INSTITUT FÜR INFORMATIK
DER LUDWIG–MAXIMILIANS–UNIVERSITÄT MÜNCHEN

Masterarbeit

Evaluation of medium-sized networks based on an original implementation of the MEADcast Router

Adrian Schmidt

Draft vom February 15, 2024

INSTITUT FÜR INFORMATIK
DER LUDWIG–MAXIMILIANS–UNIVERSITÄT MÜNCHEN

Masterarbeit

Evaluation of medium-sized networks based on an original implementation of the MEADcast Router

Adrian Schmidt

Aufgabensteller:	PD Dr. rer. nat. Vitalian Danciu
Betreuer:	Daniel Diefenthaler Fabian Dreer Cuong Tran
Abgabetermin:	1. Februar 2024

Hiermit versichere ich, dass ich die vorliegende Masterarbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

München, den 7. Juli 2077

. . . . . .
(Unterschrift des Kandidaten)

Abstract

Hier steht eine kurze Zusammenfassung der Arbeit. Sie darf auf gar keinen Fall länger als eine Seite sein, ca. eine drittel bis eine halbe Seite ist optimal.

Contents

1 Introduction

1.1 Motivation

Over the last two decades, the number of people with access to the internet has continuously increased at a rate of 5%, resulting in a current total of over 5 billion individual internet users [ITU23]. Furthermore, the unrelenting demand for data has led to an average growth in total bandwidth usage of 30% over the past five years. This development was further accelerated by the COVID-19 pandemic, as the increased reliance on remote work, online education, and digital entertainment surged the bandwidth consumption [Car21]. These growth rates are expected to remain high in the future. Especially Multimedia content like Video Streams, Conferences, and Online Games depict a major portion of the global internet traffic. Video traffic is accountable for more than half of the bandwidth consumed in 2020 [Car21]. Network Operators and Service Providers need to comply with the continuously rising demand.

Already in the late 1980s, Deering and Cheriton proposed a multicast extension to the IP protocol, to facilitate efficient multipoint communication (*1:m, m:n*) [DC90; Dee89]. Multicast offers the advantage, to greatly reduce the occupied bandwidth by condensing identical traffic into a single stream, targeted towards multiple recipients [Cai+02]. Routers may replicate packets of that stream at points where the paths towards the receivers diverge. Many of today's internet services, particularly those with high bandwidth demands, may benefit significantly from multicast delivery [RES06; TD18]. Besides technical benefits, the adoption of multicast communication could reduce the emissions caused by Communication Technology (CT). Several studies assert that networks are accountable for a major portion of today's CT emissions [AE15]. Furthermore, they forecast strongly increasing energy consumption by networks until 2030. Multicast communication has the potential to lower energy consumption and therefore emissions by promoting more efficient use of network resources and potentially reducing the need for additional infrastructure.

1.2 Challenges of Multicast

The application of IP-Multicast has yielded mixed results. Practically all of today's devices support IP-Multicast [RES06]. Furthermore, it is utilized on a local scale, and various protocols like IPv6 Neighbor Discovery [Sim+07], RIPv2 [Mal98], OSPF [Moy98] and mDNS [CK13] rely on multicast.

However, more than thirty years since the initial proposal of IP-Multicast, its global deployment and usage still lags far behind expectations [Dio+00; RES06]. Despite the potential advantages of Multicast, the majority of internet traffic continues to rely on point-to-point communication, known as *unicast* [Zha+06]. There are several reasons for the limited usage of IP-Multicast.

Feasibility Besides the aforementioned advantages, IP-Multicast entails various technical obstacles. Compared to unicast, it exhibits increased technical complexity, requiring the interaction of various protocols on multiple network layers [RES06; Dio+00]. Furthermore, multipoint communication in general interferes with the application of today's widespread security mechanisms like encryption [RH03]. Moreover, IP-Multicast is based on a fixed address space [Dee89; DH06], which has an insufficient number of addresses. This limitation makes it infeasible to map highly dynamic sessions, like conferences, onto this space [DT19]. Additionally, the protocol involves a complex routing procedure, which requires all routers along the path to maintain a per-session state [Dio+00; RES06]. Consequently, global IP-Multicast availability is constrained, as successful packet delivery necessitates all routers to support the protocol. However, this scenario is unlikely since many commercial routers come preconfigured with IP-Multicast disabled [Aru19]. Moreover, the following paragraph describes, why Network Operators are probably not willing to enable it.

Desirability So far, Network Operators and Internet Service Providers (ISPs) have shown limited efforts to deploy Multicast within their Administrative Domains [Dio+00; RES06; ST02]. The increased technical complexity of Multicast requires more extensive management efforts. Additionally, due to the complex routing procedure, infrastructure upgrades may be necessary. Despite its potential for significant bandwidth savings, ISPs seem to assess the deployment of IP-Multicast as an unsuitable investment [RES06]. Another reason is the more complex pricing model of multipoint communication. Charging for multicast services is non-trivial compared to existing unicast billing [RES06]. On top of that, IP-Multicast hampers Network Operators' ability to anticipate network load. Forecasting the number of replicas generated from a multicast packet, entering the Network Operators Administrative Domain, is unfeasible [Dio+00]. Combined with the limited security mechanisms of IP-Multicast, this represents a vulnerability, potentially exploited for amplification attacks. This fact makes intra-domain IP-Multicast even more unlikely.

Unless ISPs face pressure to expand their service offering, increasing multicast deployment is doubtful. As multicast delivery has no direct impact on receivers, customers are not expected to exert the necessary pressure. Moreover, statistics illustrate that more than half of the internet traffic volume is HTTP-based [Clo23], which is not well suited for multipoint communication. Additionally the usage of Multicast is discouraged by web browsers, the most widely utilized HTTP clients, due to their technical limitation to TCP/HTTP[1].

1.3 Goal and Contribution

The current state of the internet put forth various unicast-based alternatives, aimed at addressing the absence of a globally usable multicast protocol [Zha+06]. One such alternative, known as MEADcast [TD18], offers the capability for 1:n sender-based IPv6 multipoint communication over the internet [DT19]. Key features of MEADcast include the preservation of receiver privacy, technology-agnostic destinations, and zero network support requirements.

Building upon prior research conducted by Danciu and Tran [DT19], the primary goal of this thesis is to conduct a real-world evaluation of MEADcast, utilizing a Linux Kernel

[1] Despite the growing popularity and browser support for QUIC [IET21; JC20], also known as HTTP over UDP, it encounters similar challenges as TCP/HTTPS. This include issues such as packet acknowledgment and the client side generation of random numbers [IT21].

implementation of the required router software. This step represents a logical progression from earlier investigations of MEADcast, which were based on network simulations [TD18] and Software-defined networking (SDN) [Ngu19]. The evaluation primarily focuses on the following aspects:

Feasibility Study The first part of the evaluation assesses the feasibility of deploying MEADcast in a network. This aims to identify potential limitations and structural issues of the current protocol specification. Further investigation examines the practicality of using MEADcast on the internet, taking into consideration concerns related to IPv6 extension header processing [Gon+16].

Performance Evaluation A comparative performance analysis is conducted to evaluate MEADcast in comparison to both IP unicast and multicast. This assessment provides insights into the efficiency and effectiveness of MEADcast as a multipoint communication solution.

Scenario identification Building on the results of the previous evaluation steps, this phase involves identifying scenarios, application categories, and characteristics that justify the utilization of MEADcast. Thereby, it can be determined where MEADcast may offer advantages and excel in real-world applications.

Aligned with the established objectives, this thesis presents a series of contributions. Firstly, it introduces a Linux Kernel implementation of the MEADcast router, facilitating the deployment of MEADcast in a real network. Second, it offers a traffic generator designed to serve as the MEADcast sender, simulating the application of the protocol. These contributions are further solidified through the deployment of MEADcast in both a controlled testbed environment and a real-world network. Lastly, this research evaluates MEADcast's feasibility, performance, and potential application scenarios based on a series of conducted experiments. This evaluation provides valuable insights for its future implementation and development, ultimately enabling us to propose a revision of the protocol specification.

The findings indicate, that with certain limitations, MEADcast is applicable in real networks. The current protocol specification suffers from poor availability on the internet due to the limited processing of IPv6 packets with extension headers. However, the proposed modification effectively addresses this obstacle while enhancing receiver privacy. Our measurements suggest that MEADcast's performance falls between uni- and multicast, particularly showing promise in scenarios characterized by limited bandwidth and network control.

1.4 Method

To ensure the achievement of the previously defined goals, this thesis follows the procedure illustrated in Figure 1.1. First, we conduct a literature review of several multicast protocols and analyze the current challenges of global multipoint communication. With a clear understanding of the relevant protocols and their limitations we define the objectives of this thesis. Furthermore, we select adequate evaluation metrics and criteria to asses MEADcast's feasibility, performance, and potential application scenarios. Subsequently, we design a series of experiments aimed at capturing these metrics. Next, we outline the testbed requirements

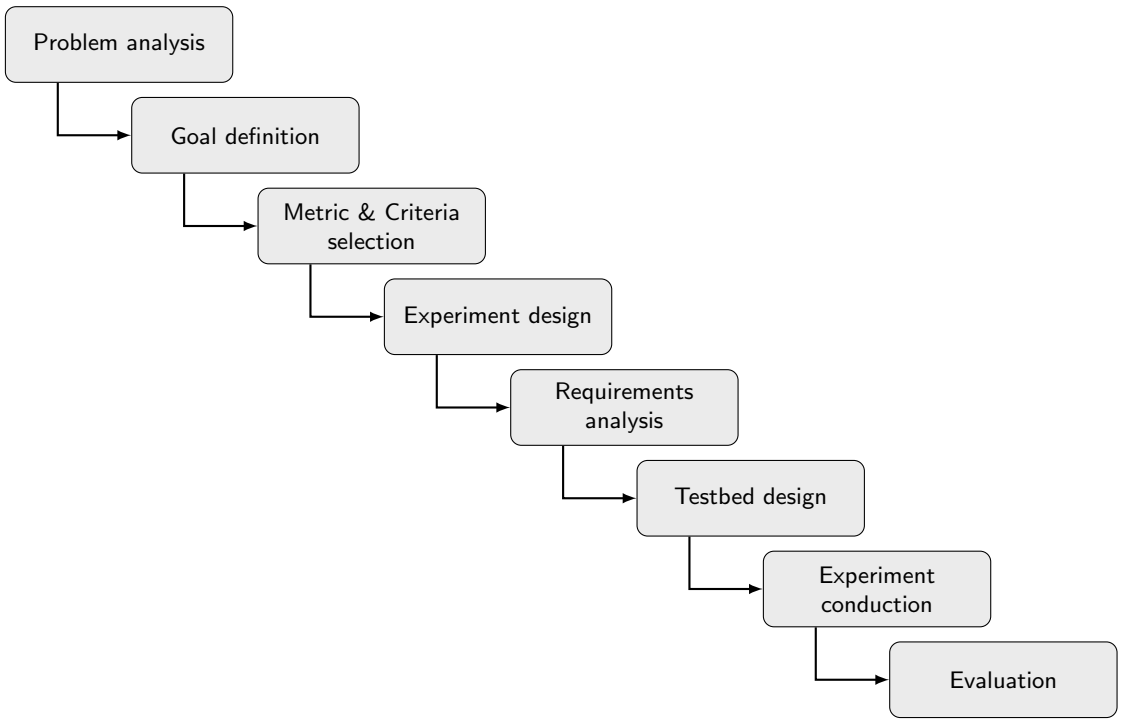

Figure 1.1: Research method

derived from these experiments and elaborate on a corresponding testbed design. MEADcast is then deployed in the testbed, and the experiments are conducted. Finally, we present the obtained results. These findings are critically evaluated with respect to the thesis' overarching goal, providing valuable insights into the feasibility and performance of MEADcast. This analysis allows us to identify scenarios and applications for which the protocol is well-suited.

1.5 Structure

The method employed in this thesis ensures a cohesive narrative, with each chapter building upon the knowledge acquired in the previous sections. Chapter 1 depicts the motivation for investigating multipoint communication, highlights the current challenges of IP-Multicast, and articulates the goal of this thesis. Moving forward, Chapter 2 establishes the theoretical foundation for subsequent investigations, by examining several multicast protocols, including IP-Multicast, Xcast, and MEADcast. Additionally, the chapter provides a brief overview of Kernel development fundamentals and the network stack, laying the groundwork for the router implementation. Chapter 3 presents our selection of evaluation metrics and criteria, outlines the testbed requirements and presents the corresponding testbed design. Chapter 4 delves into technical details about the Kernel implementation of the router software and provides detailed specifications of the testbed. This chapter serves as the foundation for the practical experiments. In Chapter 5 we present the results from our experiments and evaluate them. Finally, Chapter 6 summarizes the findings and draws conclusions from this research. Additionally, it outlines potential avenues for future work and exploration in this field.

2 Background and Related Work

2.1 Multicast Protocols

Multicast communication refers to the simultaneous delivery of data towards an arbitrary number of destinations [KPP93; LN93]. Numerous network protocols have been developed to facilitate multicast communication, encompassing both Network Layer as well as Application Layer implementations [Zha+06; ST02]. This chapter first distinguishes Multicast from other communication schemes. Subsequently, various multicast protocols are introduced.

According to the OSI model [Zim80] Layer 3, known as the Network Layer, is responsible for end-to-end delivery of data between nodes. In computer networks data is transferred through Protocol Data Units (PDUs), composed of protocol specific control information (e.g. source and destination address), along with a payload carrying the actual data. To establish a path from the sender to the destination(s), packets (Layer 3 PDUs) may traverse multiple intermediate nodes [Pos81]. This process is called routing and can be classified into various schemes. For the purpose of this thesis, our primary focus lies on the schemes depicted in Figure 2.1. *Unicast* denotes a one-to-one association between a sender and a single destination. *Broadcast* disseminates packets to all nodes within the sender's broadcast domain (Layer 2) [Mog84]. Typically, IP Routers (Layer 3) serve as the boundary of a broadcast domain. *Multicast* transmits packets to a group of destinations, accommodating both one-to-many and many-to-many communication [Dee89]. In contrast to Broadcast, Multicast does not necessarily deliver packets to all available nodes. Furthermore, Multicast packets can be delivered beyond the sender's broadcast domain, implying subsequent replication of the packets.

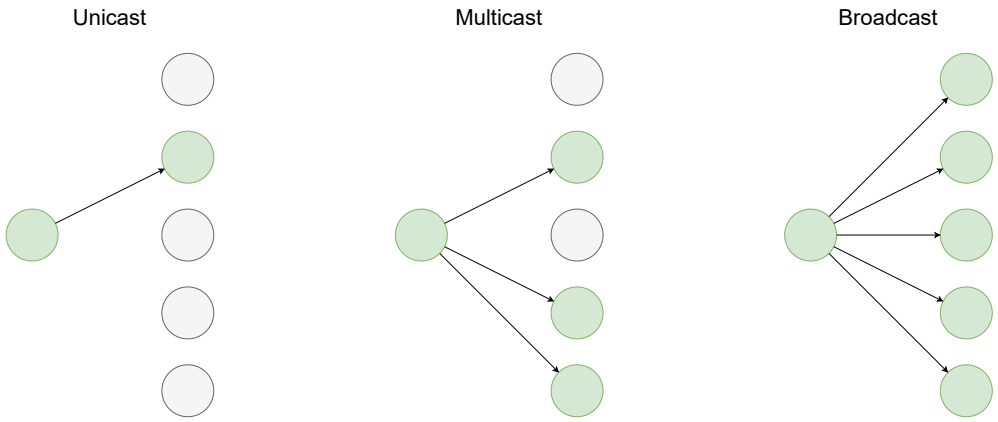

Figure 2.1: Network Layer: Routing schemes

2.1.1 IP Multicast

Already in the late 1980s, Deering and Cheriton [DC90] proposed an extension to the IP protocol, to facilitate efficient multipoint communication (*1:n*, *m:n*). This extension is known as IP-Multicast and was first standardized in RFC 988 (1986) [Dee86]. IP-Multicast offers the advantage, to greatly reduce the bandwidth by condensing identical traffic into a single stream targeted to a so called "Host Group" [Dee89]. IP-Multicast operates on a separate address space and host groups are identified based on an IP address within the designated address range. For instance, IPv4 host groups are identified by class D IP addresses starting with "`1110`" as their most significant bits (MSBs) [Dee89]. This characteristic facilitates high scalability, such that host groups may encompass thousands of receivers, since senders are not required of any knowledge about the number of receivers nor receivers identity or location. Intermediate nodes like IP routers replicate multicast packets addressed to an host group where the paths towards the receivers diverge.

- Separate address space
- IGMP / MLD
- Intra-Domain Routing: DVMRP, MOSPF, PIM dense/spare
- Inter-Domain Routing: MBGP, MSDP

2.1.2 Xcast

Traditional multicast scales well with large multicast groups but have issues with a high number of distinct groups. Xcast is a mutlicast protocol with complementary scaling properties compared to the traditional approach [Boi+07]. The protocol is designed with the key idea of supporting huge numbers of small multicast sessions. Xcast achieves this by explicitly encoding the receiver addresses in each packet, instead of using a multicast addresses [Boi+07].

2.1.3 MEADcast

2.2 Linux Kernel

2.2.1 Fundamentals

2.2.2 Network Stack

3 Experiment Design

This chapter addresses the design of a series of experiments to evaluate MEADcast. To ensure the quality of the experiments Section 3.1 discusses research questions driving our design process. Subsequently, Section 3.2 elaborates on motivations for employing multicast, potential application domains and their characteristics. This chapter deals with the selection of measurements, design of experiments and the design of a testbed based on requirements derived from the experiments.

Based on thesis goal we first formulate overarching research questions we endeavour to answer with our experiments. Therefore, these questions guide our design process of the experiments.

3.1 Measurements

Aligned with the goal of this thesis, we begin by formulating overarching research questions, guiding the design of the experiments. Additionally, this section outlines our approach to answering these questions.

RQ1 How robust is the current MEADcast specification?

While prior research on MEADcast has primarily taken place in simulated environments [TD18; DT19] and small stub networks [Ngu19], this thesis endeavors to assess MEADcast in a more realistic setting. To achieve this, we perform a stress test to evaluate the protocol's behavior in a less "clinical" environment. During this test we observe MEADcast's reaction to deliberate routing changes, to evaluate its adaptivity and suitability for dynamic network environments. Further, we simulate network disruptions by inducing router outages, to assess MEADcast's resilience and recovery capabilities. The stress test also examines MEADcast anomaly handling by injecting modified discovery responses. Additionally, a firewall is employed to intentionally drop MEADcast packets during the data delivery phase, enabling us to evaluate the efficacy of the fallback mechanism. The results of the stress test shed light on the robustness of the current MEADcast specification, especially in diverse network conditions. This investigation aims to provide valuable insights into the real-world applicability and resilience of MEADcast.

RQ2 How does MEADcast perform compared to IP-Unicast and IP-Multicast?

In addition to assessing the robustness of a protocol, its performance, especially in comparison to existing alternatives, is a pivotal factor influencing its adoption. To address this research question, comparative performance measurements for MEADcast, IP-Unicast, and IP-Multicast are conducted across a series of experiments elaborated in Section 3.2. These measurements encompass metrics such as throughput, latency, jitter, and resource utilization. Special attention is given to the impact of MEADcast's discovery phase on its performance. This encompasses assessing the overhead

produced by the discovery mechanism. Furthermore, we evaluate how the protocol's shift from extensive unicast to MEADcast delivery affects the performance metrics. The outcomes contribute not only to evaluating MEADcast performance but also to drawing conclusions for subsequent research questions.

RQ3 Which applications and characteristics are well served by MEADcast?

Addressing this research question involves designing a series of experiments depicting a diverse range of applications with distinct characteristics (see Section 3.2). The selection of scenarios is guided by an initial exploration of motivations for employing multicast communication, and more specifically, MEADcast. These motivations lay the groundwork for identifying a range of potential application domains where MEADcast deployment could prove beneficial. Subsequently, we formulate application characteristics distinguishing these domains. Based on the gathered insights, various scenarios are chosen to represent different application domains and their unique set of characteristics. This comprehensive selection ensures a throughout examination of MEADcast's suitability across a diverse spectrum of use cases. The results aim to delineate MEADcast's application space by identifying its strengths and limitations.

RQ4 In which conditions is the usage of MEADcast sensible?

The viability of employing MEADcast is influenced not only by application domains and their characteristics but also by prevailing circumstances. To investigate this aspect, the experiments are executed with various parameter configurations. For instance, the level of network control and the number of available MEADcast routers affect the performance and thus the viability of MEADcast. This applies equally to testing parameters like available bandwidth, number of endpoints, and their distribution. The goal is to provide insights into the conditions under which deploying MEADcast is advantageous compared to existing alternatives. The results aim to offer guidance for making informed decisions regarding the adoption of MEADcast in specific circumstances.

3.2 Scenarios

To design comparable experiments we have to elaborate on application domains that may benefit from Multicast-Communication. Following, we formulate application characteristics distinguishing these domains. This knowledge enables us to design a series of experiments representing a variety of application domains with differing characteristics. Thereby we aim to answer RQX.

3.2.1 Motivations for employing Multicast

To select appropriate measurements and experiments, it is essential to delve into the *motivations* for employing multicast communication and, specifically, MEADcast as a protocol.

In a broader context, the decision to utilize multicast often resolves around optimizing resource usage. In the absence of resource constraints, one might question opting for multicast rather than ubiquitous unicast communication. However, several compelling reasons advocate for the adoption of a multicast protocol. A predominant motive is overcoming *resource limitations*, which can manifest technically, such as a network connection incapable

of providing the required bandwidth, or Internet of Things (IoT) devices like IP cameras struggling to handle numerous unicast connections requesting identical content.

Financial considerations are also a driving reason for employing multicast, since most internet services operate on a subscription or pay-as-you-go pricing model (pay per usage). ISPs, mobile phone operators, and VPN providers structure their services based on factors like monthly available traffic volume, upstream and downstream bandwidth. Subscribed service levels can thus become the financial cause of resource limitations. Cloud providers, in particular, heavily rely on pay-as-you-go pricing, often contending their customers with expensive "egress costs". Deploying multicast emerges as a strategic approach to not only overcome technical and financial constraints but also to reduce operational costs.

Another motivation for multicast adoption is to facilitate new services or enhance existing ones. For example, employing multicast for delivering multimedia content enables service providers to offer higher-quality audio or video without the need to either increase operational costs or overcome prevailing resource limitations. Moreover, multicast can empower individuals to access bandwidth-intensive services despite their limited bandwidth. For instance, peer-to-peer (P2P) conferencing services could leverage multicast to bridge the asymmetric access link common in most households (high downstream and low upstream bandwidth) [Boi+07; Car21]. Additionally, network operators may strategically deploy multicast within their domain to decrease overall network load, concurrently reducing operational costs and mitigating emissions caused by CT (see Section 1.1).

As constituted in Section 1.2 and Subsection 2.1.1, in many scenarios, the usage of IP-Multicast is not a viable option. This creates a potential application space for MEADcast, particularly in circumstances with limited network control and an uncertain degree of IP-Multicast support. Another potential domain for MEADcast is in facilitating access control-based multicast communication, addressing the absence of a receiver authorization mechanism in IP-Multicast [Dio+00]. In comparison to alternatives like Xcast or Application-Layer Multicast (ALM), MEADcast might be favored in situations where the disclosure of receiver information, such as IP addresses, among other group members, is deemed unacceptable.

3.2.2 Application Domains

As highlighted in the preceding section, numerous motivations support the adoption of multicast communication. Additionally, various types of applications have the potential to benefit from multicast communication. However, it is essential to acknowledge that multicast is tailored for a specific form of communication – *simultaneous* transmission of *identical* data to multiple recipients, as delineated in Table 3.1. This inherent characteristic finds an ideal match in TV broadcasts, establishing them as a prime candidate for multicast communication. Conversely, Video on Demand services such as YouTube present a less favorable scenario for multicast adoption, as users request videos at different points in time. The advantages of multicast become particularly evident in bandwidth-intense applications, where its potential to significantly reduce the total communication volume is most pronounced. Real-time multimedia services, therefore, emerge as a particularly well-suited domain for the application of multicast. In the following, we introduce the formulated application domains, illustrating them with specific examples as detailed in Table 3.2. The characteristics distinguishing these domains are discussed in the following section.

The first application domain is *Multimedia Streaming*, encompassing various applications transmitting multimedia content to an arbitrary number of destinations, such as IPTV

	Temporal	
Content	Synchronous	Asynchronous
Identical	TV broadcast (yes)	Video on demand (no)
Different	- (no)	Web browsing (no)

Table 3.1: Suitability of communication patterns for Multicast

[DT19; RES06], Internet Radio [TD18], podcasts, and live streaming (e.g. Twitch). *Conferencing and Collaboration* is the second domain, comprising Audio and Video Conferences [ST02; DT19; KPP93], Voice over IP (VoIP) [BGC04; Boi+07], as well as real-time collaboration applications [Dio+00; Boi+07] like online Mind Maps and Whiteboards. The next domain *File Transfer*, covers numerous applications distributing files to multiple recipients, including Software Distribution and Updates, Patch Management [TD18; RES06], Logging [Dio+00] as well as file sharing and synchronization [ST02]. *Information delivery* comprises applications pushing information to multiple destinations. One example is a news application, which notifies it's users about new information of topics or channels they have subscribed [Dio+00]. Other applications include widely utilized smartphone notifications, RSS feeds [RES06], Logging (e.g. SNMP), and Stock quotes [Cis01]. The last domain is *Distributed Simulation*, with exemplary applications such physics simulations [Dio+00], virtual reality, and online gaming [RES06].

Domain	**Applications**
Multimedia streaming	IPTV, Internet Radio, podcasts, streaming platforms
Conferencing & Collaboration	Audio- & Video-Conferences, VoIP, Mindmaps, Whiteboards, ...
File Transfer	Software Distribution, Updates, Patch Management, File-sharing and -synchronization, Logging
Push notifications	RSS-feed, Logging, Stock quotes
Distributed simulation	Online Gaming, Virtual World, Simulation

Table 3.2: Multicast application domains

3.2.3 Application Characteristics

This section delineates various characteristics derived from the application domains, categorized into three groups, as detailed in Table 3.3.

Group Group communication occurs in diverse forms. The number of participants in a multicast group displays substantial variation, ranging from small (< 10), to medium-sized and up to large-sized (> 100) groups. Additionally, the session duration exhibits significant

Domain	Group/Session			Communication		Network		
	Duration[1]	Mem. ship	Size[2]	Pattern	Interval	Pkt. size	Through. (tx/rx)	Latency /Jitter
Multimedia	h-d	dyn.	l	1:n	steady	med.	high/med.	med.
Conference	m-h	stat.	s-m	m:n	steady	med.	high/high	low
File transfer	s-h	stat.	s-l	1:n	burst	large	high/high	high
Push info.	s	stat.	m-l	1:n	burst	small	med./low	med.
Dist. sim.	m-h	stat.	s-m	m:n	steady	small	low/med.	low

[1] *(s)econds, (h)ours, (d)ays*
[2] *(s)mall, (m)edium, (l)arge*

Table 3.3: Characteristics of the application domains

diversity, spanning from a few minutes up to several days. The group *membership* can either be static or dynamic. For instance, small to medium-sized conferences might last several minutes to a few hours with mostly static group membership. In contrast, broadcasts of music or sports events endure for several days, attracting millions of viewers who may join or leave at any time.

Communication The communication *pattern* within a group can either be symmetric (m:n) or asymmetric (1:n). Furthermore, the *interval* during which participants exchange messages may constitute a steady flow (e.g. video stream) or a recurring burst (e.g. file transfer). In a P2P video conference or online game, all group members continuously transmit data to each other. Conversely, in an RSS feed, a single server periodically pushes information to multiple subscribers.

Network The network requirements of different applications exhibit significant variation. Specific applications exhibit sensitivity to distinct factors, with some prioritizing low *latency* and *jitter*, others demanding high *throughput*, and yet others being sensitive to the packet *drop rate*. For instance, many online games rely on low latency while maintaining frugal bandwidth requirements [Har+07]. In contrast, ensuring a high-quality video stream necessitates elevated throughput, even though increased latency is acceptable due to prevalent client-side buffering. Furthermore, an RSS feed has low throughput and moderate latency requirements. Nevertheless, it is sensitive to the packet drop rate, since it has to ensure successful data delivery.

3.2.4 Application Scenarios

Based on the previously exhibited application domains and characteristics this chapter formulates several application scenarios.

The primary goal is, to develop a series of experiments encompassing various domains and characteristics. This comprehensive selection ensures a throughout examination of MEADcast's suitability across a diverse spectrum of use cases. Therefore, the results from the experiments are pivotal to answer *RQ3*. The characteristics of each experiment are illustrated in Table 3.4.

EX1: Live Stream The first experiment is a video live stream, representing the multimedia stream domain. As stated by Cartesian [Car21], multimedia traffic is accountable for more than half of the bandwidth consumed in 2020. Therefore, this experiment is representative for a major application space. Conceivable scenarios are, that a major streaming provider wants to lower their egress costs, or an aspiring platform has to overcome preserving resource limitations to satisfy rapidly growing bandwidth demands. Since, multimedia streams produce a high traffic volume on the sender it is of particular interest, how much MEADcast can reduce the occupied bandwidth. Moreover, it is examined if this comes with trade-offs like, increased latency/jitter or an increased CPU utilization of the sender.

EX2: File tranfer The second experiment imitates a file distribution, thus representing the file transfer domain. Especially software updates and file backups share major characteristics with this scenario. The primary focus of this experiment is to investigate, how well MEADcast copes with recurring burst of a high traffic volume. The characteristics of this experiment are predestined to measure the effects of the discover phase on metrics such as bandwidth and latency. By adjusting parameters like communication interval and group size, this experiment can be conducted analogous to *EX1*.

EX3 Video conference The third experiment simulates a P2P video conference. This experiment is designed to investigate, whether MEADcast is capable of bridging asymmetric access link bandwidth [Boi+07; Car21]. Furthermore, the characteristics of this scenario facilitate an environment in which the effects of an high MEADcast traffic volume on the MEADcast routers can be examined. Since, video conferences usually have a significant lower number of group members it is analyzed, how MEADcast copes with a small number of recipients. Since, this experiment is conducted for smaller group sizes the overhead of MEADcast is also analysed.

EX4 Online game The last experiment is an online multiplayer game, representing the application domain distributed simulation. This experiment has two primary goals. First since, online games are highly sensitive to Latency and Jitter we examine MEADcast's effect on these metrics. Secondly, the overhead of the protocol can be analyzed, hence online games have low bandwidth requirements. By adjusting parameters such as packet size and communication interval, this experiment can be conducted analogous to *EX3*.

	Communication		**Network**			**Group/Session**		
Scenario	Pattern	Interval	Pkt. size	Through. (tx/rx)	Latency /Jitter	Dura-tion[1]	Mem. ship	Size
Live Stream	1:n	steady	med.	high/med.	med.	h-d	dyn.	l
File transfer	1:n	burst	large	high/high	high	s-h	stat.	s-l
Conference	m:n	steady	med.	high/high	low	m-h	stat.	s-m
Online Game	m:n	steady	small	low/med.	low	m-h	stat.	s-m

[1] *(s)econds, (h)ours, (d)ays*

Table 3.4: Show diff scenarios meeting

3.2.5 Testing parameters

The series of experiments introduced earlier shares several testing parameters. Consequently, these experiments incorporate diverse parameter configurations intended to assess their influence on measurements, specifically addressing *RQ2* and *RQ4*.

The first parameter is the *number and distribution of receivers*. Generally, with a larger number of receivers, a multicast protocol tends to achieve proportionally higher bandwidth savings compared to unicast. However, we anticipate that the distribution of receivers significantly impacts MEADcast's performance. Therefore, the experiments encompass various distributions, ranging from highly clustered receivers with minimal internal distances to uniformly spread distributions. This parameter aims to illuminate how spatial receiver arrangement influences the protocol's efficiency.

The second testing parameter is the *degree of MEADcast support*. Analogous to the first parameter, we hypothesize that the number and placement of MEADcast-capable routers have a significant impact on the protocol's performance. To investigate this, the experiments are conducted ranging from a minimal degree of MEADcast routers to complete protocol support within the testbed. By varying this parameter, we aim to assess MEADcast's efficiency under conditions of limited network control, offering valuable insights directly addressing *RQ4*.

The third, parameter resolves around *grouping receivers* into MEADcast packets. Evaluating the balance between the number of recipients per packet and the available service data unit (SDU) size is crucial. A larger number of recipients per packet potentially reduces traffic volume by consolidating identical packet streams. However, this also enlarges the MEADcast header, consequently diminishing the available SDU size. This trade-off might inadvertently increase traffic volume since more packets are required to transmit the total payload. Furthermore, the grouping algorithm itself is considered. For instance, recipients sharing the same parent router could be distributed across different packets to achieve an optimal balance between the MEADcast header and payload size. Moreover, receivers with distinct parents might be merged under a common ancestor to accommodate them within a single packet.

Lastly, the effect of different values for the *discovery interval* is also examined. Shorter intervals facilitate quicker adaption to topology changes, potentially leading to enhanced receiver grouping and faster error recovery. However, shorter intervals also result in higher traffic volume, which may impede data delivery.

3.3 Testbed

3.3.1 Requirements

Based on the previously introduced experiments this section elaborates on requirements for the testbed. We structure the requirements into 3 categories:

Topology First, the topology must transcend basic structures like plain bus or simple ring setups, ensuring a *non-trivial connected graph*. A graph is termed connected when there exists a path between every pair of vertices, which in this context represents clients and routers. In order to thoroughly evaluate a multicast protocol, the topology needs to encompass a substantial number of nodes. Therefore, a *medium-sized topology* comprising

around 200 nodes [Aca14], featuring a proportional mix of clients and routers, is pivotal. In line with *RQ1*, the graph should reflect a *realistic network topology* of this size. The following section discusses characteristics of a realistic topology. Moreover, assessing the resilience of MEADcast in dynamic network environments necessitates the inclusion of *alternative paths* to simulate routing alterations and network disruptions. Additionally, a thorough evaluation of MEADcast's performance across diverse levels of network support and various grouping strategies, requires the topology to incorporate paths of *sufficient length*. This allows packets to traverse multiple MEADcast routers, ensuring a comprehensive assessment.

Another important requirement is the flexibility of the testbed. Diverse experiments and network conditions demand a modular and adaptable topology. Various placements of senders, receivers, and routers are necessary. Moreover, the testbed should encompass receiver distributions ranging from highly clustered to uniformly spread. Additionally, the ability to adjust the quantity of MEADcast-capable routers is crucial for evaluating various levels of MEADcast support.

Finally, several prerequisites pertain to the connectivity of our testbed. Given the evaluation of a Layer 3 IPv6-based protocol, the presence of multiple subnets and IP routing is mandatory. Beyond that, IP multicast routing is indispensable for a comparison of MEADcast and IP multicast. Moreover, to pursue *RQ1* the capability to deploy firewall mechanisms is required.

Sender To ensure a comprehensive assessment of MEADcast, the sender should include the following features:

First, the sender must periodically dispatch discovery requests to the receivers. Additionally, the sender needs to receive discovery responses and process them. After, the discovery timeout is exceeded the sender should start grouping the endpoints. Therefore, the sender is required to efficiently organize and group receivers into MEADcast packets, facilitating streamlined data transmission. Whenever possible, the software should support MEADcast data transmission. However, unicast is used as a fallback mechanism, if MEADcast is not available on certain paths. The discovery interval, discovery timeout, and grouping functionalities should be configurable, allowing flexibility for experimentation and optimization based on various network conditions and scenarios. Considering our evaluation of MEADcast is based on metrics such as latency and jitter, the sender should possess the capability to either measure these metrics directly or act as a proxy for traffic received from network measurement tools.

Router To enable the usage of MEADcast the router should encompass the following features:

The router should receive discovery requests, reply to them with a discovery response, and forward the request according to the destination address. Additionally, the router needs to receive, process, modify, replicate, and forward MEADcast data packets accordingly. For each receiver listed in the MEADcast header assigned to the router itself, the data should be transmitted the data to the respective recipient using IP unicast. In case of a MEADcast to unicast transformation Layer 4 checksum correction is may required. Lastly, it is preferable, if MEADcast can be turned on and off, to ease the deployment of various level of network support.

Non-functional Given that the performance evaluation of MEADcast encompasses time-sensitive metrics such as latency and jitter, it is imperative that the clocks of each node are synchronized. This synchronization is crucial for ensuring comparable measurements. Additionally, similar routing procedures must be employed for IP unicast, IP multicast, and MEADcast to maintain consistency in the evaluation. For instance, comparing software-based MEADcast routing with hardware-based IP routing would not yield meaningful results.

3.3.2 Topology

First, this section introduces state-of-the-art characteristics of medium to large-sized network architectures. Subsequently, we delve into the network topology of our testbed.

Several fundamental design principles must be met by any network, irrespective of its size [Aca14]. These principles are mutually interdependent and also tightly coupled to the overall design [Cis08].

Hierarchy: A hierarchical network model serves as a crucial high-level tool for crafting a reliable network topology [Aca14]. This abstraction dissects the complex challenge of network design into smaller and more manageable areas. A significant advantage of a hierarchical topology is that local traffic is confined to the local network [Cis08]. Only traffic destined for other networks is transferred to a higher layer.

Modularity: Dividing the network into components provides numerous benefits. First, each component or module can be designed with a level of independence from the overall structure [Aca14]. All modules can operate as semi-independent elements, contributing to higher overall system availability and facilitating simpler management and operations [Cis08]. Furthermore, network changes and upgrades can be applied in a controlled and staged manner, enhancing flexibility in maintaining and operating the network [Aca14]. This principle has also been leveraged in software architecture for many years, by separating the application into presentation, logic, and data layers [Ric15]. Another benefit is a higher degree of isolation, confining problems and failures to their component, leaving the overall network unaffected [Cis14]. In summary, dividing a network into a set of assembled building blocks facilitates increased stability, flexibility, and manageability for both the individual components and the network as a whole. [Cis08]

Resiliency: The network is expected to sustain availability for both regular and irregular scenarios. Typical scenarios encompass anticipated traffic flows, expected traffic patterns, and scheduled events like maintenance windows [Aca14]. Abnormal scenarios involve hardware or software failures, high traffic loads, anomalous traffic patterns, denial-of-service (DoS) events, and other unforeseen circumstances [Cis08].

Flexibility: As the operational lifespan of networks extends, it becomes imperative for their design to facilitate enhanced adaptability and flexibility [Cis08]. Adapting to evolving business environments and their associated communication requirements is a practical business and operational necessity. Flexibility is the capacity to modify specific sections of the network, introduce new services, or enhance capacity without necessitating a substantial overhaul or replacement of major hardware devices [Aca14].

To meet these ubiquitous network requirements, Haviland [Hav98] first proposed the *hierarchical three-layer internetworking model* in 1998. This model has evolved into an industry-wide standard for designing reliable, scalable, and cost-efficient networks [Aca14]. Leading networking and telecommunication providers like Cisco and Huawei recommend the implementation of the hierarchical three-layer internetworking model. [Cis08; Hua23]. The model

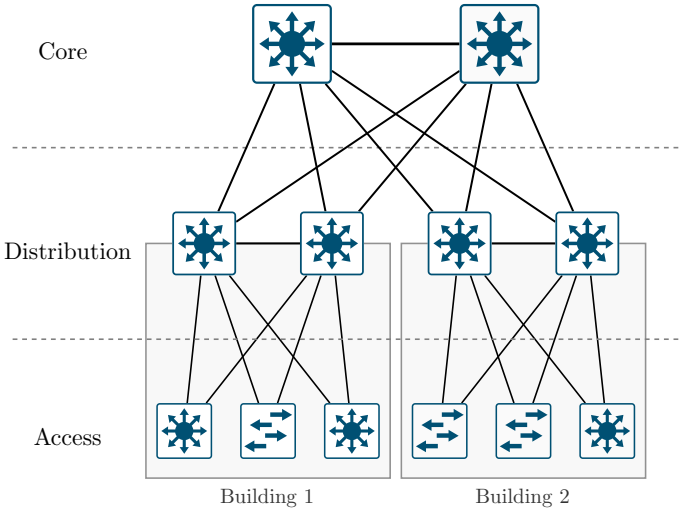

Figure 3.1: Hierarchical three-layer internetworking model (based on [Hav98])

divides networks into three layers: core, distribution, and access as illustrated in Figure 3.1 [Hav98].

Access Layer: The access layer facilitates network access for end devices, encompassing PCs, smartphones, printers, and IP cameras [Aca14]. Operating commonly at layer 2, it establishes connectivity between end devices. Given the diverse range of end devices, this layer incorporates an array of network devices such as switches, access points, and routers [Cis08]. Serving as the demarcation point between the network infrastructure and computing devices [Cis08], the access layer acts as the initial security boundary, safeguarding other users, application resources, and the network itself [Cis14].

Distribution Layer: The distribution layer aggregates traffic from multiple access layer devices and provides connectivity to the rest of the network by transmitting packets to other access layer devices or the core layer [Cis08]. In a typical scenario, connected routers at the distribution layer aggregate data originating from a building or a floor and establish connections to the core layer. For instance, in Figure 3.1, the two interconnected routers on the left and the two on the right could each symbolize a distinct building. This layer often serves as the demarcation between layer 2 domains and the layer 3 routed network [Aca14], creating fault domains containing failures and network changes to the directly affected areas [Cis14]. The distribution layer performs essential network functions such as routing, traffic filtering, and policy enforcement [Cis08].

Core Layer: The core layer functions as the backbone interconnecting the numerous modules of the network [Cis08]. It facilitates connectivity among end devices, servers, data storage, different locations, and the internet. This layer represents the most critical component of the network, characterized by a relatively simple design [Cis14]. Given that a significant portion of network traffic passes through the backbone, it is responsible for high-speed and high-volume data transmission [Cis08]. Additionally, this layer should offer high availability and redundancy. CPU-intensive packet manipulations resulting from security measures, inspections, and Quality of Service (QoS) should be avoided [Cis08].

The architecture of our testbed aligns with the previously design goals, achieved through

the implementation of the hierarchical three-layer model.

The testbed comprises 205 nodes consisting of 40 routers and 165 end devices. The network is divided into 4 domains, each potentially representing distinct buildings on a university campus. All 4 domains encompass 7-9 client networks, accommodating 5 end devices each. The access layer of each domain comprises up to 7 routers (3-digit router IDs in Figure 3.2). Since this thesis evaluates a transport protocol, we waive the application of switches on the access layer, deploying solely layer 3 devices. The distribution layer of each domain encompasses 2-3 interconnected routers (2-digit router IDs), aggregating the access layers within its respective domain and facilitating the connection to the core layer. The testbed's core consists of 4 routers (1-digit router IDs) interconnecting the 4 network domains. Additionally, the backbone possesses a high connectivity, ensuring both high availability and redundancy. To reduce management and operational overhead, the testbed incorporates fewer redundant links from the access to the distribution layer and from the distribution to the core layer compared to the recommendations provided by Cisco and Huawei [Cis14; Hua23]. This trade-off is considered acceptable, as this thesis does not delve into the details of the availability and fault tolerance of the network architecture. Additionally, router and link failures are artificially induced in any case.

The chosen architecture represents a realistic medium-sized network topology. The application of the hierarchical three-layer model ensures the required flexibility of the testbed. Various placements of senders, receivers, and routers are possible, allowing experiments with a receiver distribution ranging from high clustering within a single domain to a uniform spread across all four network domains. Moreover, the availability of alternative routes facilitates the representation of dynamic network environments, encompassing scenarios such as deliberate routing changes or router and link outages.

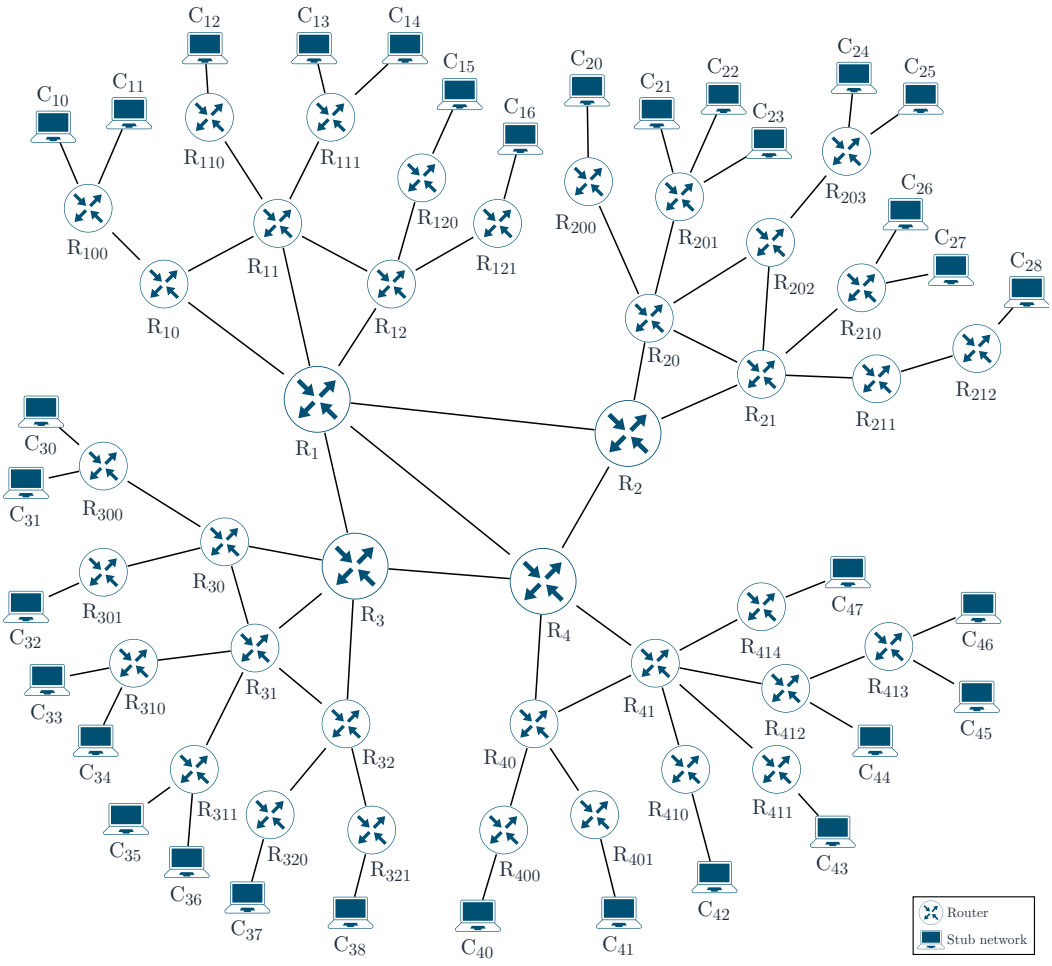

Figure 3.2: Testbed Topology

4 Implementation

This chapter introduces the employed protocol specification, the implementation of the MEADcast router and sender, as well as the testbed.

4.1 Protocol Specification

Since the existing literature on MEADcast lacks specific details regarding the header, such as the size of each field, we introduce the specifications utilized for our experiments.

Firstly, we omit the usage of an empty Hop-by-Hop IPv6 extension header as introduced by Tran and Danciu [TD18; DT19]. We do this for several reasons: First, to reduce the overhead of MEADcast. Second, if a non-MEADcast router attempts to process Hop-by-Hop extension headers the packet probably end up in the router's slow path [CJ13], potentially harming the protocol's performance. Third, since the MEADcast header is intended to be processed exclusively by MEADcast routers, over which we maintain control, the Hop-by-Hop extension header serves no purpose. Lastly, according to various research studies, packets containing a Hop-by-Hop extension header experience an increased drop rate of up to 40% compared to packets without an extension header [Gon+16], especially across multiple Autonomous systems (ASs) [Gon+21; GL22].

Next, we introduce the detailed characteristics of the MEADcast header in our implementation. The first four octets of the MEADcast header constitute the static Routing extension header, which is shared by all routing variants. This differs from previous implementations of MEADcast, which omitted the Segments Left field [Ngu19]. We decided to include this field to comply with RFC 8200 [DH17], reducing the likelihood that intermediate nodes drop MEADcast packets due to a malformed Routing header. Since there is no existing Routing Type designated for MEADcast, we utilize the experimental values of 253 and 254 in accordance with RFC3692 [Nar04]. The Segments Left field remains fixed to zero and is never altered because, according to RFC 8200 [DH17], intermediate nodes that do not recognize the employed Routing type value must ignore the Routing header and proceed to process the next header. An illustration of the MEADcast header layout can be found in Figure 4.1, along with an in-depth description of each field in Table 4.1. Due to the time frame of this thesis, we omit the usage of the optional port list as it does not affect the core MEADcast processing.

4.2 Router

This section provides an overview of the MEADcast router implementation in the Linux Kernel. The MEADcast routing software is developed and tested using the latest stable Kernel release, version 6.5.3, at the time of development. MEADcast routing capabilities are tightly coupled to the flow of IPv6 packets through the Kernel network stack, making it infeasible to create a separate Kernel module loadable at runtime. Therefore, we decided

Field	Desciption
Next Header	A 8-bit selector, identifying the type of the immediately following header [DH17]. Employs identical values as the IPv4 Protocol field (e.g. 17 = UDP, 59 = No Next Header for IPv6) [Aut24].
Hdr Ext Len	An 8-bit unsigned integer representing the length of the Routing header in 8-octet units, excluding the initial 8 octets [DH17].
Routing Type	An 8-bit identifier denoting a specific Routing header variant [DH17]. Given the absence of an existing Routing Type for MEADcast, we utilize the experimental values of 253 and 254 in accordance with RFC3692 [Nar04].
Segments Left	An 8-bit unsigned integer representing the number of remaining route segments, indicating the number of explicitly listed intermediate nodes yet to be traversed before reaching the final destination [DH17]. Remains fixed to zero and is not modified by MEADcast routers. Consequently, intermediate nodes that do not recognize the employed Routing type value are required to disregard the MEADcast header and proceed to process the subsequent header [DH17].
Num. Dst.	An 8-bit unsigned integer denoting the number of IPv6 addresses encoded in the address list. The field size sets a theoretical limit of 255 destinations.
Flags	A 2-bit Bitmap. The highest-order bit classifies the packet either as a data (0) or discovery (1) packet. The lowest-order bit indicates whether the packet is a discovery request (0) or response (1) [DT19]. Valid combinations comprise `00`, `10`, and `11`
Hops	An 6-bit unsigned integer, specifying the distance between the sender and a router, denoted in the number of MEADcast hops. This field is utilized during the discovery phase. The sender initializes the field with zero, and it is incremented by each intermediate MEADcast router encountered. The field size sets a theoretical limit of 63 hops.
Reserved	A 16-bit reserved field. Should be initialized to zero during transmission and disregarded upon reception.
Delivery Map	A 32-bit Bitmap, where each bit at position i indicates whether a router at index i in the address list has already been delivered (0 = delivered, 1 = not yet delivered) [DT19].
Router Map	A 32-bit Bitmap, where each bit at position i indicates whether an address at index i in the address list is a router (0 = receiver, 1 = router) [DT19].
Address List	A variable-length list comprising the IPv6 addresses of receivers and MEADcast routers. The field size of the Delivery and Router Bitmap imposes a practical limit of 32 addresses.

Table 4.1: MEADcast header field description

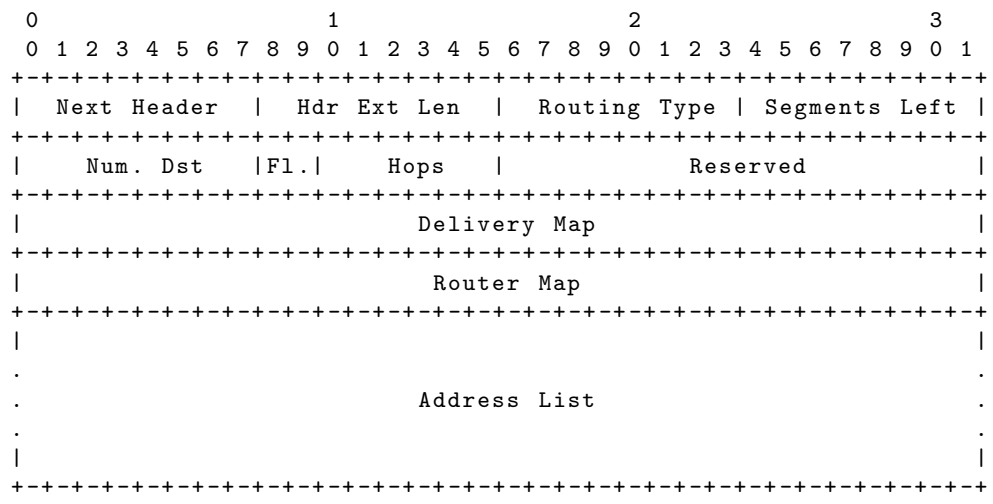

Figure 4.1: MEADcast header specification

to extend the IPv6 module with MEADcast capability. This feature can be enabled or disabled at compile-time using the `CONFIG_IPV6_MEADCAST` flag. To avoid the burden of recompiling or switching the kernel to enable or disable MEADcast support, we introduce a runtime configuration flag. This flag is managed through Sysctl, providing the virtual file `/proc/sys/net/ipv6/meadcast/enable`, which contains either 0 for disabled and 1 for enabled. For instance, the root user can enable MEADcast support by running the command `"echo 1 > /proc/sys/net/ipv6/meadcast/enable"`. However, it is crucial to ensure that this additional check does not adversely impact performance compared to a plain kernel, as it could potentially affect the reliability of experimental results.

As illustrated in Subsection 2.2.2, packets not designated to the router itself follow the "forwarding" packet flow in the Kernel. IPv6 extension headers of these packets are normally ignored by the Kernel (except for the Router Alert extension). Consequently, we embedded the entry point for MEADcast processing at the end of the `ip6_forward` method located in `net/ipv6/ipv6_output.c`. If a packet contains the routing header extension the `mdc_rcv` method located in `net/ipv6/mdcast.c` gets invoked. This method performs basic sanity checks on the MEADcast header and then invokes `mdc_dcv_rcv` for discovery and `mdc_data_rcv` for data packets.

Discovery: First, discovery packets get validated and the hop counter in the MEADcast header is incremented. To create a discovery response the socket buffer struct of the incoming discovery request is cloned. By this, we solely have to update the IP source and destination field of the cloned packet, set the response flag, and perform a routing table lookup for the sender's address. The source IP address of the discovery response is set to the IP address associated with the interface returned from the routing lookup. Lastly, the `dst_output` method is invoked, dispatching the packet to the corresponding layer 2 handler. The processing of the original packet does not need further handling, as it is already correctly done by the default forwarding flow.

Data: Data packets undergo several steps upon arrival at the MEADcast router. Firstly, they are validated, and the hop counter in the MEADcast header is incremented. Subsequently, the router iterates over the Delivery and Router Bitmap to identify undelivered

routers (indices set in both bitmaps). During this iteration, if the router is not the last undelivered router in the sequence, the socket buffer of the original packet is cloned and forwarded accordingly. However, if the router is the last undelivered router, the original packet can be forwarded without the need for costly replication. To determine whether to clone the packet, the routers monitors whether an undelivered router at index $r1$ has a successor (another undelivered router) at index $r2$, with $r1 < r2$. The algorithm's specifics are depicted in Figure 4.2.

For each undelivered router R_u specified in the MEADcast header, the processing router R_p decides whether to transmit the data via MEADcast or IP unicast. If the IP address of R_u in the MEADcast header belongs to the processing router R_p itself, all designated endpoints will be delivered via IP unicast. Additionally, if the number of endpoints assigned to the considered router R_u from the MEADcast header is below a certain threshold, the processing router R_p performs a premature MEADcast to IP unicast transformation. For example, if R_u has one designated endpoint, R_p could execute a premature MEADcast to IP unicast transformation, potentially enhancing performance. The threshold can also be configured during runtime by setting the parameter `/proc/sys/net/ipv6/meadcast/min_dsts`. Setting `min_dsts` to 0 disables premature MEADcast to IP unicast transformation.

If the packet is forwarded via MEADcast, the router updates both the delivery bitmap and the destination field of the IPv6 header. In the delivery bitmap, all bits are set to zero except for the index corresponding to the considered router. Additionally, the IP destination field is set to the address of the first receiver following the router in the address list. Lastly, the router performs a routing table lookup for the new destination address and forwards the packet accordingly.

In case of IP unicast transmission, the router converts the MEADcast packet into an IPv6 packet by copying the IPv6 header in front of the Layer 4 header and adjusting the socket buffer pointers accordingly. Additionally, the L4 checksum is corrected due to the modified destination IP address.

4.3 Sender

This section provides an overview of our MEADcast sender implementation. To ensure an efficient implementation that is comparable to existing network protocol implementations, we chose to implement the sender in the C language. The main tasks of the sender include sending periodical discovery requests to all group members, receiving discovery responses, constructing a topology tree based on the received discovery responses, heuristically grouping receivers into MEADcast packets, and transmitting data to all group members either via MEADcast or IP unicast.

Startup All receivers must be known at startup, dynamic runtime memberships are not supported. We regard this decision as justifiable, given that MEADcast does not define any mechanism for joining or leaving a group, and the primary objective of this thesis is to compare the protocol with existing alternatives. On startup the sender initializes a topology tree with itself as the root and all receivers as direct attached leafs. Moreover, a TUN interface is created, which is used as a common interface to other applications. Any data, which is send to the TUN interface will be transmitted to the MEADcast group. The Maximum transmission unit (MTU) of the TUN interface will be set to the MTU of the bind

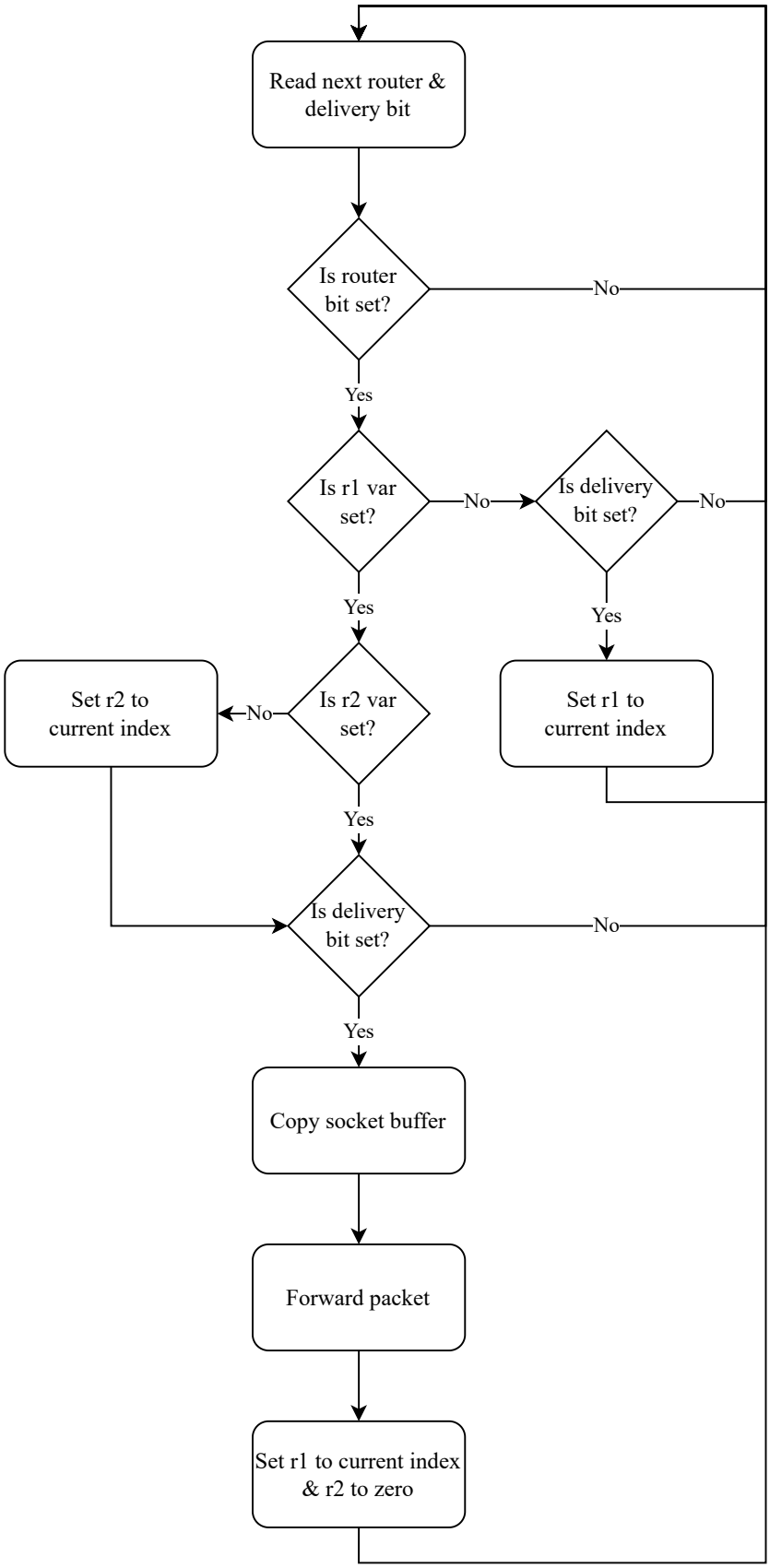

Figure 4.2: Router header parsing

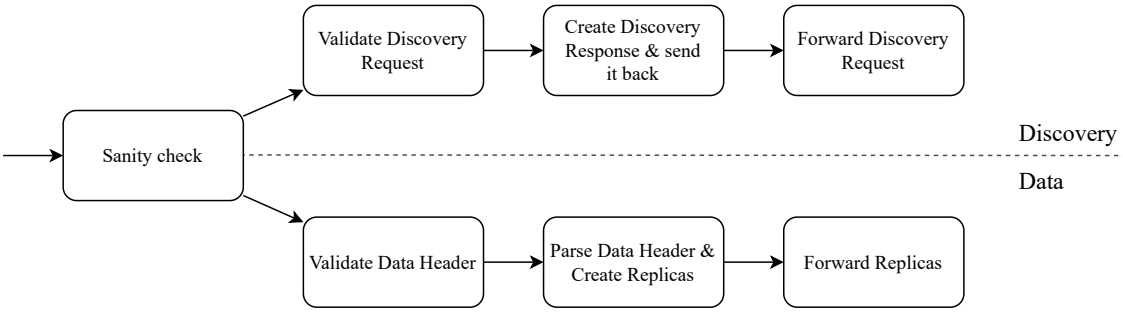

Figure 4.3: MEADcast router procedure

interface minus the maximum allowed MEADcast header size. This ensures, that all packets read from the TUN interface can be transmitted via MEADcast. Additionally, a host route to the TUN interface gets created (default: `fd15::1`). Next, a discovery and transmission thread is created. This is required to simultaneously transmit data and discovery requests and responses.

4.4 Testbed

5 Evaluation

6 Summary

6.1 Conclusion

6.2 Further Work

List of Figures

Acronyms

ALM Application-Layer Multicast. 9

AS Autonomous system. 19

CT Communication Technology. 1, 9

DoS denial-of-service. 15

IoT Internet of Things. 9

ISP Internet Service Provider. 2, 9

MSB most significant bit. 6

MTU Maximum transmission unit. 22

P2P peer-to-peer. 9, 11, 12

PDU Protocol Data Unit. 5

QoS Quality of Service. 16

SDN Software-defined networking. 3

SDU service data unit. 13

VoIP Voice over IP. 10

Bibliography

[Aca14] Cisco Networking Academy. *Connecting Networks. Companion Guide*. 1st ed. Cisco Press, May 2014. ISBN: 978-1-58713-332-9.

[AE15] Anders SG Andrae and Tomas Edler. "On global electricity usage of communication technology: trends to 2030". In: *Challenges* 6.1 (2015), pp. 117–157.

[Aru19] Aruba. *Aruba 2530 Multicast and Routing Guide for ArubaOS-Switch 16.09*. 1st ed. June 2019. URL: https://www.arubanetworks.com/techdocs/AOS-Switch/16.09/Aruba%202530%20Multicast%20and%20Routing%20Guide%20for%20AOS-S%20Switch%2016.09.pdf (visited on 09/20/2023).

[Aut24] Internet Assigned Numbers Authority. *Protocol Numbers*. Internet Assigned Numbers Authority, Jan. 8, 2024. URL: https://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml (visited on 02/15/2024).

[BGC04] Ali Boudani, Alexandre Guitton, and Bernard Cousin. "GXcast: Generalized explicit multicast routing protocol". In: *Proceedings. ISCC 2004. Ninth International Symposium on Computers And Communications (IEEE Cat. No. 04TH8769)*. Vol. 2. IEEE. 2004, pp. 1012–1017.

[Boi+07] Rick Boivie et al. *Explicit multicast (Xcast) concepts and options*. Tech. rep. 2007.

[Cai+02] Bradley Cain et al. *Internet Group Management Protocol, Version 3*. RFC 3376. Oct. 2002. DOI: 10.17487/RFC3376. URL: https://www.rfc-editor.org/info/rfc3376.

[Car21] Cartesian. *US-Broadband-Household-Bandwidth-Demand-Study*. July 2021. URL: https://www.cartesian.com/wp-content/uploads/2021/07/Cartesian_NCTA-US-Broadband-Household-Bandwidth-Demand-Study-July-2021.pdf (visited on 09/19/2023).

[Cis01] Cisco. *IP Multicast Technology Overview*. Cisco, Oct. 2001. URL: https://www.cisco.com/c/en/us/td/docs/ios/solutions_docs/ip_multicast/White_papers/mcst_ovr.html (visited on 10/24/2023).

[Cis08] Inc. Cisco Systems. *Enterprise Campus 3.0 Architecture: Overview and Framework*. Cisco Systems, Inc., Apr. 2008. URL: https://www.cisco.com/c/en/us/td/docs/solutions/Enterprise/Campus/campover.html (visited on 01/10/2024).

[Cis14] Inc. Cisco Systems. *Campus Wired LAN Technology Design Guide*. Cisco Systems, Inc., Aug. 2014. URL: https://www.cisco.com/c/dam/en/us/td/docs/solutions/CVD/Aug2014/CVD-CampusWiredLANDesignGuide-AUG14.pdf (visited on 01/10/2024).

[CJ13] Brian E. Carpenter and Sheng Jiang. *Transmission and Processing of IPv6 Extension Headers*. RFC 7045. Dec. 2013. DOI: 10.17487/RFC7045. URL: https://www.rfc-editor.org/info/rfc7045.

[CK13] Stuart Cheshire and Marc Krochmal. *Multicast DNS*. RFC 6762. Feb. 2013. DOI: 10.17487/RFC6762. URL: https://www.rfc-editor.org/info/rfc6762.

[Clo23] Cloudflare. *Cloudflare Radar Internet traffic trends*. Cloudflare, 2023. URL: https://radar.cloudflare.com/traffic (visited on 09/20/2023).

[DC90] Stephen E Deering and David R Cheriton. "Multicast routing in datagram internetworks and extended LANs". In: *ACM Transactions on Computer Systems (TOCS)* 8.2 (1990), pp. 85–110.

[Dee86] Dr. Steve E. Deering. *Host extensions for IP multicasting*. RFC 988. July 1986. DOI: 10.17487/RFC0988. URL: https://www.rfc-editor.org/info/rfc988.

[Dee89] Dr. Steve E. Deering. *Host extensions for IP multicasting*. RFC 1112. Aug. 1989. DOI: 10.17487/RFC1112. URL: https://www.rfc-editor.org/info/rfc1112.

[DH06] Dr. Steve E. Deering and Bob Hinden. *IP Version 6 Addressing Architecture*. RFC 4291. Feb. 2006. DOI: 10.17487/RFC4291. URL: https://www.rfc-editor.org/info/rfc4291.

[DH17] Dr. Steve E. Deering and Bob Hinden. *Internet Protocol, Version 6 (IPv6) Specification*. RFC 8200. July 2017. DOI: 10.17487/RFC8200. URL: https://www.rfc-editor.org/info/rfc8200.

[Dio+00] Christophe Diot et al. "Deployment issues for the IP multicast service and architecture". In: *IEEE network* 14.1 (2000), pp. 78–88.

[DT19] Vitalian Danciu and Cuong Ngoc Tran. "MEADcast: Explicit Multicast with Privacy Aspects". In: *International Journal on Advances in Security* 12.1 & 2 (2019), pp. 13–28. ISSN: 1942-2636.

[GL22] Fernando Gont and Will (Shucheng) LIU. *Recommendations on the Filtering of IPv6 Packets Containing IPv6 Extension Headers at Transit Routers*. RFC 9288. Aug. 2022. DOI: 10.17487/RFC9288. URL: https://www.rfc-editor.org/info/rfc9288.

[Gon+16] Fernando Gont et al. *Observations on the Dropping of Packets with IPv6 Extension Headers in the Real World*. RFC 7872. June 2016. DOI: 10.17487/RFC7872. URL: https://www.rfc-editor.org/info/rfc7872.

[Gon+21] Fernando Gont et al. *Operational Implications of IPv6 Packets with Extension Headers*. RFC 9098. Sept. 2021. DOI: 10.17487/RFC9098. URL: https://www.rfc-editor.org/info/rfc9098.

[Har+07] Szabolcs Harcsik et al. "Latency evaluation of networking mechanisms for game traffic". In: *Proceedings of the 6th ACM SIGCOMM workshop on Network and system support for games*. 2007, pp. 129–134.

[Hav98] Geoff Haviland. "Designing High-Performance Campus Intranets with Multilayer Switching". In: *White Paper, Cisco Systems, Inc* (1998). URL: https://web.archive.org/web/20000903190239/http://www.cisco.com/warp/public/cc/so/cuso/epso/entdes/highd_wp.pdf (visited on 01/12/2024).

[Hua23] Ltd. Huawei Technologies Co. "Typical Networking Architectures for Campus Networks and Case Practice". In: *Data Communications and Network Technologies*. Singapore: Springer Nature Singapore, 2023, pp. 471–500. ISBN: 978-981-19-3029-4. DOI: 10.1007/978-981-19-3029-4_15. URL: https://doi.org/10.1007/978-981-19-3029-4_15 (visited on 01/10/2024).

[IET21] IETF. *QUIC in the Internet industry*. IETF, June 3, 2021. URL: https://www.ietf.org/blog/quic-industry/ (visited on 09/23/2023).

[IT21] Jana Iyengar and Martin Thomson. *QUIC: A UDP-Based Multiplexed and Secure Transport*. RFC 9000. May 2021. DOI: 10.17487/RFC9000. URL: https://www.rfc-editor.org/info/rfc9000.

[ITU23] ITU. *Measuring digital development Facts and Figures 2022*. 2023. URL: https://www.itu.int/itu-d/reports/statistics/facts-figures-2022/ (visited on 09/17/2023).

[JC20] Matt Joras and Yang Chi. *How Facebook is bringing QUIC to billions*. Meta, Oct. 21, 2020. URL: https://engineering.fb.com/2020/10/21/networking-traffic/how-facebook-is-bringing-quic-to-billions/ (visited on 09/23/2023).

[KPP93] Vachaspathi P. Kompella, Joseph C. Pasquale, and George C. Polyzos. "Multicast routing for multimedia communication". In: *IEEE/ACM transactions on Networking* 1.3 (1993), pp. 286–292.

[LN93] Xiaola Lin and Lionel M. Ni. "Multicast communication in multicomputer networks". In: *IEEE transactions on Parallel and Distributed Systems* 4.10 (1993), pp. 1105–1117.

[Mal98] Gary S. Malkin. *RIP Version 2*. RFC 2453. Nov. 1998. DOI: 10.17487/RFC2453. URL: https://www.rfc-editor.org/info/rfc2453.

[Mog84] J.C. Mogul. *Broadcasting Internet datagrams in the presence of subnets*. RFC 922. Oct. 1984. DOI: 10.17487/RFC0922. URL: https://www.rfc-editor.org/info/rfc922.

[Moy98] John Moy. *OSPF Version 2*. RFC 2328. Apr. 1998. DOI: 10.17487/RFC2328. URL: https://www.rfc-editor.org/info/rfc2328.

[Nar04] Dr. Thomas Narten. *Assigning Experimental and Testing Numbers Considered Useful*. RFC 3692. Jan. 2004. DOI: 10.17487/RFC3692. URL: https://www.rfc-editor.org/info/rfc3692.

[Ngu19] Duc Minh Nguyen. "Deployment of MEADcast in Stub Software-Defined Networks". Ludwig-Maximilians-University of Munich, Mar. 31, 2019.

[Pos81] Jon Postel. *Internet Protocol*. RFC 791. Sept. 1981. DOI: 10.17487/RFC0791. URL: https://www.rfc-editor.org/info/rfc791.

[RES06] Sylvia Ratnasamy, Andrey Ermolinskiy, and Scott Shenker. "Revisiting IP multicast". In: *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*. 2006, pp. 15–26.

[RH03] Sandro Rafaeli and David Hutchison. "A survey of key management for secure group communication". In: *ACM Computing Surveys (CSUR)* 35.3 (2003), pp. 309–329.

[Ric15] Mark Richards. *Software architecture patterns*. 2015.

[Sim+07] William A. Simpson et al. *Neighbor Discovery for IP version 6 (IPv6)*. RFC 4861. Sept. 2007. DOI: 10.17487/RFC4861. URL: https://www.rfc-editor.org/info/rfc4861.

[ST02] Sherlia Y Shi and Jonathan S Turner. "Routing in overlay multicast networks". In: *Proceedings. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*. Vol. 3. IEEE. 2002, pp. 1200–1208.

[TD18] Cuong Ngoc Tran and Vitalian Danciu. "Privacy-Preserving Multicast to Explicit Agnostic Destinations". In: *Proceedings of the Eighth International Conference on Advanced Communications and Computation (INFOCOMP 2018)*. 2018, pp. 60–65.

[Zha+06] Beichuan Zhang et al. "Universal IP multicast delivery". In: *Computer Networks* 50.6 (2006), pp. 781–806.

[Zim80] Huber Zimmermann. "OSI reference model-the ISO model of architecture for open systems interconnection". In: *IEEE Transactions on communications* 28.4 (1980), pp. 425–432.