Contents lists available at IdNNS

Indonesian Journal of Cybersecurity and Emerging Risks

Volume 1 ; Issue 1

journal homepage: https://journal.idnns.org/index.php/ijcyber

STUN-Based Detection of Differentiated Services Code Point (DSCP) Mangling in IP Networks

Ahmad Irfan Nabihan bin Shukri[a], Muhammad Zaid bin Mohd Riduan[a] and Adamu Abubakar Ibrahim[a,*]

[a]*Department of Computer Science, Kulliyyah of Information and Communication Technology, International Islamic University Malaysia, Kuala Lumpur, 53100, Malaysia*

ARTICLE INFO

Keywords:
DSCP mangling
Differentiated Services
Quality of Service
STUN
QoS security
traffic classification
network monitoring

ABSTRACT

Differentiated Services Code Point (DSCP) is widely used in IP networks to classify packets and support Quality of Service (QoS) for latency-sensitive applications such as voice, video conferencing, and real-time communication. However, DSCP values may be modified, downgraded, or cleared by intermediate network devices, either because of policy enforcement, misconfiguration, or malicious manipulation. This behavior, known as DSCP mangling, can reduce QoS reliability, weaken traffic classification, and allow unauthorized traffic to obtain or lose priority treatment. This paper proposes and evaluates a STUN-based mechanism for detecting DSCP mangling by embedding transmitted and received DSCP information into STUN messages and comparing the declared DSCP value with the observed value at the receiver. The proposed approach is validated in a controlled IP network environment using multiple traffic patterns and DSCP classes. Experimental results show that the mechanism can identify DSCP modification events accurately while introducing minimal processing overhead. The study demonstrates that STUN can be repurposed as a lightweight active probing method for DSCP integrity verification and can support future QoS security monitoring in real-time network environments.

1. Introduction

Differentiated Services Code Point (DSCP) is a key mechanism in the Differentiated Services (DiffServ) architecture for classifying IP packets and assigning forwarding treatment in IPv4 and IPv6 networks. DSCP is carried in the Differentiated Services field and is used by network nodes to map packets to a Per-Hop Behavior (PHB), enabling scalable traffic differentiation without requiring per-flow state in every router Nichols et al. (1998); Blake et al. (1998). In practice, DSCP supports Quality of Service (QoS) by allowing latency-sensitive and high-priority traffic to receive different treatment from best-effort traffic. QoS mechanisms commonly rely on packet classification, marking, policing, shaping, queueing, bandwidth allocation, and trust boundaries to manage packet delay, jitter, loss, and throughput Cisco (2019); Fortinet (2023).

The importance of DSCP-based QoS has increased with the growth of real-time and multimedia applications, including Voice over IP (VoIP), video conferencing, online learning platforms, cloud collaboration, and Web Real-Time Communication (WebRTC). These applications are sensitive to delay, packet loss, and jitter, making traffic prioritization necessary for maintaining acceptable user experience. WebRTC-related standards recommend DSCP markings for different browser-based audio, video, and data traffic classes, while DiffServ guidance documents describe how traffic classes can be mapped to forwarding treatments such as Assured Forwarding (AF) and Expedited Forwarding (EF) Jones et al. (2021); Black and Jones (2015); Babiarz et al. (2006). AF provides multiple forwarding classes with different drop precedences, whereas EF is designed for low-loss, low-latency, low-jitter traffic such as voice Heinanen et al. (1999); Davie et al. (2002). Additional DiffServ guidance also discusses service-class aggregation and capacity-admitted real-time traffic, showing that DSCP remains important for scalable and controlled QoS deployment Chan et al. (2008); Baker et al. (2010).

Despite its usefulness, DSCP is not always preserved end-to-end. Intermediate routers, switches, firewalls, access networks, or service-provider domains may remark, downgrade, bleach, or otherwise modify DSCP values according to local policies. These modifications can be legitimate when performed by a network operator, but they can also occur because of misconfiguration or malicious manipulation.

*Corresponding author
ipansekeri@gmail.com (A.I.N.b. Shukri); ipansekeri@gmail.com (M.Z.b.M. Riduan); adamu@iium.edu.my (A.A. Ibrahim)
ORCID(s):

Prior Internet measurements have shown that DSCP values can change along network paths and that DSCP remarking behavior varies significantly across networks Custura et al. (2018). Recent IETF guidance also emphasizes that new DSCP assignments must consider real-world remarking and bleaching behavior because a DSCP value selected by a sender may not be the same value observed by a receiver Custura et al. (2023). This creates a practical problem for QoS reliability because applications and administrators may assume that DSCP markings remain intact when, in reality, the network path may alter them.

DSCP manipulation also has security implications. An attacker or non-compliant endpoint may mark ordinary traffic with a high-priority DSCP value to obtain better forwarding treatment, bypass traffic-management policies, or consume resources intended for legitimate real-time traffic. Conversely, critical traffic may be downgraded, reducing the quality of voice, video, or control traffic. Previous research has treated DSCP value modification as an attack-notification problem and proposed mechanisms for identifying abnormal DSCP behavior Alarood et al. (2023). Other studies have explored DSCP-based traffic filtering for attack prevention, DSCP tracking in QoS-enabled networks, and anomaly-based protection for multimedia transmission under attack conditions Malikovich et al. (2021); Liu (2007); Luo and Shyu (2007). These works show that DSCP should be considered not only a QoS feature but also a security-relevant field that requires monitoring and validation.

The challenge addressed in this paper is the detection of DSCP mangling, defined as the modification of DSCP values between packet transmission and reception. Because DSCP is part of the IP header, a receiver can observe the DSCP value that arrives, but it may not know the value originally transmitted by the sender. A detection mechanism therefore needs a way to compare the transmitted DSCP value with the received DSCP value. Session Traversal Utilities for NAT (STUN) provides a lightweight request-response protocol widely used for NAT traversal, connectivity checks, and real-time communication support Petit-Huguenin et al. (2020). STUN is also used within Interactive Connectivity Establishment (ICE), and may operate with Traversal Using Relays around NAT (TURN) when direct connectivity is not possible Keranen et al. (2018); Mahy et al. (2010). An expired IETF Internet-Draft proposed extending STUN with a DSCP value attribute that carries transmitted and received DSCP information, enabling endpoints to detect whether DSCP markings survive across a network path Martinsen and Nandakumar (2016).

Based on this idea, this paper proposes and evaluates a STUN-based DSCP mangling detection mechanism for IP networks. The proposed approach embeds DSCP information into STUN messages and compares the declared transmitted DSCP value with the DSCP value observed at the receiver. If the values differ, the system identifies the packet as having experienced DSCP mangling. The mechanism is evaluated in a controlled network environment using multiple traffic patterns and DSCP classes. The study focuses on determining whether STUN-based active probing can provide a lightweight and accurate method for detecting DSCP modification without requiring deep packet inspection of application payloads.

This research contributes to DSCP security and QoS monitoring in three ways. First, it demonstrates how STUN can be repurposed to verify DSCP integrity along a network path. Second, it provides experimental evidence that DSCP mangling can be detected by comparing transmitted and observed DSCP values. Third, it supports future development of hybrid DSCP monitoring systems that combine active probing, packet fingerprinting, and intelligent traffic classification. Recent work on DSCP-based deep learning classification suggests that machine learning can further enhance packet classification and anomaly detection in future QoS security systems Khan and Ibrahim (2024). Similarly, broader QoS traffic-engineering research shows that scalable and adaptive QoS mechanisms remain necessary for next-generation networks with increasing multimedia and IoT traffic Mazhar et al. (2023).

The remainder of this paper is organized as follows. Section 2 reviews related work on DSCP, QoS, DSCP manipulation, and STUN-based detection. Section 3 describes the proposed detection mechanism and experimental setup. Section **??** presents and analyzes the detection results. Section **??** concludes the paper and discusses future research directions.

2. Literature Review

There are many studies related to DSCP functionality, QoS-based traffic management, and the security vulnerabilities associated with DSCP manipulation. Therefore, this section reviews existing findings, highlights major contributions, and identifies the research gaps addressed by the current study Nichols et al. (1998); Blake et al. (1998); Alarood et al. (2023); Custura et al. (2018).

2.1. DSCP and QoS-Based Traffic Management

Jabbar et al. studied QoS mechanisms in LTE and WiFi networks and showed that QoS support for IP packets can rely on Differentiated Services (DiffServ) and Integrated Services (IntServ) models Jabbar et al. (2019). These models are designed to classify and control network traffic so that service quality can be maintained across IP-based systems. In DiffServ, traffic classification is performed using the Differentiated Services Code Point (DSCP), which occupies six bits of the eight-bit Differentiated Services field in the IPv4 and IPv6 headers Nichols et al. (1998). The DSCP value labels packets with service requirements and allows network devices to apply different forwarding treatments based on traffic priority Nichols et al. (1998); Blake et al. (1998).

DiffServ provides a scalable QoS framework because it groups traffic into behavior aggregates rather than requiring routers to maintain per-flow state Blake et al. (1998). Jabbar et al. also explained that DiffServ can support differentiated services across different network environments by mapping DSCP/IP markings to wireless access categories Jabbar et al.

(2019). This is important because packets may travel through heterogeneous networks, and QoS treatment depends on whether the DSCP marking is interpreted and preserved correctly. Practical QoS deployment also uses DSCP marking and remarking to classify packets, enforce policies, and ensure that critical traffic receives suitable priority during congestion Cisco (2019); Babiarz et al. (2006).

2.2. DSCP Manipulation and Security Vulnerabilities

Although DSCP is useful for QoS, it can be exploited by irresponsible users or attackers. A sender may spoof or modify DSCP values to mark ordinary traffic as high-priority traffic, allowing it to gain unauthorized QoS treatment when entering the network Alarood et al. (2023); Malikovich et al. (2021). DSCP values may also be changed by intermediate routers, switches, or service-provider domains during transmission. Custura et al. showed that DSCP values are not always preserved across Internet paths, while RFC 9435 highlights that DSCP remarking and bleaching are practical issues that must be considered when assigning or using DSCP values Custura et al. (2018, 2023). Therefore, DSCP values must be inspected from source to destination to determine whether the original packet marking has been modified.

Alarood et al. demonstrated DSCP modification attacks in a simulated environment and proposed a notification method for detecting changes in DSCP values Alarood et al. (2023). Their method represents attack and non-attack entries using binary vectors, where each vector corresponds to a packet and indicates whether the packet is classified as malicious or normal Alarood et al. (2023). The detection mechanism compares the inspected DSCP value of each packet against the expected value. If the received DSCP value does not match the value originally assigned at the source, the packet is flagged and an alert is generated Alarood et al. (2023). This work is closely related to the current study because both approaches focus on identifying unauthorized DSCP changes, but the current research uses a STUN-based detection mechanism to compare transmitted and received DSCP values.

The ability to modify DSCP values creates a risk for both security and service quality. For example, real-time voice or video traffic may be marked with a DSCP value intended for low-delay treatment, but this value may be downgraded or removed along the path, causing the traffic to receive lower priority service Jones et al. (2021); Black and Jones (2015); Custura et al. (2018). Conversely, non-critical traffic may be marked with a high-priority DSCP value to obtain unfair access to network resources Alarood et al. (2023); Malikovich et al. (2021). These cases show that DSCP manipulation can affect both QoS performance and network policy enforcement.

2.3. DSCP Filtering, Tracking, and QoS Protection

Malikovich et al. proposed a packet filtering method based on DSCP values to improve the security of DHCP communications and reduce the risk of DHCP starvation attacks Malikovich et al. (2021). Their study emphasized that DSCP values can be used to classify and limit traffic as part of a network security policy. The authors presented a flowchart-based filtering algorithm that identifies and manages packets according to DSCP values, thereby reducing the possibility of network attacks Malikovich et al. (2021). However, their method is mainly focused on DHCP security, meaning that further research is needed to explore how DSCP-based filtering and detection can be applied to other network services and traffic types.

Liu studied DSCP marking in a QoS-enabled triple-play IP network and showed that DSCP tracking is important for networks that carry voice, video, and data services together Liu (2007). This work is relevant because it demonstrates the need to monitor DSCP markings in practical QoS environments. However, Liu's work focuses on tracking DSCP markings in a managed QoS network, while the current study focuses on detecting whether DSCP values are modified between sender and receiver.

Luo and Shyu addressed multimedia transmission over the Internet and proposed a differentiated-service protection framework for preserving QoS during traffic anomalies and Denial-of-Service (DoS) attacks Luo and Shyu (2007). Their framework uses anomaly detection and resource management to protect legitimate multimedia traffic when malicious or suspicious traffic is detected. When an attack is identified, the system adjusts resource allocation to reduce packet loss and maintain better multimedia quality Luo and Shyu (2007). This study shows that early detection and adaptive resource management can reduce the impact of attacks on multimedia QoS. It is relevant to the current research because DSCP mangling can also reduce the QoS of legitimate real-time traffic if priority markings are modified or removed.

2.4. Research Gap

The reviewed literature shows that DSCP is widely used for QoS classification and traffic prioritization, especially in DiffServ-based networks Nichols et al. (1998); Blake et al. (1998); Jabbar et al. (2019). Previous studies have also shown that DSCP values may be modified, spoofed, filtered, or tracked for security and QoS purposes Alarood et al. (2023); Malikovich et al. (2021); Liu (2007); Custura et al. (2018). However, existing approaches do not fully address a lightweight endpoint-based mechanism for comparing the DSCP value transmitted by the sender with the DSCP value observed by the receiver. Therefore, this research focuses on STUN-based DSCP mangling detection to verify whether DSCP markings remain unchanged across the network path.

3. Methodology

The research design for this study is based on an exploratory and experimental approach. It aims to investigate the phenomenon of DSCP mangling across network infrastructures and to evaluate different strategies for detecting DSCP value modifications. DSCP is selected as the main object of inspection because it is carried in the IP header and is used by the DiffServ architecture to classify packets and determine their forwarding treatment Nichols et al. (1998);

Blake et al. (1998). Since DSCP values may be remarked, downgraded, or cleared by intermediate network devices, this study focuses on comparing the DSCP value assigned at the source with the DSCP value observed at the destination Custura et al. (2018, 2023).

This study employs an experimental simulation-based approach to evaluate the proposed DSCP inspection and authentication algorithm. The simulation environment replicates real-world traffic flow and potential DSCP manipulation across a controlled IP network using GNS3. This approach is suitable because previous studies have shown that DSCP modification can occur in network paths and may affect QoS reliability and security Custura et al. (2018); Alarood et al. (2023). The experiment focuses on the behaviour of DSCP values, authentication, and policy enforcement through programmable network elements. QoS policy enforcement is important because routers and switches commonly use DSCP marking, remarking, policing, and classification to manage different traffic classes Cisco (2019); Babiarz et al. (2006).

The experiment is conducted within the GNS3 platform because it provides a flexible virtual network environment for integrating routers, firewalls, and virtual machines to model routing and traffic-inspection scenarios. In this environment, Scapy is used to craft packets containing selected DSCP values. These crafted packets allow the experiment to observe whether the DSCP value remains unchanged or is modified while passing through the simulated network. This process supports the main detection objective, which is to identify whether the DSCP value transmitted by the sender is the same as the DSCP value received at the destination. The same comparison principle is used in previous DSCP attack-notification work, where inspected DSCP values are compared against expected values to detect abnormal modification Alarood et al. (2023). The STUN-based part of this methodology is also supported by the DSCP mangle detection concept, where STUN messages can carry DSCP information so that endpoints can compare transmitted and received DSCP values Martinsen and Nandakumar (2016); Petit-Huguenin et al. (2020).

3.1. Network Scenarios

Using GNS3 and the provided virtual machines, several network scenarios were designed to investigate and analyse the behaviour of DSCP values across the network for data collection purposes. These topologies mimic real-world conditions where packets pass through multiple network nodes that may inspect, preserve, remark, or alter DSCP values before reaching their destination. This is important because DSCP is carried in the IP header and is used by DiffServ-capable nodes to classify packets and apply forwarding treatment Nichols et al. (1998); Blake et al. (1998). Previous studies have also shown that DSCP values may be modified across network paths, while recent DSCP guidance highlights that remarking and bleaching are practical issues in real deployments Custura et al. (2018, 2023).

Data collection was performed by capturing packets generated from the developed script. The script generates packets with selected DSCP values and sends them to the destination machine. After the packets are transmitted, analysis is performed to observe system behaviour, memory allocation, video quality, and packet-capture results. Packet-capture analysis is used to determine whether the DSCP value assigned at the source remains unchanged or is modified before reaching the destination. This comparison is consistent with DSCP attack-notification research, where inspected DSCP values are compared with expected values to identify abnormal modification Alarood et al. (2023).

3.1.1. Scenario 1: Examining DSCP Functionality in QoS-Based Traffic Classification

The objective of Scenario 1 is to examine the functionality of DSCP in QoS-based traffic classification. This experiment involves generating different traffic types with specific DSCP values using Scapy. The generated packets are then analysed under both normal and congested traffic conditions. In this setup, the generator node sends DSCP-marked packets, while the analyser node captures and analyses the received packets. This scenario represents normal DSCP operation, where DSCP markings are expected to support traffic classification and QoS treatment.

DiffServ uses DSCP values to group packets into behaviour aggregates and apply different Per-Hop Behaviours (PHBs), enabling scalable QoS management Blake et al. (1998). Practical QoS deployment also uses DSCP marking, classification, policing, and queueing to manage traffic priority across routers and switches Cisco (2019); Babiarz et al. (2006). Under congestion, DSCP becomes more important because different traffic classes compete for bandwidth, delay, and forwarding priority Fortinet (2023); Mazhar et al. (2023). For example, Expedited Forwarding (EF) is intended for low-loss, low-latency, and low-jitter traffic, while Assured Forwarding (AF) provides multiple forwarding classes and drop-precedence levels Davie et al. (2002); Heinanen et al. (1999).

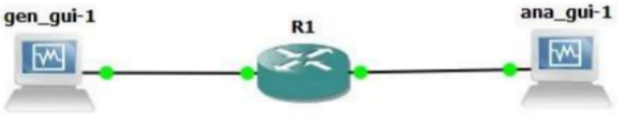

Figure 1: Topology for DSCP functionality and mangling simulation.

3.1.2. Scenario 2: DSCP Mangling

Scenario 2 is designed to simulate DSCP mangling through two approaches: source-based spoofing and in-transit tampering. In the first case, traffic is crafted to spoof a high-priority DSCP value and flood the network with these packets. For example, bulk data traffic may be marked with a high-priority DSCP value such as EF 46, even though the traffic does not represent a legitimate real-time service. This scenario models an irresponsible or malicious endpoint attempting to obtain unauthorized QoS priority by marking ordinary traffic as high-priority traffic Alarood et al. (2023); Malikovich et al. (2021).

In the second case, DSCP mangling occurs through in-transit tampering, where an intermediate router modifies the DSCP field after the packet leaves the source. This scenario represents a network path where DSCP values may be remarked, downgraded, or cleared before reaching the destination. Such behaviour may occur because of provider policy, router configuration, or unintended DSCP bleaching. This scenario is supported by prior Internet measurement work showing DSCP modification pathologies and by RFC guidance explaining that DSCP remarking and bleaching must be considered in real deployments Custura et al. (2018, 2023).

The purpose of Scenario 2 is to determine whether the proposed detection mechanism can identify a mismatch between the DSCP value transmitted by the sender and the DSCP value observed at the receiver. If the transmitted and received DSCP values are different, the packet is classified as DSCP-mangled. This scenario provides the abnormal traffic condition required to evaluate the proposed STUN-based, fingerprint-based, and hybrid DSCP mangling detection methods Martinsen and Nandakumar (2016); Petit-Huguenin et al. (2020).

3.2. Technical Implementation

This section implements and evaluates two distinct algorithm-based methodologies for DSCP mangling detection: STUN-based (active) and fingerprint-based (passive), as well as a hybrid approach that combines both methods.

3.2.1. STUN-Based DSCP Mangling Detection

The main purpose of STUN is to allow a device to identify its public IP address and determine the type of NAT running on its gateway Petit-Huguenin et al. (2020). STUN also helps the device learn which external port has been assigned to its internal connection. Because not all NAT configurations perform port translation, STUN is important for establishing reliable external communication. STUN can also act as a keep-alive mechanism in Restricted and Port-Restricted Cone NAT setups by periodically refreshing NAT bindings before they expire Galea (2015). STUN uses a client-server model, where the server typically listens on both TCP and UDP on port 3478. A client sends a request to the server, which may also instruct the client to perform additional tests using alternate IP addresses or ports Petit-Huguenin et al. (2020).

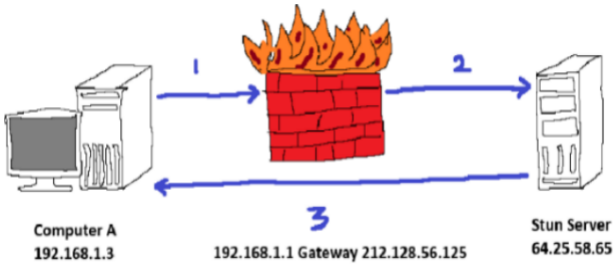

Figure 2: STUN exchange flow.

A typical STUN exchange works as follows. First, a device sends a STUN request from inside its network through its gateway to the external STUN server. Next, the gateway forwards the request to the server and applies NAT translation, which may include changing the source port. Finally, the server responds with the public IP address and port from which the request was observed. With this information, the device can inform external peers where to send return traffic, enabling the establishment of a UDP session even when both sides are behind NAT Petit-Huguenin et al. (2020); Galea (2015). STUN is commonly used together with ICE for connectivity establishment and with TURN when direct communication is not possible Keranen et al. (2018); Mahy et al. (2010).

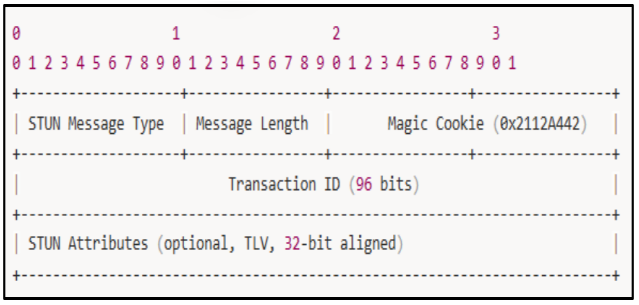

Figure 3: Original STUN header (20 bytes).

The STUN-based method repurposes the Session Traversal Utilities for NAT (STUN) protocol to detect whether the Differentiated Services Code Point (DSCP) value in an IP packet header has been altered as it travels from a client to a server Martinsen and Nandakumar (2016). Instead of analysing all network traffic, this method uses dedicated, lightweight STUN probe packets sent through the network path. By embedding the original DSCP value in a custom STUN attribute and having the server report back the DSCP value it observed, the client can compare the two values and identify any unauthorised changes Martinsen and Nandakumar (2016).

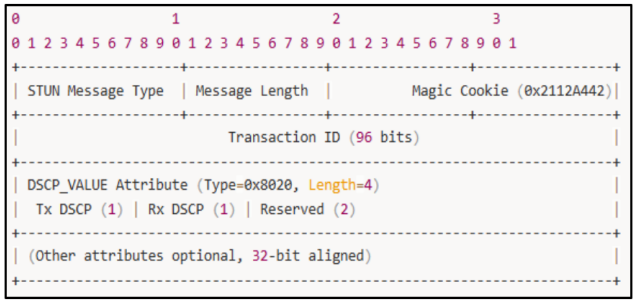

Figure 4: Lab DSCP-STUN header (20 bytes + custom attribute).

The STUN-based detection process consists of four steps. First, the client generates a STUN Binding Request message and sets the desired DSCP value in the IP header's Type of Service (TOS) field using IP.TOS = DSCP $\ll$ 2. It populates the custom `DSCP_VALUE` attribute with the transmitted DSCP value (Tx_{DSCP}) and sets the received DSCP field (Rx_{DSCP})

to zero. The client then sends this specially crafted STUN packet towards the server. This custom attribute follows the DSCP mangle detection concept proposed by Martinsen and Nandakumar Martinsen and Nandakumar (2016).

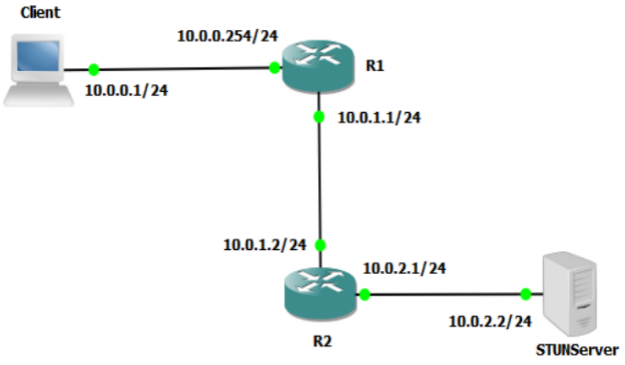

Figure 5: STUN-based mangling detection topology.

Second, the packet travels through the network and passes through intermediate routers (R1, R2, etc.). Intermediate devices may have QoS policies that modify the DSCP value in the IP header Cisco (2019); Babiarz et al. (2006). For example, EF (46) might be changed to AF11 (10). As a result, the IP header's DSCP value is altered, but the custom `DSCP_VALUE` attribute inside the STUN payload remains unchanged. This is the core mechanism that makes the detection possible, because DSCP remarking only affects the IP header and does not modify the application payload Custura et al. (2018, 2023).

Third, the server receives the STUN request and extracts the Rx_{DSCP} value by reading the `IP.TOS` field from the received packet's header. This is the potentially mangled value. The server also reads the original Tx_{DSCP} from the `DSCP_VALUE` attribute in the request payload. The server then generates a STUN Binding Response and populates the `DSCP_VALUE` attribute in the response with both the Tx_{DSCP} (echoed from the request) and the Rx_{DSCP} (the value it observed) Martinsen and Nandakumar (2016).

Lastly, the client receives the STUN Binding Response and parses the `DSCP_VALUE` attribute to obtain both the original Tx_{DSCP} and the Rx_{DSCP} reported by the server. The client then applies the following comparison logic:

- If $Tx_{DSCP} = Rx_{DSCP}$: the DSCP value was preserved along the path. No mangling is detected.
- If $Tx_{DSCP} \neq Rx_{DSCP}$: the DSCP value was changed between the client and server. Mangling is detected.

3.2.2. *Fingerprint-Based DSCP Mangling Detection*

In contrast to the active probing method used by the STUN-based approach, the fingerprint-based detection mechanism functions as a passive network-based monitoring system. This approach does not rely on dedicated probe packets or require cooperation from end-hosts. Instead, it operates by deploying Internal Security Devices (ISDs) at strategic ingress and egress boundaries of a monitored network domain. ISDs were chosen instead of end devices because border devices are strategically located where traffic enters and exits the network. This gives the system complete coverage of packets that traverse the network and makes it ideal for observing potential DSCP changes across all paths Alarood et al. (2023); Liu (2007).

Table 1: Router R2 DSCP mangling policy configuration.

Class Name	Original DSCP	Original Value	New DSCP	New Value
DSCP-EF	EF	46	AF11	10
DSCP-AF41	AF41	34	CS3	24
DSCP-CS3	CS3	24	AF41	34
DSCP-AF21	AF21	18	AF11	10
DSCP-AF11	AF11	10	BE (default)	0

The mangling policy in Table 1 shows how Router R2 systematically alters DSCP values. For example, EF traffic (DSCP 46) is downgraded to AF11 (DSCP 10), and AF11 traffic (DSCP 10) is further downgraded to Best Effort (DSCP 0). These configurations simulate real-world DSCP remarking that may occur because of provider policy, misconfiguration, or deliberate manipulation Custura et al. (2018, 2023). The selected DSCP classes represent commonly used DiffServ service categories, including Expedited Forwarding for low-latency traffic Davie et al. (2002), Assured Forwarding for traffic with differentiated drop precedence Heinanen et al. (1999), and Class Selector values used for backward compatibility Babiarz et al. (2006).

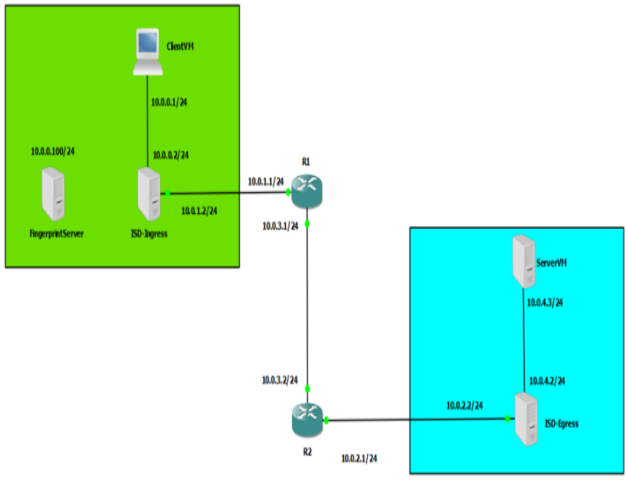

Figure 6: Fingerprint-based mangling detection topology.

The core principle involves generating a unique cryptographic fingerprint for each packet entering the network. This fingerprint serves as a stable identifier that allows the system to track the packet across the network path and compare the original DSCP value at ingress with the DSCP value observed at egress. Any inconsistency between these values is flagged as a DSCP mangling event Alarood et al. (2023). This method provides comprehensive and real-time visibility into all user traffic that traverses the domain without modifying

the packets themselves, making it a suitable tool for network integrity monitoring.

As packets enter the network from the client (10.0.0.1), the ISD-Ingress passively captures them. For each packet, a unique fingerprint is generated using a cryptographic hash function (MD5) over a concatenation of immutable header fields and the payload: source IP address, destination IP address, IP protocol number, and the packet payload. The fingerprint function is defined as:

$$F(P) = H\left([s_1\ s_2\ s_3\ s_4] \| [d_1\ d_2\ d_3\ d_4] \| [0\,0\,0\ p] \| [b_1\ b_2\ b_3 \dots b_n]\right) \tag{1}$$

where $s_1 \dots s_4$ are the source IP octets, $d_1 \dots d_4$ are the destination IP octets, p is the IP protocol number, $b_1 \dots b_n$ is the packet payload, and H is the MD5 hash function. DSCP is deliberately excluded from the fingerprint calculation because it is the field under inspection.

The packet continues its normal path through the network and traverses intermediate routers (e.g., R1, R2). As configured in the testbed, Router R2 acts as the mangling point, applying a QoS policy map that systematically alters DSCP values using the same configuration as in the STUN-based detection scenario Cisco (2019). When the packet arrives at the ISD-Egress, the same fingerprint is recalculated using the identical method. The system queries the Fingerprint Server with this fingerprint to retrieve the stored original DSCP value. The DSCP value currently in the packet's header (observed at egress) is then compared with the original DSCP value retrieved from the database. If $DSCP_{ingress} \neq DSCP_{egress}$, a mangling event is logged and an alert is generated. The alert includes details such as the fingerprint, original DSCP, altered DSCP, and source/destination IP addresses.

Although using a simple packet identifier such as the IP Identification field could be considered for fingerprinting, this field is only 16 bits, providing a maximum of 65,536 unique values. In moderate-to-high-traffic networks, this limited space is exhausted rapidly, leading to a high probability of collision where two different packets share the same identifier. This makes it impossible to reliably and uniquely identify a specific packet for comparison, rendering the detection system unreliable. The fingerprint method used in this study employs a 128-bit MD5 hash over multiple immutable fields, which makes collision computationally infeasible for all practical purposes.

3.2.3. *Hybrid Detection (Combination of STUN and Fingerprint-Based)*

The hybrid method combines the STUN-based and fingerprint-based approaches to evaluate detection efficiency across network paths. This method is designed to leverage the strengths of both active probing and passive monitoring in a single detection framework Martinsen and Nandakumar (2016); Alarood et al. (2023).

First, the client generates a unified packet containing both fingerprint data and STUN protocol attributes in a single payload. The payload includes:

- **Fingerprint data**: packet identifiers, timestamps, and entropy-based markers.
- **STUN attributes**: the declared intended DSCP value (both Tx_{DSCP} and Rx_{DSCP}).

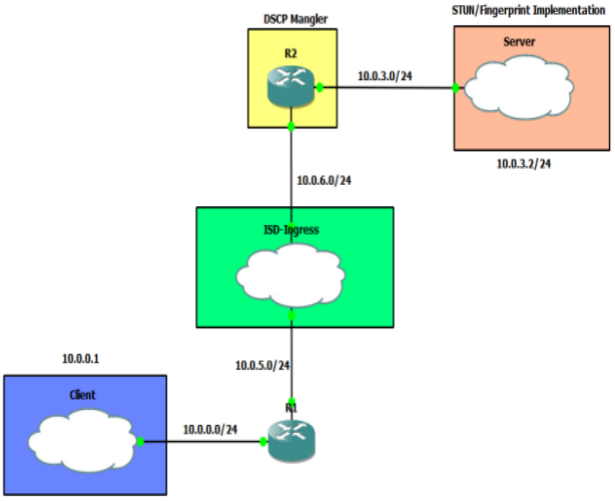

Figure 7: Hybrid mangling detection topology.

After the packet travels through the network, intermediate devices may modify the DSCP field in the IP header while leaving the payload intact Custura et al. (2018, 2023). The server then extracts both the fingerprint data and the STUN attributes from the same packet payload. The fingerprinting engine analyses the payload characteristics to confirm that the received packet is the exact packet sent by the client. The STUN parser extracts the original DSCP value that was explicitly declared when the packet was created. The system also reads the DSCP value from the received packet's IP header. Lastly, the declared DSCP from the STUN attribute is compared against the received DSCP from the IP header. Any difference indicates that DSCP manipulation has occurred.

The ingress device is positioned in the middle of the network path to serve as a controlled policy enforcement point that guarantees DSCP modifications will occur, creating an ideal experimental environment for testing the detection system. In real-world networks, intermediate devices such as firewalls, ISP routers, load balancers, and QoS policy enforcement points often rewrite DSCP values as traffic crosses administrative boundaries or policy domains Custura et al. (2018, 2023); Babiarz et al. (2006). This scenario is simulated by placing the ingress device between R1 and R2. This placement ensures that packets leaving the client with DSCP markings will arrive at the server with potentially altered values, allowing the study to validate whether the hybrid detection method can identify how those markings were changed.

Thus, the ingress device is not merely another hop but a deliberately introduced variable that transforms a simple path into a controlled DSCP modification environment. This design allows the hybrid method to be evaluated under conditions where both the STUN-based comparison and

the fingerprint-based comparison can independently verify whether DSCP mangling has occurred Martinsen and Nandakumar (2016); Alarood et al. (2023).

4. Result and Analysis

The analysis focuses on five traffic patterns: Sequential, Random Uniform, VoIP Heavy, Bulk Data, and Burst. These patterns were used to assess detection accuracy, system performance, and overall operational reliability. The active STUN-based method embeds DSCP markers into STUN messages to probe the network path, while the passive fingerprint-based method generates a unique packet identifier at the ingress point and verifies it at the egress point to compare the original and observed DSCP values.

4.1. STUN-Based Detection Results

The STUN client generated traffic across five distinct patterns, each designed to simulate different network conditions. A custom `DSCP_VALUE` attribute with type `0x8020` was embedded in STUN Binding Requests and Responses to carry the transmitted (Tx) and received (Rx) DSCP values. The intermediate router R2 was configured with a policy map to mangle DSCP values and provide the ground truth for detection validation.

Table 2: Cross-pattern performance summary for STUN-based detection.

Traffic Pattern	Total Packets	Response Rate	Mangling Rate	Avg. RTT (ms)
Sequential	60	91.7%	83.6%	74.6
Bulk	116	94.8%	87.3%	74.8
VoIP	177	98.3%	98.3%	77.3
Random	70	92.9%	75.4%	74.8
Burst	40	85.0%	76.5%	78.0

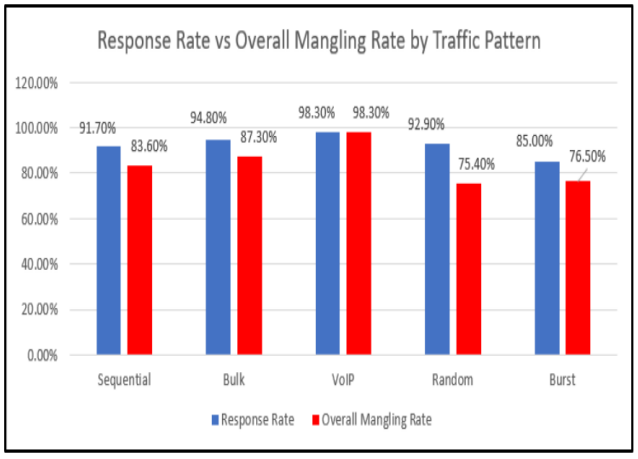

Figure 8: Bar chart of response rate versus overall mangling rate by traffic pattern.

Table 2 provides a summary of the system performance across all tested scenarios. The key finding is the stability of the average round-trip time (RTT), which remains between 74 ms and 78 ms. This indicates that the STUN-based detection method adds minimal and consistent latency. The variation in overall mangling rate, ranging from 75.4% to 98.3%, is not a measure of detection accuracy. Instead, it represents the proportion of non-Best-Effort traffic sent in each traffic pattern. The system correctly identified mangled packets in all tested cases.

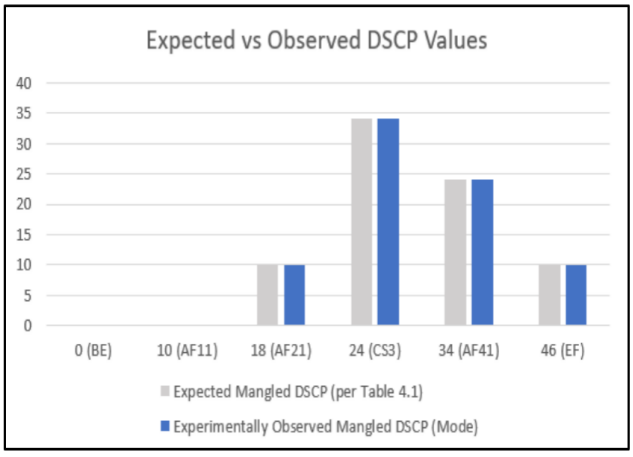

Figure 9: Bar chart of expected versus observed DSCP values.

The final validation cross-references the expected outcome from the configured policy with the experimentally observed result. The match across all tested DSCP values confirms 100% detection accuracy with zero false positives and zero false negatives. These results validate the core hypothesis that a STUN-based active probing mechanism can reliably detect DSCP mangling events along a network path.

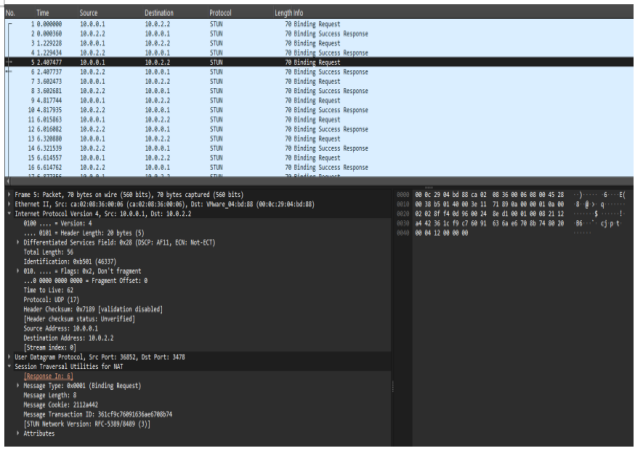

Figure 10: Packet 5 STUN Binding Request from the sequential traffic pattern.

Figure 10 shows a representative STUN Binding Request packet captured on the server interface `ens33`, sent from the client at `10.0.0.1`. This packet was captured during the sequential traffic pattern simulation. The IP header's TOS field shows `0x28`, which represents AF11 with DSCP value 10, while the custom attribute declares the transmitted DSCP value as `0x12`, which represents AF21 with DSCP value 18.

Table 3
Consolidated detection performance across all traffic patterns for fingerprint-based detection.

Traffic Pattern	Total Packets	Detected	Expected	Accuracy	False Positives	Avg. Processing Rate (pkts/s)
Sequential	100	83	83	100%	0	9.4
Random Uniform	100	84	84	100%	0	3.5
VoIP-Heavy	100	94	94	100%	0	10.7
Bulk Data	100	55	55	100%	0	9.2
Bursty	100	60	60	100%	0	10.4

This shows that the DSCP value was mangled from AF21 to AF11, which aligns with the R2 policy map.

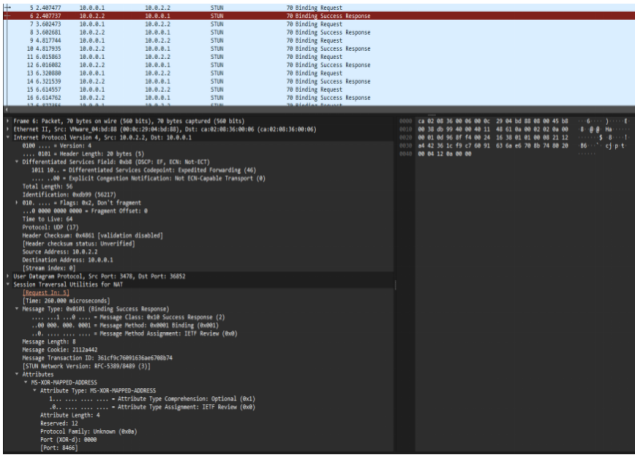

Figure 11: Packet 6 STUN Binding Success Response from the sequential traffic pattern.

Figure 11 shows the corresponding STUN Binding Response, where the server fills in the observed *Rx* DSCP value. The result confirms that the server successfully inserted the observed DSCP value into the response and returned it to the client. The client then used this information to compare the transmitted and received DSCP values. The capture also shows that attribute type `0x8020` is correctly included in both request and response packets. The 4-byte value structure, consisting of *Tx* DSCP, *Rx* DSCP, and reserved fields, was properly implemented, and the server successfully parsed the request attribute and filled the *Rx* DSCP field.

4.2. Fingerprint-Based Detection Results

The fingerprint-based lab topology consisted of a client VM at `10.0.0.1`, an ingress Internal Security Device (ISD) at `10.0.0.2`, two routers (R1 and R2), an egress ISD at `10.0.2.2`, a server VM at `10.0.4.3`, and a central fingerprint server at `10.0.0.100`. Router R2 was configured as the DSCP mangling point using the same policy map as the STUN-based experiment. The system was evaluated using detection accuracy, mangling rate, false positive rate, processing rate, and system overhead.

The fingerprint-based system achieved 100% detection accuracy across all five traffic patterns with zero false positives. This confirms the reliability of the MD5 fingerprinting mechanism and the ingress-egress comparison logic in a controlled environment. The observed mangling rate directly correlated with the traffic mix, ranging from 55% to 94%. This demonstrates that the system's detection capability is independent of traffic patterns. The average packet processing rate remained between 3.5 and 10.7 packets per second. The lower rate in the Random Uniform test may be attributed to system latency during random packet generation and capture. Overall, the processing overhead is minimal, indicating that the system is suitable for real-time monitoring.

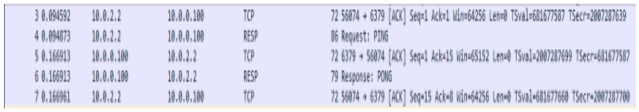

Figure 12: Redis initialization sequence.

To prove that the fingerprint-based detection system operates as designed, packet-level analysis was conducted by tracing specific packets through the complete detection pipeline. This analysis validates the algorithmic integrity from fingerprint generation to Redis database interaction.

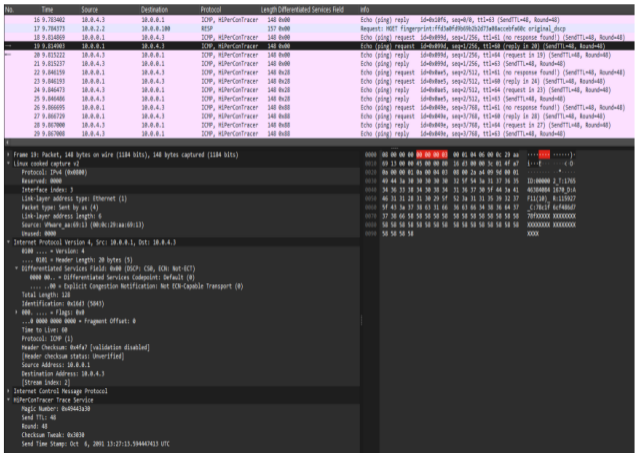

Figure 13: ICMP request-reply pair for packets 19 and 20.

The initial TCP handshake, followed by the Redis `PING/PONG` exchange, confirms successful connectivity between the ISD-Egress device and the central fingerprint server. This establishes the operational foundation for subsequent fingerprint queries.

Packet 19 is an ICMP request with a DSCP value of CS0, which corresponds to DSCP 0. When the MD5 fingerprint is recalculated using the algorithm defined in Chapter 3, the resulting fingerprint is `87155d2fc72eda46f377a81edd4df09c`. By examining the RESP protocol traffic, the same fingerprint was previously queried in Frame 46. The Redis response in Frame 53 returned an original DSCP value of 10, which

corresponds to AF11. This confirms that the fingerprint calculated from Packet 19 matches the fingerprint stored during ingress. The fingerprint server confirms that the original DSCP value was AF11, while Packet 19 shows CS0. Therefore, the DSCP value was mangled from AF11 to Best Effort, which aligns with R2's mangling policy.

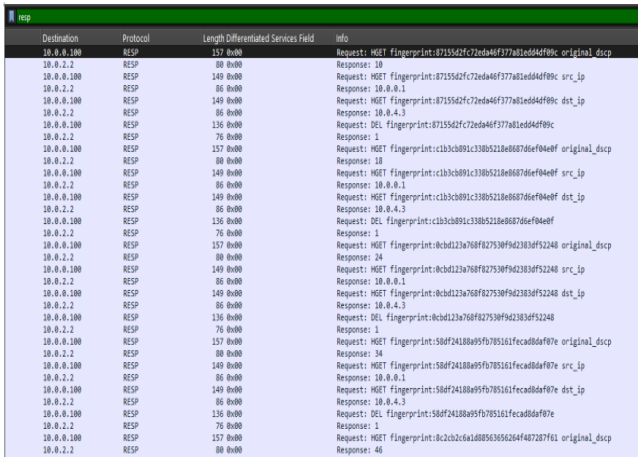

Figure 14: Fingerprint verification for `87155d2fc72eda46f377a81edd4df09c`.

4.3. Hybrid Detection Results

In the hybrid mechanism, TCP traffic presented a more complex detection environment because of its connection-oriented nature, sequencing, and acknowledgement model. These characteristics introduced timing variations that affected detection consistency. The hybrid method achieved 95% detection success for TCP traffic, with an average processing time of 62.65 ms, which is slightly higher than UDP. In contrast, UDP traffic is connectionless and stateless, allowing more straightforward payload embedding. As a result, UDP achieved 100% detection accuracy with a lower average processing time of 60.25 ms.

Table 4: Mangling rate comparison between UDP and TCP traffic.

Type of Traffic	Mangling Rate
UDP	68%
TCP	52%

UDP traffic experienced the highest mangling rate at 68%, showing the importance of detecting DSCP modification in connectionless traffic. This higher rate may be due to UDP's association with real-time applications such as VoIP and video streaming, where networks often enforce strict policy controls to prevent bandwidth abuse. UDP also showed more predictable DSCP modification patterns, primarily downgrading high-priority traffic to Best Effort.

TCP traffic exhibited a more moderate mangling rate of 52%, with a distinct pattern of selective modification. Network policies appeared more conservative with connection-oriented traffic, likely because of TCP's built-in congestion control and its association with critical business applications. These findings show that different traffic types may experience different QoS treatment patterns in real network environments. Therefore, a DSCP mangling detection system is necessary to verify whether DSCP markings are preserved or modified across the network path.

4.4. Overall Analysis

The overall results show that the proposed DSCP mangling detection methods can accurately identify DSCP modification events across different traffic patterns and detection approaches. The STUN-based method provides lightweight active probing and validates DSCP changes by comparing declared and observed DSCP values. The fingerprint-based method provides passive packet-level verification by comparing ingress and egress DSCP values using a stored packet fingerprint. The hybrid method combines both approaches and improves validation by using both payload-declared DSCP information and packet-level fingerprint confirmation.

Across the tested scenarios, the STUN-based and fingerprint-based methods achieved 100% detection accuracy with zero false positives or false negatives. The hybrid method also performed effectively, especially for UDP traffic, while TCP introduced additional timing and sequencing complexity. These results confirm that DSCP mangling can be detected reliably by comparing the original DSCP marking with the value observed after transmission through the network path.

References

Alarood, A. A., Ibrahim, A. A., and Alsubaei, F. S. (2023). Attacks notification of differentiated services code point (dscp) values modifications. *IEEE Access*, 11:126950–126966.

Babiarz, J., Chan, K. H., and Baker, F. (2006). Configuration guidelines for diffserv service classes. RFC 4594, RFC Editor.

Baker, F., Polk, J., and Dolly, M. (2010). A differentiated services code point (dscp) for capacity-admitted traffic. RFC 5865, RFC Editor.

Black, D. and Jones, P. (2015). Differentiated services (diffserv) and real-time communication. RFC 7657, RFC Editor.

Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., and Weiss, W. (1998). An architecture for differentiated services. RFC 2475, RFC Editor.

Chan, K. H., Babiarz, J., and Baker, F. (2008). Aggregation of diffserv service classes. RFC 5127, RFC Editor.

Cisco (2019). Quality of service (qos) configuration guide, cisco ios xe gibraltar 16.12.x (catalyst 3850 switches): Configuring qos.

Custura, A., Fairhurst, G., and Secchi, R. (2023). Considerations for assigning a new recommended differentiated services code point (dscp). RFC 9435, RFC Editor.

Custura, A., Secchi, R., and Fairhurst, G. (2018). Exploring dscp modification pathologies in the internet. *Computer Communications*, 127:86–94.

Davie, B., Charny, A., Bennett, J. C. R., Benson, K., Le Boudec, J.-Y., Courtney, W., Davari, S., Firoiu, V., and Stiliadis, D. (2002). An expedited forwarding phb (per-hop behavior). RFC 3246, RFC Editor.

Fortinet (2023). What is qos (quality of service)?

Galea, N. (2015). What is the stun protocol, its purpose and how it works.

Heinanen, J., Baker, F., Weiss, W., and Wroclawski, J. (1999). Assured forwarding phb group. RFC 2597, RFC Editor.

Jabbar, A. K., Karimi, B., Jamel, T. M., and Abood, A. (2019). Qos mapping method based on dscp/ip in lte and edca ac/mac in wifi network. In *2019 12th International Conference on Developments in eSystems Engineering (DeSE)*, pages 662–667.

Jones, P., Dhesikan, S., Jennings, C., and Druta, D. (2021). Differentiated services code point (dscp) packet markings for webrtc qos. RFC 8837, RFC Editor.

Keranen, A., Holmberg, C., and Rosenberg, J. (2018). Interactive connectivity establishment (ice): A protocol for network address translator (nat) traversal. RFC 8445, RFC Editor.

Khan, F. A. and Ibrahim, A. A. (2024). Network traffic classification analysis on differentiated services code point using deep learning models for efficient deep packet inspection. *International Journal of Innovative Computing*, 14(2):15–24.

Liu, C. J. (2007). Tracking dscp marking of packets in a qos enabled triple-play ip network. In *International Conference on Networking and Services (ICNS 2007)*.

Luo, H. and Shyu, M.-L. (2007). Differentiated service protection of multimedia transmission via detection of traffic anomalies. In *2007 IEEE International Conference on Multimedia and Expo*, pages 1539–1542.

Mahy, R., Matthews, P., and Rosenberg, J. (2010). Traversal using relays around nat (turn): Relay extensions to session traversal utilities for nat (stun). RFC 5766, RFC Editor.

Malikovich, K. M., Rajaboevich, G. S., Sobirovna, T. S., and Temurmalik, E. (2021). Differentiated services code point (dscp) traffic filtering method to prevent attacks. In *2021 International Conference on Information Science and Communications Technologies (ICISCT)*.

Martinsen, P.-E. and Nandakumar, S. (2016). Dscp mangle detection. Internet-Draft draft-martinsen-tram-dscp-mangle-detection-00, IETF. Expired Internet-Draft.

Mazhar, T., Malik, M. A., Mohsan, S. A. H., Li, Y., Haq, I., Ghorashi, S., Karim, F. K., and Mostafa, S. M. (2023). Quality of service (qos) performance analysis in a traffic engineering model for next-generation wireless sensor networks. *Symmetry*, 15(2):513.

Nichols, K., Blake, S., Baker, F., and Black, D. (1998). Definition of the differentiated services field (ds field) in the ipv4 and ipv6 headers. RFC 2474, RFC Editor.

Petit-Huguenin, M., Salgueiro, G., Rosenberg, J., Wing, D., Mahy, R., and Matthews, P. (2020). Session traversal utilities for nat (stun). RFC 8489, RFC Editor.