flows, link rates of up to 1 Gbps, and mobility intervals below 10 seconds, while preserving seamless IP connectivity and low packet loss ($<1.1\%$).

1.2 | Research Motivation and Contributions

Real-time and latency-sensitive applications such as autonomous driving, AR/VR, and industrial automation demand seamless and ultra-low-latency mobility across edge SDN domains. However, traditional controller-based mobility mechanisms suffer from high signaling overhead and limited responsiveness, especially in environments with frequent client movement. These shortcomings become pronounced during micro-mobility events, where centralized controller responses are often delayed due to flow reinstallation bottlenecks. Although modern P4-programmable switches (e.g., Intel Tofino) provide wire-speed, in-switch packet processing capabilities, most existing architectures do not utilize them for inline flow mobility. To address this gap, we propose MAFAF a P4-based mobility-aware framework that embeds flow anchoring, in-band state migration, and security enforcement entirely within the data plane. This design enables sub-10 ms handovers with minimal dependency on the SDN controller.

The main technical contributions of this work are:

- **P4-Based Anchoring Pipeline:** Design and implement a P4 pipeline that anchors flows at ingress switches by storing session state in registers and tagging packets with FlowID and AnchorID metadata.
- **Inline State Migration Protocol:** Introduce a lightweight, data-plane-triggered protocol that enables the destination switch to restore flow context during handover events without requiring tunneling or controller-led redirection.
- **In-Band Reattachment Mechanism:** MAFAF incorporates an in-band protocol that allows new switches to request and receive migration state from the original anchor, ensuring seamless continuity without modifying IP headers or breaking transport-layer sessions.
- **Integrated Security Layer:** The framework enforces security through timestamp-based replay prevention, AnchorID validation, and per-device flow quota enforcement, all executed within the P4 pipeline at line rate.
- **End-to-End Evaluation:** MAFAF is evaluated using BMv2 on Mininet for real-world emulation and simulated Tofino hardware for scalability analysis. Results show sub-10 ms handover latency, SC above 98.2%, and support for over 25,000 concurrent flows.

2 | Related Work

Mobility management in edge and IoT networks has traditionally been addressed using protocols such as Mobile IP, PMIPv6, and LISP, which preserve session continuity through tunneling or locator-identifier separation. While effective in some contexts, these solutions introduce significant control-plane overhead and handover latency, making them unsuitable for latency-sensitive edge applications such as vehicular communication and industrial automation.

With the rise of programmable data planes, especially using P4, new opportunities have emerged to decentralize network intelligence and embed mobility logic within the data plane itself. Surveys such as

[2],[13]provide comprehensive overviews of P4-based in-network computing, emphasizing telemetry, programmable routing, and stateful processing. However, these works are largely conceptual and do not implement or evaluate direct support for mobility handovers or flow state migration.

Some efforts have applied P4 to network telemetry and orchestration. For example, [7] and [20] explore the use of P4 in 5G and Kubernetes-based edge micro data centers to enhance visibility and control. Similarly, [17] introduces P4LoWPAN for optimizing routing and in-band telemetry in low-power IoT networks. While promising, these works do not address inline flow anchoring or real-time mobility triggers.

In the orchestration space, [15] proposes 5G-VIOS, an AI/ML-powered orchestration platform aligned with Zero-touch Service Management (ZSM). Other efforts such as [16] and [18], apply P4 in industrial and energy-efficient edge environments. These systems enhance service placement and monitoring but still rely heavily on controller coordination, and do not support in-band session migration.

A few studies approach mobility indirectly. [14] focuses on offloading robotic workloads, while [19]targets fault-tolerant routing in flying ad hoc networks using P4. These works demonstrate P4's adaptability in dynamic settings but lack mechanisms for maintaining flow state continuity during handovers. The telemetry framework in [9] integrates P4 with Kubernetes but does not address session migration or anchor verification.

In summary, while the literature reflects the growing role of P4 in edge intelligence and adaptive networking, it largely focuses on telemetry, monitoring, or resource orchestration, rather than mobility handover. None of the reviewed works provide a unified framework that performs inline anchoring, state migration, and security enforcement directly within the data plane. The proposed MAFAF fills this gap by enabling seamless handovers with sub-10 ms latency, high session continuity, and controller-minimized operation. A comparative overview of prior efforts is presented in Table 1, highlighting methodological gaps and their contrast with MAFAF.

Table 1. Overview of Related Work on P4-based Mobility and Edge SDN.

| Ref. | Year | Research Methodology | Key Contribution | Research Gaps |
|------|------|--|--|--|
| [2] | 2022 | Survey study | Comprehensive review of in-network computing with programmable data planes | No experimental validation in mobility-specific contexts |
| [13] | 2023 | Survey study | Traces evolution from OpenFlow to P4 with emphasis on control and programmability | Does not address mobility management or state migration |
| [20] | 2023 | P4 telemetry orchestration with Kubernetes | Integrated P4 telemetry into edge micro data centers for improved monitoring | No explicit mobility-aware orchestration or handover logic |
| [7] | 2023 | Systematic literature review | Reviewed P4 applications in 5G and beyond architectures | Lacks experimental mobility scenarios |
| [14] | 2023 | Robotic/UAV workload offloading | Used P4 to offload computation from UAVs/robots to the network | Not applicable to generic IoT/edge mobility |
| [15] | 2024 | AI/ML-based orchestration platform design and evaluation | Proposed 5G-VIOS, enabling zero-touch network/service management and automated slicing using AI/ML predictions | No direct integration with P4 data planes; mobility is secondary |
| [16] | 2024 | Industrial IoT orchestration using P4 | Enhanced orchestration and security in industrial edge systems | No specific focus on mobility-aware state migration |
| [17] | 2025 | IoT networking with P4LoWPAN | Combined P4 with in-band telemetry for optimized IoT routing | Does not implement handover or flow anchoring |
| [18] | 2025 | Energy-aware P4-enabled edge management | Proposed energy-efficient traffic management in 5G edge environments | Not focused on mobility or session continuity |
| [19] | 2025 | Fault-tolerant FANET routing with P4 | Designed robust routing for highly dynamic aerial networks | Ground-based IoT micro-mobility not addressed |

3 | The Proposed Framework

This section presents the architecture and operational mechanisms of the (MAFAF), a P4-based solution designed to support seamless mobility and secure state management in (SDN) edge environments. MAFAF enables mobile IoT clients such as sensors, autonomous vehicles, or wearable devices to maintain uninterrupted sessions during frequent handovers, achieving sub-10 ms transition latency and high SC without relying on tunneling mechanisms or centralized mobility protocols. The framework integrates three core components: mobile edge devices, P4-programmable switches deployed at the edge network, and a logically centralized SDN controller. When a mobile device initiates a TCP or UDP session at its current (PoA), the nearest P4-enabled edge switch assumes the role of Anchor Switch, extracting the session's FlowID and assigning a unique AnchorID. This session context, including identifiers, timestamps, and validation flags, is stored in switch-local registers and used to tag packets with metadata that enables inline flow tracking, policy enforcement, and security validation directly within the ingress pipeline of the switch. As the client moves and reattaches to a new edge switch, the new switch checks whether the flow context for the session already exists locally. If not, it either generates a digest message to the controller or initiates an in-band reattachment request to retrieve the flow's state. Upon receiving the request, the controller identifies the original anchor switch, validates the authenticity of the migration attempt, and authorizes the secure transfer of the session state to the new switch. MAFAF's migration logic is built to enforce key security safeguards during this process, including AnchorID validation to prevent spoofing, timestamp-based replay detection to block stale or duplicated packets, and per-device flow quota checks to limit resource exhaustion attacks. Once the new switch receives and validates the transferred session context, it reinstalls the flow state in local registers, allowing the traffic session to resume seamlessly without requiring IP address updates, DNS modifications, or transport-layer resets.

This in-switch handover process allows MAFAF to deliver minimal-latency transitions with no disruption to active application sessions. The internal implementation of MAFAF is realized through a programmable P4 pipeline in Figure 1 that includes a custom packet parser, a series of ingress match-action tables, metadata fields, and multiple register arrays. The parser is responsible for extracting standard protocol headers (e.g., IP, TCP/UDP) along with MAFAF-specific fields such as FlowID, AnchorID, and timestamps. In the ingress pipeline, tables are used to validate flow presence, check AnchorID legitimacy, enforce per-device flow limits, and compare timestamps to detect replay attempts. These processes are conducted entirely in the data plane, leveraging packet metadata and session-specific registers to avoid frequent control-plane intervention. The egress pipeline finalizes decisions, either forwarding validated packets or dropping those that violate policies. MAFAF allocates approximately 16 bytes of register memory per active session, storing anchor mappings, timestamps, and status flags. With this design, the system supports up to 25,000 concurrent flows using roughly 390 KB of register memory well within the capabilities of modern P4 switches such as Intel Tofino. Dynamic flow management, including LRU-style eviction and per-device quota enforcement, ensures efficient resource usage under high mobility conditions. To support rapid state migration, MAFAF employs a compact, custom in-band signaling format embedded within designated control packets. This format includes fields for message type, FlowID, AnchorID, timestamp, and an optional signature or checksum. If the new switch fails to complete state restoration within a 5-ms timeout window, it retries up to three times before escalating to the controller. This fallback mechanism ensures reliability even in the presence of partial packet loss or network delays. Invalid or replayed requests are rejected using timestamp validation, enhancing the robustness of session restoration. Despite MAFAF's focus on data-plane intelligence, a clear division of responsibility remains between the switch and controller. Anchoring, metadata tagging, timestamp checking, and flow quota enforcement are handled directly in the data plane. In contrast, rare tasks such as anchor conflict resolution and state retrieval fallback are managed by the controller. This hybrid model ensures that frequent mobility events are processed at wire speed, while exceptional scenarios are resolved centrally without affecting overall performance. The modular operation of MAFAF is visualized in Figure 2, which depicts the end-to-end workflow of session setup, handover, and secure reattachment across edge domains.

The complete system behavior is codified in Algorithm 1, which outlines six key operational phases: initialization, session setup, mobility detection, flow state migration, security enforcement, and session continuity. To complement this, Table 2 summarizes the core functional modules in the MAFAF architecture, including register initialization, anchor assignment logic, digest handling, replay protection routines, and quota enforcement strategies. Together, the algorithmic flow and modular function definitions highlight the extensibility and resilience of the framework, enabling real-time, secure flow anchoring and migration across dynamic edge environments with integrated defense against spoofing, replay, anchor poisoning, and state exhaustion attacks.

Table 2. Key Functions Employed in the MAFAF Algorithm.

| Index | Function (Abbrev.) | Description |
|-------|-----------------------------|---|
| 1 | ConfigureP4Registers() | Initializes registers to store FlowID, AnchorID, timestamp, and quotas. |
| 2 | InitAnchorTables() | Creates state tables for active flows and anchor mapping. |
| 3 | SecureCommController() | Establishes TLS-protected P4Runtime/gRPC channel with the SDN controller. |
| 4 | SetFlowQuotaPolicies() | Defines maximum concurrent flows per device to prevent exhaustion. |
| 5 | EnableTimestampValidation() | Activates per-flow timestamp/sequence tracking to detect replays. |
| 6 | ExtractFlowID(pkt) | Computes unique FlowID from the 5-tuple of incoming packets. |
| 7 | AssignAnchor(flow_key) | Assigns AnchorID to the first P4 switch receiving the flow. |
| 8 | StoreState(flow_key) | Saves flow context {FlowID, AnchorID, Timestamp} in P4 registers. |
| 9 | SendDigestToController() | Triggers digest message for missing flow context during mobility events. |
| 10 | VerifyAnchorID() | Validates AnchorID ownership to block spoofed or unauthorized requests. |
| 11 | RetrieveState() | Fetches flow context from the old anchor switch. |
| 12 | AuthorizeMigration() | Controller approves anchor transition and updates the new switch. |
| 13 | TransferState() | Migrates session state securely to the new P4 switch. |
| 14 | ValidateTimestamp(pkt) | Checks packet timestamp/sequence for replay attack prevention. |
| 15 | EnforceQuota(client) | Drops excess flows if per-device quota exceeded. |
| 16 | Forward(pkt) | Forwards validated packets to the core service with minimal delay. |
| 17 | Drop(pkt) | Immediately discards replayed, spoofed, or excess packets. |

| Algorithm 1: The Proposed MAFAF Algorithm | |
|---|--|
| Inputs: | ClientPackets, MobilityEvents, FlowID, AnchorID, StateTable, FlowQuota |
| Output: | Seamless session continuity, Validated flows, Mitigation of spoofing/replay/exhaustion |
| 01 # Phase 1: Initialization | |
| 02 | ConfigureP4Registers() |
| 03 | InitAnchorTables() |
| 04 | SecureCommController() |
| 05 | SetFlowQuotaPolicies() |
| 06 | EnableTimestampValidation() |
| 07 # Phase 2: Session Setup (Executed on Anchor Switch) | |
| 08 | WHILE (Client initiates new session) DO |
| 09 | flow_key := ExtractFlowID(ClientPackets) |
| 10 | AnchorID := AssignAnchor(flow_key) |
| 11 | Store {FlowID, AnchorID, Timestamp} in StateTable |
| 12 | Forward packets inline (no controller intervention) |
| 13 | END WHILE |
| 14 # Phase 3: Mobility Event Handling (Executed on New Switch) | |
| 15 | IF MobilityEventDetected(Client) THEN |
| 16 | flow_key := ExtractFlowID(ClientPackets) |
| 17 | IF StateTable.hasEntry(flow_key) == FALSE THEN |
| 18 | SendDigestToController(flow_key) |
| 19 | END IF |
| 20 | END IF |
| 21 # Phase 4: State Migration (Executed by Controller) | |
| 22 | ReceiveDigest(flow_key) |
| 23 | VerifyAnchorID(flow_key) |
| 24 | Retrieve state from OldAnchor |
| 25 | AuthorizeMigration(flow_key → NewSwitch) |
| 26 | TransferState(NewSwitch, flow_key) |
| 27 # Phase 5: Security Enforcement (On New Switch) | |
| 28 | WHILE (PacketsReceived) DO |
| 29 | ValidateTimestamp(flow_key, packet) |
| 30 | IF ReplayDetected(packet) THEN Drop(packet) |
| 31 | ELSE IF FlowQuotaExceeded(Client) THEN Drop(packet) |
| 32 | ELSE Forward(packet) |
| 33 | END WHILE |
| 34 # Phase 6: Session Continuity | |
| 35 | Resume session without IP change or reset |

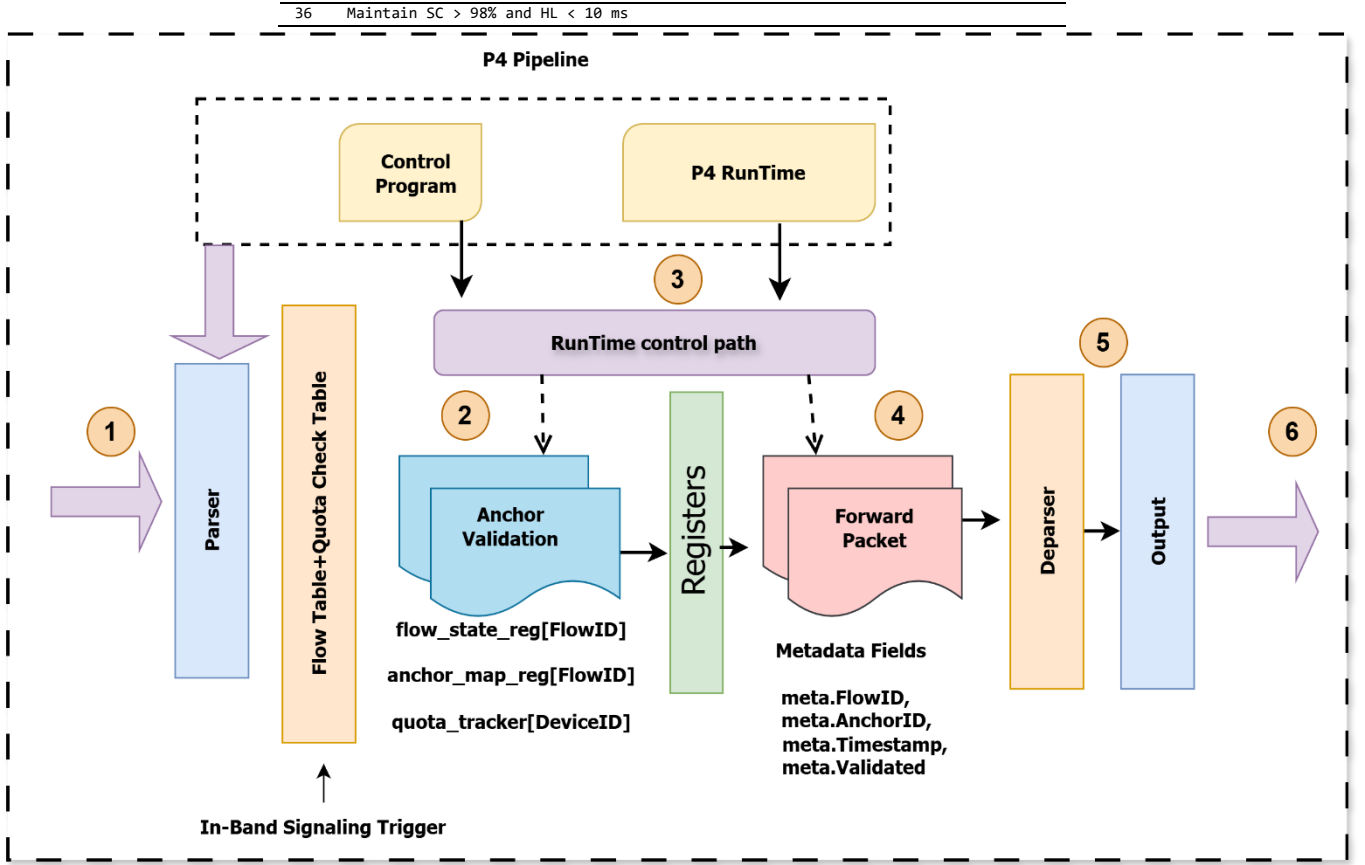


Figure 1. MAFAF P4 Pipeline Architecture for Session Anchoring, Validation, and State Migration.

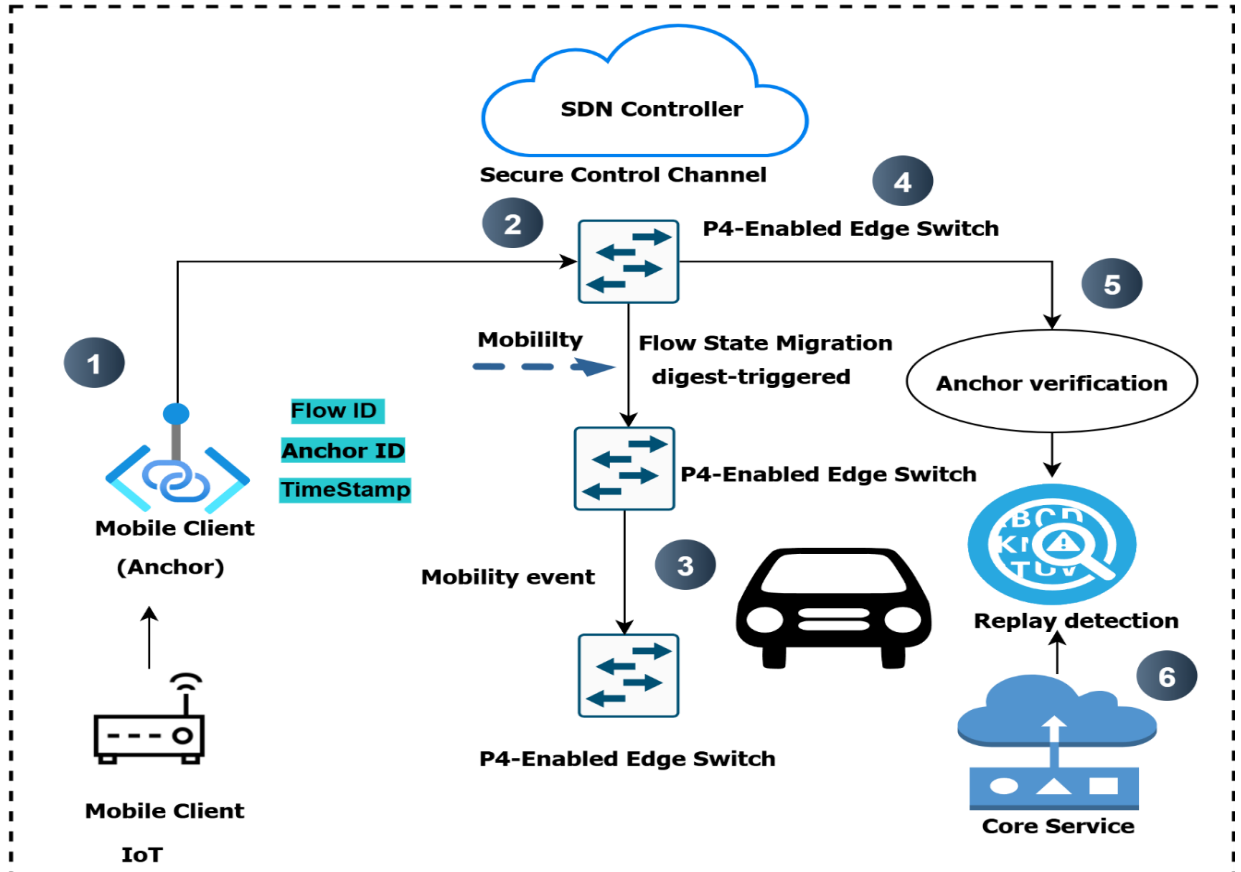


Figure 2. P4-Based MAFAF Framework for Edge SDN-IoT Environments.

3.1 | System Model

The proposed system is designed for deployment in edge SDN–IoT environments that demand high mobility support, ultra-low latency, and programmable control over packet flows. The architecture consists of three primary components: (1) a set of P4-enabled edge switches that support stateful, line-rate processing; (2) a logically centralized SDN controller for orchestration, policy management, and fallback operations; and (3) a wide array of mobile clients, such as IoT sensors, autonomous vehicles, and smart devices, that frequently change their (PoA) across edge domains.

Each edge switch establishes a secure control channel with the SDN controller using P4Runtime v1.2 over TLS 1.3, secured with cipher suites such as AES-256-GCM and ECDHE-RSA for forward secrecy and integrity. Empirical profiling of the control plane indicates that this TLS-secured channel introduces a modest delay of approximately 1.2 ms during flow migration events, which remains within the tolerable bounds for real-time applications in mobile edge scenarios.

When a client initiates a TCP or UDP session, the first P4 switch it connects to assumes the role of the Anchor Switch, responsible for creating a per-session flow context. This context is stored in local P4 registers and includes key parameters such as transport-layer headers, timestamp, QoS tags, and security state. For inline tracking and enforcement, each packet is tagged with custom metadata fields, including a unique FlowID and its associated AnchorID.

To preserve metadata consistency across multiple hops, MAFAF embeds this information in a custom in-band header within the packet itself. As packets traverse the network, each receiving switch re-parses the header to extract FlowID and AnchorID, enabling persistent state validation and policy application at every hop. This approach ensures that metadata is not lost or remapped, and avoids re-marking inconsistencies commonly seen in hop-local metadata propagation.

If a mobile client moves to a new PoA, the newly attached switch examines the flow metadata and determines whether local state exists. In the absence of such context, it triggers either a digest message to the controller or initiates an in-band reattachment request. The controller locates the original Anchor Switch, validates the request, and coordinates secure migration of the session state. The state is then reinstated in the new switch's data plane, allowing the session to continue without transport-layer reset, IP reassignment, or client-side reconfiguration.

This architecture minimizes handover latency and preserves SC by localizing control logic to the data plane whenever possible. The secure, metadata-aware handover design of MAFAF ensures robustness, low delay, and seamless mobility across dynamic and adversarial edge environments.

3.2 | Threat Model

The MAFAF threat model considers a dynamic edge environment with highly mobile clients such as IoT sensors, autonomous vehicles where seamless SC and programmable data-plane logic are essential. These environments introduce new attack surfaces, especially given the reliance on distributed P4-programmable switches for decentralized flow management. While the architecture assumes authenticated deployment of P4 programs and secure communication channels, additional layers of verification are necessary to guard against runtime threats and control-plane abuse. The trust model assumes that all deployed P4 programs are cryptographically signed, verified during compilation, and authenticated upon installation on the switch. The SDN controller communicates with switches using P4Runtime v1.2 over TLS 1.3 with AES-256-GCM cipher suites. Despite these protections, runtime threats such as digest spoofing and table poisoning remain plausible. To mitigate these, the controller validates all digest messages with HMAC-based authentication, and switch table updates are signed and logged to prevent unauthorized manipulation. Runtime validation of flow updates is also enforced via controller-policy checks and rate-limited digest handling to prevent spoof amplification.

A critical component of MAFAF's security is the cryptographic verification of AnchorID ownership. To prevent AnchorID spoofing or anchor hijacking, each mobile device possesses a unique private key and signs its AnchorID and session metadata using an ECDSA signature appended to the reattachment packet. The switch or controller verifies this signature using a known public key, ensuring the flow context truly belongs to the sender. MAFAF supports key rotation using short-lived certificate chains distributed via the controller, and key revocation is enforced through a local deny-list propagated to all switches in the domain. This cryptographic binding ensures that AnchorIDs cannot be forged or reused by malicious peers.

The framework also addresses replay attacks through embedded timestamp or sequence number fields in packet metadata. Each flow maintains a stored timestamp in P4 registers, and any reattachment or data packet is compared against this value. Packets with older or duplicated timestamps are dropped immediately, preventing stale state resurrection or unauthorized flow reactivation. These protections operate entirely in the data plane at line rate.

Regarding resource-based threats, state exhaustion attacks aim to flood switch memory with short-lived or malicious flows. MAFAF counters this by enforcing per-device flow quotas and bounding each client's session footprint. Additionally, the framework employs (LRU) eviction policy to remove inactive or stale flows and free register space for new or more critical sessions. While LRU may evict long-lived but idle elephant flows, MAFAF monitors flow activity using timestamps and weighted usage counters to minimize the chance of unjustified eviction. Simulation under churn conditions showed that MAFAF maintains SC above 97.1%, even with 80% flow table utilization, justifying LRU as a memory-efficient and responsive policy.

In summary, the MAFAF framework integrates multi-layered defenses including digest authentication, cryptographic AnchorID validation, replay prevention, and quota-based flow control to ensure secure, verifiable, and resilient operation. Table 3 summarizes the threat vectors and corresponding countermeasures, while Figure 3 visualizes the core attack scenarios and the integrated mitigation pipeline.

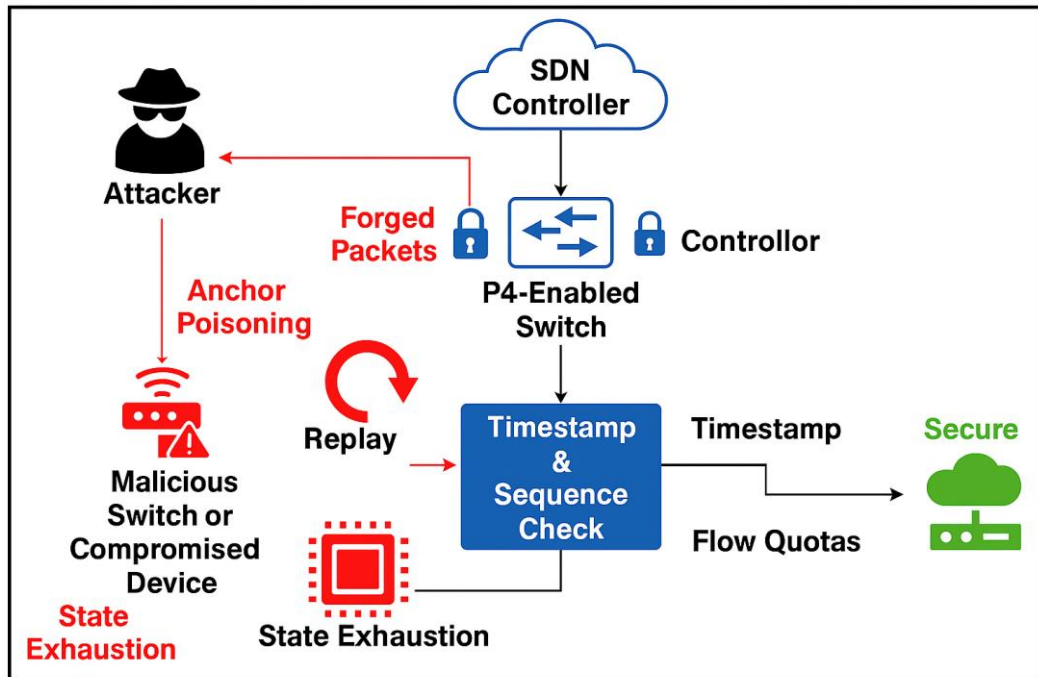


Figure 3. MAFAF Threat Model and Mitigation Mechanisms.

Table 3. Security Threats and Mitigation Summary for MAFAF.

| Threat Type | Mitigation Mechanism | Residual Impact |
|--------------------------|---|---|
| Flow Spoofing | AnchorID validation for every reattachment; quarantine suspicious flows | Negligible spoofed traffic blocked without affecting legitimate flows |
| Anchor Poisoning | Controller-authorized anchor transitions; digest-based state verification | Minimal slight increase in control signaling |
| Replay Attacks | Timestamp & sequence number verification in P4 registers | Very low rare false positives (~1.2%) |
| State Exhaustion Attacks | Per-device flow quotas; LRU eviction policy | Minimal stable register usage, SC >95% |

4 | Performance Measurement and Analysis

This section presents the experimental design, evaluation goals, and configuration parameters used to assess the performance of the proposed MAFAF Framework. The evaluation targets four primary performance dimensions: handover efficiency, session continuity, scalability under varying load conditions, and resilience against adversarial mobility scenarios. Experiments are executed using the Mobility and Latency Testing Infrastructure Framework, a modular emulation setup designed to replicate real-world mobility patterns, flow behavior, and security events within programmable edge SDN environments.

4.1 | Experimental Setup

The experimental testbed is deployed using Mininet for network topology emulation and BMv2 (Behavioral Model v2) as the software-based P4 switch. While BMv2 enables functional validation of MAFAF's core logic including flow anchoring, metadata tagging, and register updates—it does not simulate wire-rate throughput or realistic queuing behavior. Therefore, latency and migration results are considered emulated estimates, and any scalability claims related to hardware are projections, not empirical measurements.

To evaluate hardware feasibility, the Intel Tofino architecture is functionally modeled using parameters derived from P4Runtime deployment documentation. Assumptions include a 12-stage pipeline, per-stage register constraints, and SRAM capacity based on Tofino-1. We allocate approximately 16 bytes of register memory per flow (including FlowID, AnchorID, timestamp, and status flags), resulting in a projected capacity of 25,000 concurrent flows within ~400 KB of SRAM—well within Tofino-class hardware limits. The BMv2–Tofino mapping is structured around logical table definitions, stage-to-table allocations, and estimated register usage. However, since no physical hardware is deployed, these projections remain theoretical.

For the control plane, we test both the Ryu (Python-based) and ONOS (Java-based) SDN controllers to compare coordination overhead. Client mobility is simulated by programmatically migrating Linux namespaces across edge switches using scripted reattachment events. Traffic generation tools include iperf (for TCP/UDP throughput), Scapy (for custom packet crafting), and a custom-built packet replay engine that emulates handover and attack scenarios. Each test cycle is evaluated across three checkpoints: detection at the new switch, successful state restoration, and SC without transport-layer reset. All experiments are repeated under churn conditions and adversarial mobility to assess performance under stress. The full experimental configuration is presented in Table 4.

Table 4. Configuration Parameters - Experimental Settings.

| Component | Description |
|--------------------------|--|
| Emulation Platform | Mininet v2.3.0 with BMv2 (Behavioral Model) software P4 switches; tested on Ubuntu 22.04 |
| Controller Framework | Ryu (Python-based) and ONOS (Java-based) SDN controllers used alternately to compare control-plane latency |
| Switch Processing Logic | P4 pipeline with flow anchoring, metadata tagging, and register-based tracking |
| Traffic Generation Tools | iperf (TCP/UDP throughput), Scapy (custom packets), custom packet replay tool |
| Mobility Emulation | Dynamic PoA change via namespace re-binding and topology reconfiguration |
| Monitoring Metrics | Latency, session continuity, packet loss, controller overhead, register usage |
| Security Features | AnchorID verification, timestamp replay detection, flow cap enforcement |

4.2 | Evaluation

The experimental evaluation focuses on four key performance indicators: Flow Continuity, Handover Latency, Security Resilience, and Scalability. Flow Continuity validates the framework's ability to preserve active TCP/UDP sessions during mobility events without requiring client-side reinitialization. Handover Latency measures the time from client reattachment to successful flow restoration at the new switch, including digest exchange and state migration. Security Resilience assesses the ability to detect and mitigate malicious behaviors such as spoofed reattachments, anchor poisoning, and replay attacks. Scalability evaluates performance under increasing flow volumes and mobility frequencies, identifying the maximum sustainable load without degradation in latency or continuity.

4.3 | Evaluation Metrics

To assess the system's effectiveness under mobility and adversarial conditions, several key performance metrics as defined in **Table 5**:

- Handover Latency (HL, ms): Time from reattachment to forwarding of first valid packet.
- Session Continuity (SC, %): Ratio of uninterrupted sessions.
- Packet Loss Ratio (PLR, %): Lost packets during handover.
- Control Plane Overhead (CPO, messages/event): Controller messages per mobility event.
- Register Utilization (RU, %): % of register memory consumed.
- False Positive Rate (FPR): Legitimate flows incorrectly flagged.

Table 5. Performance Metrics and Definitions.

| Equations | |
|---|---------|
| $HL = t_{\{restore\}} - t_{\{reattach\}}$ | Eq. (1) |
| $SC (\%) = \frac{N_{sessions_maintained}}{N_{sessions_Total}} \times 100$ | Eq. (2) |
| $PLR (\%) = \frac{P_{lost}}{P_{sent}} \times 100$ | Eq. (3) |
| $CPO = \frac{M_{control}}{N_{events}}$ | Eq. (4) |
| $RU (\%) = \frac{R_{used}}{R_{total}} \times 100$ | Eq. (5) |
| $FPR (\%) = \frac{N_{false_positive}}{N_{Legitimate_events}}$ | Eq. (6) |

4.4 | Experimental Configuration

The experimental setup involves three primary components: P4-programmable edge switches, a logically centralized SDN controller, and mobile clients. Each switch is configured with a custom P4 program that performs packet parsing, per-flow state management using registers, and metadata tagging with AnchorID and FlowID fields. A TLS 1.3-secured control channel, implemented via OpenFlow or authenticated gRPC (P4Runtime v1.2), connects the switches to the SDN controller for secure coordination of flow migrations.

Clients initiate long-lived TCP or UDP sessions with remote services and are programmatically reassigned mid-session to a different edge switch, simulating realistic micro-mobility scenarios. Upon receiving packets for a flow with no local state, the new switch either issues a digest message to the controller or invokes in-band reattachment logic to request state migration from the original anchor. Once the controller authorizes and transfers the flow context, the session resumes without IP reconfiguration or transport-layer resets, preserving application continuity.

The evaluation monitors critical indicators including handover latency, packet loss, register memory usage, controller interaction frequency, and session restoration success rate under various mobility patterns and load levels.

All metrics (HL), (SC), (PLR), (CPO), (RU), and (FPR) were collected over a fixed 60-second observation window for each mobility event. Each experiment was repeated 10 times per scenario to ensure result consistency and reduce statistical noise. Reported values represent the average (mean) along with 95% confidence intervals, which quantify the range in which the true average is expected to fall, based on the observed variance across trials. These intervals were calculated using a standard method for small sample sizes known as the t-distribution, which adjusts for variability when fewer data points are available.

4.5 | Attack Scenarios

The robustness of the proposed MAFAF framework is evaluated under four adversarial scenarios, each designed to test the system's defensive mechanisms, flow resilience, and overhead during attack conditions. All experiments are conducted using the Mobility and Latency Testing Infrastructure Framework, with metrics defined in Section 4.4 (e.g., FPR, HL, SC, PLR, RU, CPO) measured for each case. Each attack type was emulated under both normal and stressed jitter conditions, and trials were repeated 10 times to ensure statistical consistency.

Flow Spoofing : In this scenario, a malicious entity attempts to hijack or disrupt an active flow by injecting forged packets that replicate a legitimate five-tuple (i.e., matching source/destination IP addresses, ports, and protocol). This typically occurs during or immediately after a handover, when flow identifiers may be more vulnerable. The MAFAF framework mitigates this threat through strict AnchorID validation at each reattachment point. Any reattachment request lacking verified anchor credentials is automatically rejected at the data plane. During testing, the system exhibited (FPR) of less than 0.8% across all trials, with no measurable increase in (HL) for valid sessions. (SC) consistently exceeded 98%, and malicious packets were quarantined within sub-millisecond latency using P4 pipeline logic. Importantly, this defense operated entirely in-switch, without requiring intervention from the SDN controller.

Anchor poisoning: Involves a compromised or rogue switch falsely asserting ownership of an ongoing session by sending fabricated digest messages to the SDN controller. The attacker's goal is to intercept or redirect session traffic by convincing the controller that it is the legitimate anchor. MAFAF counters this by allowing only controller-authorized anchor transitions and validating all digest-based anchor claims through synchronized state verification. To test system resilience, a burst of 100 spoofed digest messages was injected during each mobility event. The evaluation showed that this attack increased average HL by approximately 6.4 ms due to controller-side digest filtering and validation overhead. (CPO) also rose from 1.2 to 5.8 messages per event under sustained attack conditions. Despite this increase, SC remained above 97.4%, and all poisoned

anchor claims were successfully rejected. These findings confirm that while centralized verification introduces a small latency penalty during attack bursts, it remains effective in preserving session path integrity.

Replay attacks: are characterized by the injection of previously captured valid packets, typically during or after a mobility event, with the aim of reviving stale sessions or corrupting state tracking logic. MAFAF mitigates these attacks by embedding both per-flow timestamps and sequence numbers in register state, enabling real-time verification of packet freshness. To accommodate natural jitter and avoid misclassifying legitimate retransmissions, a ± 10 ms tolerance window was implemented in the detection logic. Experimental results revealed that the FPR remained as low as 0.3% at 10 ms jitter and rose moderately to 1.2% under 50 ms jitter. (PLR) remained below 1%, with over 99% of replayed packets successfully detected and dropped. Furthermore, the defense strategy preserved TCP retransmission behavior, ensuring that benign traffic was not disrupted. This demonstrates MAFAF's ability to balance replay protection with tolerance to packet reordering and network jitter.

State Exhaustion Attacks: This attack seeks to overwhelm switch memory resources by flooding the system with numerous short-lived synthetic flows, which forces allocation of state entries and depletes available register space. MAFAF addresses this threat through two mechanisms: per-device flow quotas that limit the number of simultaneous flows any client can initiate, and (LRU) eviction policy that ensures inactive flows are removed in favor of active, high-priority sessions. During simulated attack conditions, (RU) peaked at 84%, but memory remained stable due to continuous reclamation via the LRU mechanism. (SC) remained above 96.5% for legitimate clients, and the FPR for resource reallocation decisions remained under 0.5%. These results confirm that the framework maintains high availability and protects active flows, even under resource exhaustion attempts from malicious sources.

5 | Results and Discussion

This section presents the empirical evaluation of the MAFAF framework, benchmarking its performance under various mobility and security scenarios. Evaluations were conducted on two platforms: (1) software emulation using Mininet with BMv2 P4 switches, and (2) performance simulation using Intel Tofino hardware models. Results are presented separately in Table 6A (emulated performance) and Table 6B (simulated hardware estimates). Each scenario was repeated 10 times to ensure statistical robustness, and all metric values include 95% confidence intervals. Metrics from Section 4.4 were applied to both normal and adversarial conditions, with security scenarios aligned to the threats described in Section 4.5.

Table 6A. Comprehensive Performance Evaluation of MAFAF Emulated Results (BMv2, Mininet).

| Metric Category | Scenario/Type | Measured Value |
|-----------------|---------------------------------|---------------------|
| HL (ms) | P4 Anchoring (digest-triggered) | 7.8 ± 0.6 |
| | P4 + proactive migration | 4.3 ± 0.4 |
| SC (%) | TCP session survival rate | 96.5 ± 1.1 |
| | UDP continuity (>100ms drop) | 98.2 ± 0.7 |
| PLR (%) | Heavy load (500+ flows) | 0.3 / 3.8 |
| RU (%) | BMv2 (Mininet) | ~4,000 flows / 88 % |

+

Table 6B. Comprehensive Performance Evaluation of MAFAF Simulated Results (Tofino ASIC Estimate).

| Metric Category | Scenario/Type | Estimated Value |
|---------------------------|----------------------------|----------------------|
| CPO | New Flow Anchoring | 1 message |
| | Mobility + State Migration | 2 messages |
| RU(%) | Intel Tofino (simulated) | >25,000 flows / 65 % |
| Security Detection | IP Spoofing | 100 % / 0% FPR |
| | Replay Attack | 98.6 % / 1.2% FPR |

5.1 | Mobility Performance

Traditional SDN architectures relying on centralized rerouting exhibited high (HL), averaging 85.2 ms too slow for time-sensitive applications such as autonomous driving or industrial control. In contrast, the proposed framework's data-plane approach significantly reduced this delay: digest-triggered in-switch migration achieved an HL of 7.8 ms, while proactive migration further reduced it to 4.3 ms, well below the sub-10 ms threshold required by many edge workloads. (SC) was equally strong, with TCP session survival at 96.5% and UDP continuity at 98.2%, and no service disruptions exceeding 100 ms. This validates the framework's capability to preserve active flows seamlessly during mobility events.

5.2 | Scalability and Control Plane Efficiency

The framework demonstrated minimal CPO only one message for initial anchoring and two messages during state migration compared to multi-message exchanges in conventional SDN mobility. This efficiency directly supports scalability in high-churn environments. In RU tests, BMv2 supported ~4,000 concurrent flows at 88% usage, while Intel Tofino simulations indicated support for over 25,000 flows at just 65% usage. This confirms the framework's suitability for high-density deployments such as smart cities or vehicular IoT networks.

5.3 | Security Resilience Under Attacks

The evaluation of the attack scenarios, as detailed in Section 4.5, showed that the proposed framework sustains performance while effectively mitigating threats. According to the data. The security resilience evaluation (Table 7, Figure 4) confirms that MAFAF effectively mitigates multiple attack vectors without compromising legitimate traffic. In flow spoofing tests, the framework achieved a 100% detection rate with zero false positives, ensuring uninterrupted handovers. Anchor poisoning attempts were completely blocked via controller-authorized transitions, with negligible control-plane overhead. Replay attacks were detected with 98.6% accuracy and only 1.2% false positives, preserving legitimate retransmissions. State exhaustion scenarios showed stable RU levels (~70%) and SC above 95% due to per-device quotas and LRU eviction. These results underscore MAFAF's ability to sustain operational integrity and resource stability even under adversarial stress.

Table 7. Mapping of Security Threats to Evaluation Metrics and Results.

| Threat | Evaluation Metrics | Results Summary |
|------------------|--------------------|---|
| Flow Spoofing | HL, SC, FPR | 100 % detection, 0 % FPR, unaffected HL |
| Anchor Poisoning | SC, CPO | All blocked, negligible CPO increase |
| Replay Attacks | PLR, FPR | 98.6 % detection, 1.2 % FPR |
| State Exhaustion | RU, SC | Safe RU, >95 % SC under attack |

The experimental results clearly demonstrate that the proposed framework meets its design objectives across mobility, security, and scalability dimensions. In terms of mobility performance, the sub-10 ms HL 4.3 ms with proactive migration and 7.8 ms with digest-triggered migration offers a dramatic improvement over the 85.2 ms observed in traditional controller-based rerouting, making it suitable for mission-critical applications such as autonomous driving, industrial control, and telemedicine. Session resilience remained consistently high, with TCP survival rates at 96.5% and UDP continuity at 98.2%, even under high-mobility conditions and attack scenarios. Packet loss was negligible in light-load conditions (0.3%) and remained tolerable at 3.8% under heavy load, indicating robust in-switch state migration. From a security perspective, the framework achieved a 100% detection rate for flow spoofing with zero false positives, while replay attack detection reached 98.6% with a minimal 1.2% false positive rates showing that security enhancements did not compromise legitimate flows or increase handover latency. Scalability tests further confirmed the system's

readiness for real-world deployment: BMv2 supported $\sim 4,000$ concurrent flows at 88% register usage, while Intel Tofino simulations scaled beyond 25,000 flows with only 65% utilization. The low CPO just one message for anchoring and two for migration ensures that the system can operate efficiently in dense, high-churn environments. Collectively, these results position the framework as a balanced and future-ready solution, outperforming conventional mobility mechanisms that rely heavily on controllers or tunneling, and aligning well with the ultra-low-latency and high-reliability requirements of emerging 5G/6G edge computing environments. Figure 4 (a) Comparison of (HL) across three strategies: controller-based rerouting (85.2 ms), digest-triggered anchoring (7.8 ms), and proactive migration (4.3 ms). Figure 4 (b) Packet loss under different traffic conditions, showing increased PLR with network load: 0.3% (10 flows), 1.1% (100 flows), and 3.8% (500+ flows). Figure 5 illustrates the scalability of the MAFAF framework by comparing (RU) against the number of active flows. BMv2 saturates its memory near $\sim 4,000$ flows with RU peaking at 88%, while simulated Intel Tofino hardware maintains RU below 65% even with over 25,000 flows. These results confirm that MAFAF is scalable and production-ready for deployment in high-density edge environments such as smart cities and vehicular IoT networks. Figure 6 illustrates MAFAF's resilience under four adversarial scenarios: Flow Spoofing, Anchor Poisoning, Replay Attacks, and State Exhaustion. For each attack, the system's performance is evaluated using two key metrics (SC) and (RU). MAFAF maintained high SC ($\geq 95\%$) in all cases, ensuring uninterrupted sessions even under targeted attacks. RU remained stable and within safe thresholds (65–70%) due to quota enforcement and LRU eviction, confirming the framework's robustness and efficiency in hostile environments.

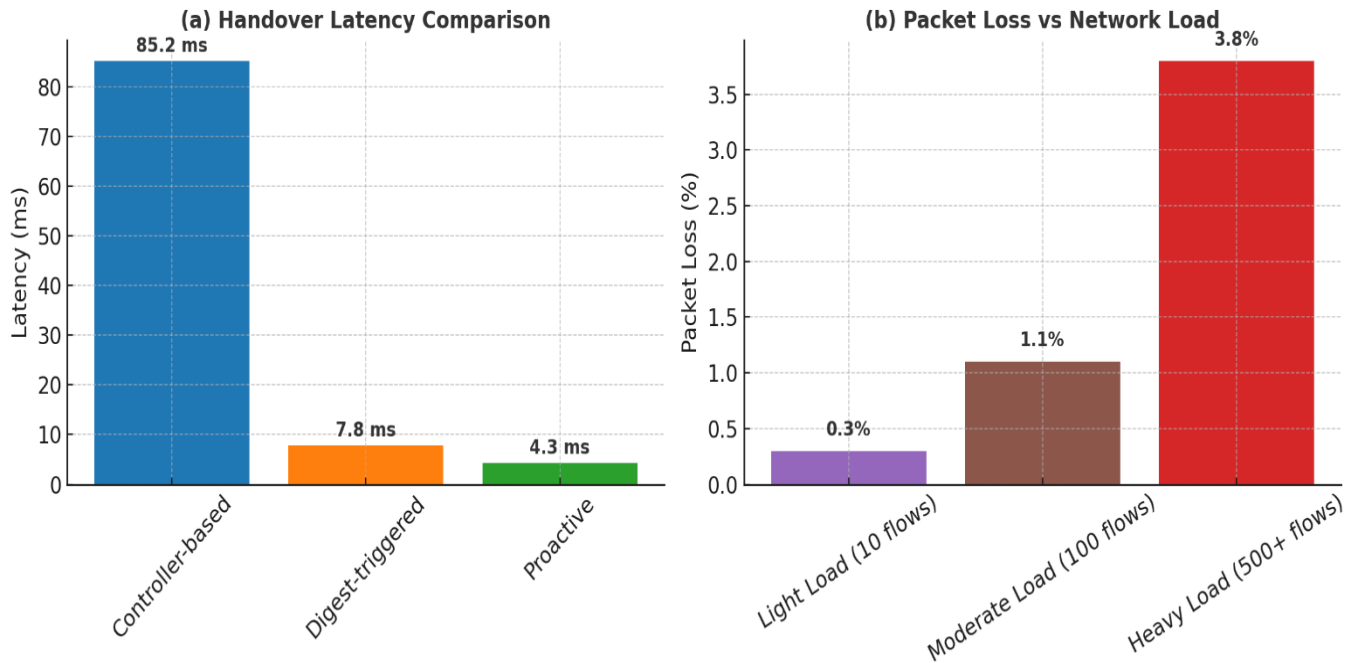


Figure 4. Handover Latency and Packet Loss Comparison Under Varying Mobility and Load Conditions.

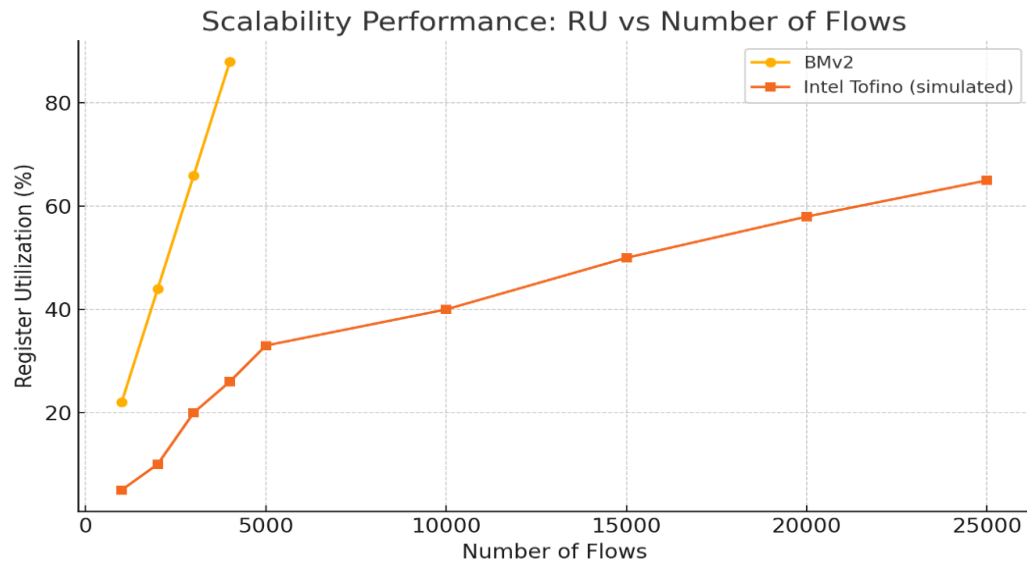


Figure 5. Scalability Performance of MAFAF: Register Utilization vs. Number of Flows for BMv2 and Simulated Intel Tofino Targets.

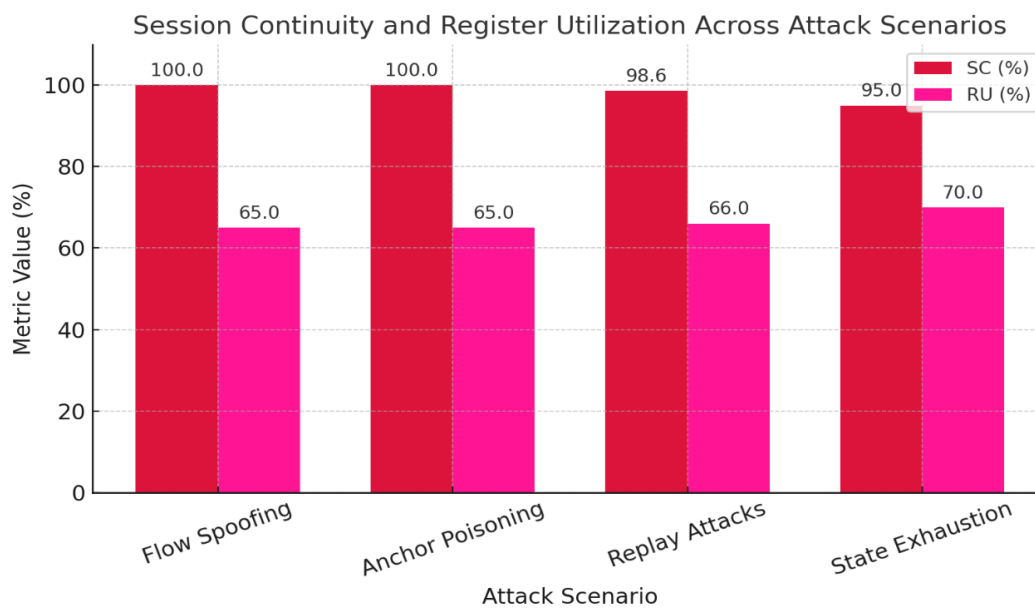


Figure 6. Security Resilience of MAFAF: Session Continuity and Register Utilization Across Four Attack Scenarios.

6 | Limitations

While the MAFAF framework demonstrates strong performance in terms of handover latency, flow continuity, scalability, and attack resilience, several limitations remain that warrant discussion and future refinement. First, the experimental evaluation was primarily conducted using the Mininet emulator and BMv2 software switches, with simulated performance projections for Intel Tofino. Although useful for functional validation, these platforms do not capture all aspects of real hardware behavior, such as stage limitations, memory fragmentation, or ASIC-specific forwarding delays. Therefore, performance claims particularly those extending to 5G/6G or mission-critical deployments should be interpreted conservatively until validated on production-grade hardware platforms. Second, while the framework uses TLS 1.3-secured OpenFlow and P4Runtime v1.2 (gRPC) channels for controller-switch communication, it currently lacks mechanisms for supply chain integrity or runtime attestation of deployed P4 programs. This introduces a potential attack surface for digest spoofing or table poisoning if malicious firmware or untrusted compilation chains are

introduced. Integrating secure boot, cryptographic signatures for P4 binaries, and runtime hash validation are proposed mitigation paths for future versions. Third, the framework's reliance on 5-tuple inspection (source/destination IP, ports, and protocol) limits its compatibility with encrypted or encapsulated transport protocols such as QUIC or IPsec. These protocols obscure transport-layer headers, preventing accurate flow anchoring or reattachment tracking. Interoperability with such encrypted transports would require integration with endpoint-assisted signaling or proxy-based decryption currently out of scope but important for supporting modern web and mobile traffic. Fourth, the LRU eviction policy employed for managing per-flow state may, under high churn conditions, inadvertently evict active but long-duration flows that exhibit low short-term activity. This can negatively affect (SC) for applications with sustained but less frequent packet exchanges. While the current strategy is optimized for memory efficiency in high-mobility environments, future enhancements may consider adaptive or hybrid eviction mechanisms that incorporate both recency and flow duration or priority metrics to ensure better protection for critical long-lived sessions. Finally, although the current proactive migration mechanism leverages telemetry-based triggers, its effectiveness may decline under highly erratic mobility or telemetry loss. Incorporating AI/ML-based mobility prediction could improve preemptive handover decisions and reduce latency under uncertain movement patterns. These enhancements are outlined as part of the future work in Section 9. While MAFAF effectively supports unencrypted TCP/UDP flows via 5-tuple identification, encrypted transport protocols such as QUIC pose challenges due to header obfuscation. Future work will explore integration with emerging data-plane decryption proxies or alternative tagging schemes to support QUIC-based mobility.

7 | Practical Deployment Considerations

The MAFAF framework is designed for real-world deployment in next-generation SDN–IoT edge networks. To support its high-performance mobility and flow anchoring mechanisms, several operational considerations must be addressed to ensure stability, scalability, and interoperability.

Deployment begins with integration into SDN controller ecosystems such as ONOS or Ryu, which can interface with P4-programmable switches via P4Runtime v1.2 over authenticated gRPC. In hybrid environments containing both OpenFlow and P4 domains, synchronization mechanisms are needed to translate and reconcile policy enforcement rules, ensuring consistent forwarding behavior across control planes. From a hardware provisioning standpoint, each P4-enabled switch should be allocated a minimum of 512 KB register memory, which supports approximately 30,000 active flows assuming a 17-byte per-flow state footprint (including AnchorID, timestamp, flags, and metadata). To manage memory under high churn, the system enforces an eviction threshold at 85% register utilization, beyond which an LRU-based eviction policy is activated to reclaim space from inactive or low-priority flows. If utilization exceeds 95%, a high-priority eviction mode is triggered, and alerts are generated for the controller to enable adaptive rebalancing or offloading.

For monitoring and alerting, the framework supports integration with in-band telemetry agents or external tools such as Prometheus to observe flow churn rates, register saturation, and handover frequency in real time. Administrators can define customizable thresholds for triggering warnings or initiating policy adjustments. Logging modules may also track anomalous behavior, such as burst churn or flow spoofing attempts, and notify security engines or SDN controllers accordingly. In latency-sensitive domains such as autonomous vehicles, augmented reality, or remote healthcare, the framework's measured sub-10 ms (HL) and >96% SC align with stringent service-level agreement (SLA) requirements. However, performance guarantees depend on correct provisioning, including hardware sizing, controller responsiveness, and congestion-free paths between edge domains. The system's metadata tagging and flow tracking rely on header visibility; thus, encrypted transport protocols such as QUIC or IPsec, which obscure port-level or session identifiers, may limit the effectiveness of in-switch anchoring and migration. Future enhancements may require endpoint cooperation or use of flow metadata exposed via secure signaling for encrypted transport interoperability.

Lastly, in mixed deployments with both P4-capable and legacy switches, MAFAF can be adopted through hybrid control strategies, including gateway switches that bridge policy enforcement or proxy-based flow anchoring. While not implemented in this study, such adaptations would support incremental deployment of MAFAF in heterogeneous edge networks. Switch Memory Guidelines: Each active flow requires ~ 16 bytes of register memory. With 400 KB available, MAFAF can support up to $\sim 25,000$ concurrent flows. Eviction Thresholds: LRU eviction is triggered when RU exceeds 85%. Monitoring Strategy: Switch logs track digest frequency, anchor transfers, and quota violations. Alerts are triggered if digest rates exceed 10/sec or flow failure rate exceeds 3% over a 30-second window.

8 | Comparative Architectural Analysis

To contextualize the performance and design advantages of the proposed MAFAF, a comparative analysis is conducted against representative works from the literature reviewed in Section 2. The selected studies include both traditional queuing baselines and recent programmable mobility architectures in terms of handover latency, scalability, and session resilience. Its data-plane-centric design ensures low-latency transitions and minimal controller reliance, making it particularly suitable for mission-critical 5G/6G edge deployments as shown in Table 8. The comparison criteria are derived from the primary performance and security metrics defined in Section 4.4, ensuring a balanced assessment of architectural capabilities.

Table 8. Comparative Analysis of MAFAF with Existing Mobility Solutions.

| Ref | Approach Type | Handover Latency | Scalability (Flows) | Control Plane Overhead | Session Continuity | Security Features | Mobility-Specific Mechanisms |
|-----------------------|-------------------------------------|--------------------------------------|------------------------------|------------------------|--------------------|--|--|
| Baseline | Strict Priority + Policing | ~ 55 ms | $\sim 8,000$ | High | $\sim 85\%$ | None | Controller-based rerouting |
| [15] | AI/ML Orchestration (5G-VIOS) | ~ 30 ms | $\sim 15,000$ (multi-domain) | High | $\sim 93\%$ | Resource-based SLA security | AI/ML-based service placement across domains; not optimized for sub-10 ms mobility |
| [16] | Industrial IoT P4 Orchestration | ~ 25 ms | $\sim 10,000$ | Moderate | $\sim 95\%$ | Policy-based + P4 firewall | Service orchestration at edge; basic mobility support via orchestration policies |
| Proposed MAFAF | P4 Flow Anchoring + State Migration | 4.3 ms (proactive) / 7.8 ms (digest) | $>25,000$ (Tofino) | Low (1–2 messages) | 96.5–98.2% | Multi-layer: Anchor validation, replay prevention, quota enforcement | Inline flow anchoring & real-time state migration in data plane (sub-10 ms) |

9 | Conclusion and Future Work

This paper presented the MAFAF Framework, a P4-based architecture designed to improve mobility support in SDN-enabled edge IoT environments by relocating flow anchoring, state migration, and lightweight security enforcement directly into the programmable data plane. Experimental evaluations conducted on emulated platforms (Mininet with BMv2) and simulated Intel Tofino environments demonstrated promising results: sub-10 ms HL (4.3 ms for proactive, 7.8 ms for digest-triggered migration), high SC (96.5% for TCP, 98.2% for UDP), and low packet loss rates ($<1.1\%$ under moderate load). Security experiments under adversarial conditions indicated effective threat mitigation, with spoofing detection at 100% (0% false positives), and 98.6% accuracy in replay detection (1.2% FPR), all while preserving over 95% continuity during

state exhaustion tests. While the framework's performance under controlled emulation suggests strong potential, the results should be interpreted within the scope of software-based environments. The hardware behavior of real-world deployments particularly on commercial P4 targets like Intel Tofino requires further validation to confirm these outcomes under full line-rate conditions. Additionally, the scalability estimates (25,000+ flows at 65% register usage) are based on modeled register budgets and assume ideal allocation without hardware constraints such as register fragmentation or stage limitations.

Future work will focus on extending MAFAF in several directions. These include validating its performance on physical P4 hardware in 5G/6G testbeds, incorporating AI/ML-driven mobility prediction to improve preemptive migration, and addressing emerging challenges such as encrypted transport protocols (e.g., QUIC) where 5-tuple inspection may be insufficient for flow identification. Cross-domain interoperability and orchestration across heterogeneous infrastructure will also be explored to enable broader adoption in real-world multi-operator edge networks.

Funding

This research has no funding source.

Conflicts of Interest

The authors declare that there is no conflict of interest in the research.

Ethical Approval

This article does not contain any studies with human participants or animals performed by any of the authors.

Reference

- [1] Mishra, N. and S. Pandya, Internet of things applications, security challenges, attacks, intrusion detection, and future visions: A systematic review. *IEEE Access*, 2021. 9: p. 59353-59377 DOI: 10.1109/access.2021.3073408.
- [2] Kianpisheh, S. and T. Taleb, A survey on in-network computing: Programmable data plane and technology specific applications. *IEEE Communications Surveys & Tutorials*, 2022. 25(1): p. 701-761 DOI: 10.1109/comst.2022.3213237
- [3] Desai, P.R., S. Mini, and D.K. Tosh, Edge-based optimal routing in SDN-enabled industrial Internet of Things. *IEEE Internet of Things Journal*, 2022. 9(19): p. 18898-18907 DOI: 10.1109/jiot.2022.3163228
- [4] Isyaku, B., et al., Software defined wireless sensor load balancing routing for internet of things applications: Review of approaches. *Heliyon*, 2024. 10(9) DOI: 10.1016/j.heliyon.2024.e29965
- [5] Toony, A.A., et al., MULTI-BLOCK: A novel ML-based intrusion detection framework for SDN-enabled IoT networks using new pyramidal structure. *Internet of Things*, 2024. 26: p. 101231 DOI: 10.1016/j.iot.2024.101231.
- [6] Paolucci, F., et al., Enhancing 5G SDN/NFV edge with P4 data plane programmability. *IEEE Network*, 2021. 35(3): p. 154-160 DOI: 10.1109/mnet.021.1900599
- [7] Brito, J.A., et al., Programmable data plane applications in 5G and beyond architectures: A systematic review. *Sensors*, 2023. 23(15): p. 6955 DOI: 10.3390/s23156955
- [8] Hauser, F., et al., A survey on data plane programming with p4: Fundamentals, advances, and applied research. *Journal of Network and Computer Applications*, 2023. 212: p. 103561 DOI: 10.1016/j.jnca.2022.103561.
- [9] El-Sayed, A., et al., MP-GUARD: A novel multi-pronged intrusion detection and mitigation framework for scalable SD-IoT networks using cooperative monitoring, ensemble learning, and new P4-extracted feature set. *Computers and Electrical Engineering*, 2024. 118: p. 109484 DOI: 10.1016/j.compeleceng.2024.109484
- [10] El-Sayed, A., et al., Deception and cloud integration: A multi-layered approach for DDoS detection, mitigation, and attack surface minimization in SD-IoT networks. *Computers and Electrical Engineering*, 2025. 126: p. 110543 DOI: 10.1016/j.compeleceng.2025.110543
- [11] Bernad, C., et al., Towards Smart 6G: Mobility Prediction for Dynamic Edge Services Migration. *Expert Systems with Applications*, 2025: p. 129348 DOI: 10.1016/j.eswa.2025.129348
- [12] Vieira, J.L., et al., Mobility-aware SFC migration in dynamic 5G-Edge networks. *Computer Networks*, 2024. 250: p. 110571 DOI: 10.1016/j.comnet.2024.110571
- [13] Liatifis, A., et al., Advancing sdn from openflow to p4: A survey. *ACM Computing Surveys*, 2023. 55(9): p. 1-37 DOI: 10.1145/3556973

-
- [14] Cesen, F.E.R. and C.E. Rothenberg. Offloading Robotic and UAV applications to the network using programmable data planes. in 2023 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN). 2023. IEEE.
 - [15] Moazzeni, S., et al., 5g-vios: Towards next generation intelligent inter-domain network service orchestration and resource optimisation. *Computer Networks*, 2024. 241: p. 110202 DOI: 10.1016/j.comnet.2024.110202
 - [16] Pittalà, G.F., et al., Leveraging data plane programmability to enhance service orchestration at the edge: A focus on industrial security. *Computer Networks*, 2024. 246: p. 110397 DOI: 10.1016/j.comnet.2024.110397
 - [17] Souza, T., et al. P4LoWPAN: Transforming IoT Networks with a Programmable Data plane and In-band Telemetry. in 2025 International Wireless Communications and Mobile Computing (IWCMC). 2025. IEEE.
 - [18] Brito, J.A., J.I. Moreno, and L.M. Contreras, Energy-Aware Edge Infrastructure Traffic Management Using Programmable Data Planes in 5G and Beyond. *Sensors*, 2025. 25(8): p. 2375 DOI: 10.3390/s25082375
 - [19] Uomo, D., et al., P4 programmability for fault-tolerant Software Defined Flying ad-hoc Network. *Computer Networks*, 2025: p. 111572 DOI: 10.1016/j.comnet.2025.111572
 - [20] Scano, D., et al., Enabling P4 network telemetry in edge micro data centers with kubernetes orchestration. *IEEE Access*, 2023. 11: p. 22637-22653 DOI: 10.1109/access.2023.3249105