Stream: Independent Submission
RFC: 8743
Category: Informational
Published: March 2020
ISSN: 2070-1721
Authors: S. Kanugovi, *Nokia Bell Labs*; F. Baboescu, *Broadcom*; J. Zhu, *Intel*; S. Seo, *Korea Telecom*

RFC 8743
Multi-Access Management Services (MAMS)

Abstract

In multiconnectivity scenarios, the clients can simultaneously connect to multiple networks based on different access technologies and network architectures like Wi-Fi, LTE, and DSL. Both the quality of experience of the users and the overall network utilization and efficiency may be improved through the smart selection and combination of access and core network paths that can dynamically adapt to changing network conditions.

This document presents a unified problem statement and introduces a solution for managing multiconnectivity. The solution has been developed by the authors based on their experiences in multiple standards bodies, including the IETF and the 3GPP. However, this document is not an Internet Standards Track specification, and it does not represent the consensus opinion of the IETF.

This document describes requirements, solution principles, and the architecture of the Multi-Access Management Services (MAMS) framework. The MAMS framework aims to provide best performance while being easy to implement in a wide variety of multiconnectivity deployments. It specifies the protocol for (1) flexibly selecting the best combination of access and core network paths for the uplink and downlink, and (2) determining the user-plane treatment (e.g., tunneling, encryption) and traffic distribution over the selected links, to ensure network efficiency and the best possible application performance.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This is a contribution to the RFC Series, independently of any other RFC stream. The RFC Editor has chosen to publish this document at its discretion and makes no statement about its value for implementation or deployment. Documents approved for publication by the RFC Editor are not candidates for any level of Internet Standard; see Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc8743.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Table of Contents

1. Introduction

Multi-Access Management Services (MAMS) is a programmable framework that provides mechanisms for the flexible selection of network paths in a multi-access (MX) communication environment, based on the application's needs. The MAMS framework leverages network intelligence and policies to dynamically adapt traffic distribution across selected paths and user-plane treatments (e.g., encryption needed for transport over Wi-Fi, or tunneling needed to overcome a NAT between client and multipath proxy) to changing network/link conditions. The network path selection and configuration messages are carried as user-plane data between the functional elements in the network and the client, and thus without any impact on the control-plane signaling schemes of the underlying access networks. For example, in a multi-access network with LTE and Wi-Fi technologies, existing LTE and Wi-Fi signaling procedures will be used to set up the LTE and Wi-Fi connections, respectively, and MAMS-specific control-plane messages are carried as LTE or Wi-Fi user-plane data. The MAMS framework defined in this document provides the capability to make a smart selection of a flexible combination of access paths and core network paths, as well as to choose the user-plane treatment when the traffic is distributed across the selected paths. Thus, it is a broad programmable framework that provides functions beyond the simple sharing of network policies such as those provided by the Access Network Discovery and Selection Function (ANDSF) [ANDSF], which offers policies and rules for assisting 3GPP clients to discover and select available access networks. Further, it allows the choice and configuration of user-plane treatment for the traffic over the paths, depending on the application's needs.

The MAMS framework mechanisms are not dependent on any specific access network types or user-plane protocols (e.g., TCP, UDP, Generic Routing Encapsulation (GRE) [RFC2784] [RFC2890], Multipath TCP (MPTCP) [RFC6824]). The MAMS framework coexists and complements the existing protocols by providing a way to negotiate and configure those protocols to match their use to a given multi-access scenario based on client and network capabilities, and the specific needs of each access network path. Further, the MAMS framework allows load balancing of the traffic flows across the selected access network paths, and the exchange of network state information to be used for network intelligence to optimize the performance of such protocols.

This document presents the requirements, solution principles, functional architecture, and protocols for realizing the MAMS framework. An important goal for the MAMS framework is to ensure that it requires either minimum dependency or (better) no dependency on the actual access technologies of the participating links, beyond the fact that MAMS functional elements form an IP overlay across the multiple paths. This allows the scheme to be "future proof" by allowing independent technology evolution of the existing access and core networks as well as seamless integration of new access technologies.

The solution described in this document has been developed by the authors, based on their experiences in multiple standards bodies, including the IETF and the 3GPP. However, this document is not an Internet Standards Track specification, and it does not represent the consensus opinion of the IETF.

2. Terminology

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**NOT RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

Client: An end-user device that supports connections with multiple access nodes, possibly over different access technologies. Also called a user device or user equipment (UE).

Multiconnectivity Client: A client with multiple network connections.

Access Network: The segment in the network that delivers user data packets to the client via an access link such as a Wi-Fi airlink, an LTE airlink, or DSL.

Core: The functional element that anchors the client IP address used for communication with applications via the network.

Network Connection Manager (NCM): A functional entity in the network that handles MAMS control messages from the client and configures the distribution of data packets over the available access and core network paths, and manages the user-plane treatment (e.g., tunneling, encryption) of the traffic flows.

Client Connection Manager (CCM): A functional entity in the client that exchanges MAMS signaling messages with the NCM, and which configures the network paths at the client for the transport of user data.

Network Multi-Access Data Proxy (N-MADP): A functional entity in the network that handles the forwarding of user data traffic across multiple network paths. The N-MADP is responsible for MAMS-related user-plane functionalities in the network.

Client Multi-Access Data Proxy (C-MADP): A functional entity in the client that handles the forwarding of user data traffic across multiple network paths. The C-MADP is responsible for MAMS-related user-plane functionalities in the client.

Anchor Connection: Refers to the network path from the N-MADP to the user-plane gateway (IP anchor) that has assigned an IP address to the client.

Delivery Connection: Refers to the network path from the N-MADP to the client.

Uplink (also referred to as "UL" in this document): Refers to the direction of a connection from a client toward the network.

Downlink (also referred to as "DL" in this document): Refers to the direction of a connection from the network toward a client.

3. Problem Statement

Typically, a client has access to multiple communication networks based on different technologies for accessing application services, for example, LTE, Wi-Fi, DSL, or MulteFire. Different technologies exhibit benefits and limitations in different scenarios. For example, Wi-Fi provides high throughput for end users when their Wi-Fi coverage is good, but the throughput degrades significantly as a given user moves closer to the edge of its Wi-Fi coverage area (typically in the range of a few tens of meters) or if the user population is large (due to a contention-based Wi-Fi access scheme). In LTE networks, the capacity is often constrained by the limited availability of licensed spectrum. However, the quality of the service is predictable even in multi-user scenarios, due to controlled scheduling and licensed-spectrum usage.

Additionally, the use of a particular access network path is often coupled with the use of its associated core network and the services that are offered by that network. For example, in an enterprise that has deployed both Wi-Fi and LTE networks, the enterprise services, such as printers and corporate audio/video conferencing, are accessible only via Wi-Fi access connected to the enterprise-hosted (Wi-Fi) core, whereas the LTE access can be used to get operator services, including access to the public Internet.

Thus, application performance in different scenarios becomes dependent on the choice of access networks (e.g., Wi-Fi, LTE) and the network and transport protocols used (e.g., VPN, MPTCP, GRE). Therefore, to achieve the best possible application performance in a wide range of scenarios, a framework is needed that allows the selection and flexible combination of access and core network paths as well as the protocols used for uplink and downlink data delivery.

For example, in uncongested scenarios and when the user's Wi-Fi coverage is good, to ensure best performance for enterprise applications at all times, it would be beneficial to use Wi-Fi access for both the uplink and downlink for connecting to enterprise applications. However, in congested scenarios or when the user is getting close to the edge of its Wi-Fi coverage area, the use of Wi-Fi in the uplink by multiple users can lead to degraded capacity and increased delays due to contention. In this case, it would be beneficial to at least use the LTE access for increased uplink coverage, while Wi-Fi may still continue to be used for the downlink.

4. Requirements

The requirements set out in this section define the behavior of the MAMS mechanism and the related functional elements.

4.1. Access-Technology-Agnostic Interworking

The access nodes **MAY** use different technology types (LTE, Wi-Fi, etc.). The framework, however, **MUST** be agnostic about the type of underlying technology used by the access network.

4.2. Support for Common Transport Deployments

The network path selection and user data distribution **MUST** work transparently across various transport deployments that include end-to-end IPsec, VPNs, and middleboxes like NATs and proxies.

4.3. Independent Access Path Selection for Uplink and Downlink

A client **SHOULD** be able to transmit on the uplink and receive on the downlink, using one or more access networks. The selections of the access paths for the uplink and downlink **SHOULD** happen independently.

4.4. Core Selection Independent of Uplink and Downlink Access

A client **SHOULD** flexibly select the core independently of the access paths used to reach the core, depending on the application's needs, local policies, and the result of MAMS control-plane negotiation.

4.5. Adaptive Access Network Path Selection

The framework **MUST** have the ability to determine the quality of each of the network paths, e.g., access link delay and capacity. This information regarding network path quality needs to be considered in the logic for the selection of the combination of network paths to be used for transporting user data. The path selection algorithm can use the information regarding network path quality, in addition to other considerations like network policies, for optimizing network usage and enhancing the Quality of Experience (QoE) delivered to the user.

4.6. Multipath Support and Aggregation of Access Link Capacities

The framework **MUST** support the distribution and aggregation of user data across multiple network paths at the IP layer. The client **SHOULD** be able to leverage the combined capacity of the multiple network connections by enabling the simultaneous transport of user data over multiple network paths. If required, packet reordering needs to be done at the receiver. The framework **MUST** allow the flexibility to choose the flow-steering and aggregation protocols based on capabilities supported by the client and the network user-plane entities. The multiconnection aggregation solution **MUST** support existing transport and network-layer protocols like TCP, UDP, and GRE. The framework **MUST** allow the use and configuration of existing aggregation protocols such as MPTCP and SCTP [RFC4960].

4.7. Scalable Mechanism Based on User-Plane Interworking

The framework **MUST** leverage commonly available transport, routing, and tunneling capabilities to provide user-plane interworking functionality. The addition of functional elements in the user-plane path between the client and the network **MUST NOT** impact the access-technology-specific procedures. This makes the solution easy to deploy and scale when different networks are added and removed.

4.8. Separate Control-Plane and User-Plane Functions

The client **MUST** use the control-plane protocol to negotiate the following with the network: (1) the choice of access and core network paths for both the uplink and downlink, and (2) the user-plane protocol treatment. The control plane **MUST** configure the actual user-plane data distribution function per this negotiation. A common control protocol **SHOULD** allow the creation of multiple user-plane function instances with potentially different user-plane (e.g., tunneling) protocol types. This enables maintaining a clear separation between the control-plane and user-plane functions, allowing the framework to be scalable and extensible, e.g., using architectures and implementations based on Software-Defined Networking (SDN).

4.9. Lossless Path (Connection) Switching

When switching data traffic from one path (connection) to another, packets may be lost or delivered out of order; this will have negative impact on the performance of higher-layer protocols, e.g., TCP. The framework **SHOULD** provide the necessary mechanisms to ensure in-order delivery at the receiver, e.g., during path switching. The framework **MUST NOT** cause any packet loss beyond losses that access network mobility functions may cause.

4.10. Concatenation and Fragmentation for Adaptation to MTU Differences

Different network paths may have different security and middlebox (e.g., NAT) configurations. These configurations will lead to the use of different tunneling protocols for the transport of data between the network user-plane function and the client. As a result, different effective payload sizes per network path are possible (e.g., due to variable encapsulation header overheads). Hence, the MAMS framework **SHOULD** support the fragmentation of a single payload across MTU-sized IP packets to avoid IP packet fragmentation when aggregating packets from different paths. Further, the concatenation of multiple IP packets into a single IP packet to improve efficiency in packing the MTU size **SHOULD** also be supported.

4.11. Configuring Network Middleboxes Based on Negotiated Protocols

The framework **SHOULD** enable the identification of optimal settings, like radio link dormancy timers, binding expiry times, and supported MTUs, based on parameters negotiated between the client and the network, that may be used to configure middleboxes for efficient operation of user-plane protocols, e.g., configuring a NAT with a longer binding expiry time when UDP versus TCP is used.

4.12. Policy-Based Optimal Path Selection

The framework **MUST** support both the implementation of policies at the client and guidance from the network for network path selection that will address different application requirements.

4.13. Access-Technology-Agnostic Control Signaling

The control-plane signaling **MUST NOT** be dependent on the underlying access technology procedures, i.e., it is carried transparently, like application data, on the user plane. The MAMS framework **SHOULD** support the delivery of control-plane signaling over existing Internet protocols, e.g., TCP or UDP.

4.14. Service Discovery and Reachability

There can be multiple instances of the control-plane and user-plane functional elements of the framework, either collocated or hosted on separate network elements and reachable via any of the available user-plane paths. The client **MUST** have the flexibility to choose the appropriate control-plane instance in the network and use the control-plane signaling to choose the desired user-plane functional element instances. The client's choice can be based on considerations such as, but not limited to, the quality of the link through which the network function is reachable, client preferences, preconfiguration, etc.

5. Solution Principles

This document describes the Multi-Access Management Services (MAMS) framework for dynamic selection of a flexible combination of access and core network paths for the uplink and downlink, as well as the user-plane treatment for the traffic spread across the selected links. The user-plane paths, and access and core network connections, can be selected independently for the uplink and downlink. For example, the network paths chosen for the uplink do not apply any constraints on the choice of paths for the downlink. The uplink and downlink network paths can be chosen based on the application needs and on the characteristics and available resources on different network connections. For example, a Wi-Fi connection can be chosen for the downlink for transporting high-bandwidth data from the network to the client, whereas an LTE connection can be chosen to carry the low-bandwidth feedback to the application server.

Also, depending on the characteristics of the access network link, different processing would be needed on the user-plane packets on different network paths. Encryption would be needed on a Wi-Fi link to secure user-plane packets, but not on an LTE link. Tunneling would be needed to ensure client and network end-point reachability over NATs. Such differentiated user-plane treatment can be accomplished by configuration of user plane-protocols (e.g., IPsec) specific to each link.

The MAMS framework consists of clearly separated control- and user-plane functions in the network and the client. The control-plane protocol allows the configuration of the user-plane protocols and desired network paths for the transport of application traffic. The control-plane messages are carried as user-plane data over any of the available network paths between the peer control-plane functional elements in the client and the network. Multiple user-plane paths are dynamically distributed across multiple access networks and aggregated in the network (by the N-MADP). The access network's diversity is not exposed to the application servers, but is kept

within the scope of the elements defined in this framework. This reduces the burden placed on application servers that would otherwise have to react to access link changes caused by mobility events or changing link characteristics.

The selection of paths and user-plane treatment of the traffic is based on (1) the negotiation of client and network capabilities, and (2) link probing (i.e., checking the quality of links between the user-plane functional elements at the client and the network). This framework enables leveraging network intelligence to set up and dynamically configure the best access network path combination based on client and network capabilities, an application's needs, and knowledge of the network state.

6. MAMS Reference Architecture

Figure 1 illustrates the MAMS architecture for the scenario where a client is served by multiple (n) networks. It also introduces the following functional elements:

- The NCM and the CCM in the control plane.
- The N-MADP and the C-MADP in the user plane.

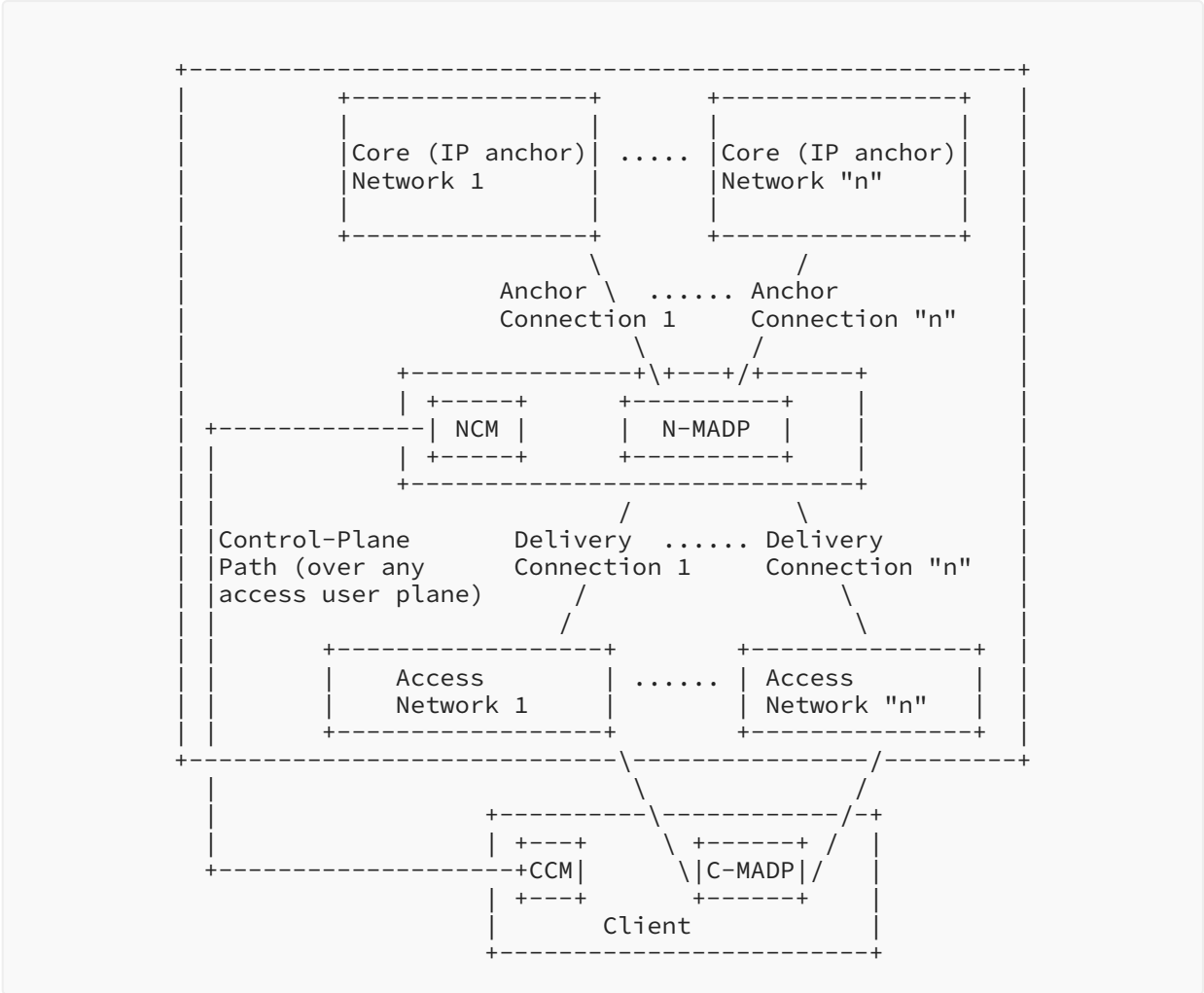

Figure 1: MAMS Reference Architecture

The NCM is the functional element in the network that handles the MAMS control-plane procedures. It configures the network (N-MADP) and client (C-MADP) user-plane functions, such as negotiating with the client for the use of available access network paths, protocols, and rules for processing the user-plane traffic, as well as link-monitoring procedures. The control-plane messages between the NCM and the CCM are transported as an overlay on the user plane, without any impact on the underlying access networks.

The CCM is the peer functional element in the client for handling MAMS control-plane procedures. It manages multiple network connections at the client. The CCM exchanges MAMS signaling messages with the NCM to support such functions as the configuration of the UL and DL user network path for transporting user data packets and the adaptive selection of network path by the NCM by reporting on the results of link probing. In the downlink, for user data received by the client, it configures the C-MADP such that application data packets can be received over any access link so that the packets will reach the appropriate application on the client. In the uplink,

for the data transmitted by the client, it configures the C-MADP to determine the best access links to be used for uplink data based on a combination of local and network policies delivered by the NCM.

The N-MADP is the functional element in the network that handles the forwarding of user data traffic across multiple network paths, as well as other user-plane functionalities (e.g., encapsulation, fragmentation, concatenation, reordering, retransmission). The N-MADP is the distribution node that routes (1) the uplink user-plane traffic to the appropriate anchor connection toward the core network, and (2) the downlink user traffic to the client over the appropriate delivery connections. In the downlink, the NCM configures the use of delivery connections and user-plane protocols at the N-MADP for transporting user data traffic. The N-MADP **SHOULD** implement ECMP support for the downlink traffic. Alternatively, it **MAY** be connected to a router with ECMP functionality. The load-balancing algorithm at the N-MADP is configured by the NCM, based on static and/or dynamic network policies like assigning access and core paths for a specific user data traffic type, user-volume-based percentage distribution, and link availability and feedback information from the exchange of MAMS signaling messages with the CCM at the client. The N-MADP can be configured with appropriate user-plane protocols to support both per-flow and per-packet traffic distribution across the delivery connections. In the uplink, the N-MADP selects the appropriate anchor connection over which to forward the user data traffic received from the client (via the delivery connections). The forwarding rules in the uplink at the N-MADP are configured by the NCM based on application requirements, e.g., enterprise-hosted application flows via a Wi-Fi anchor or mobile-operator-hosted applications via the cellular core.

The C-MADP is the functional element in the client that handles the MAMS user-plane data procedures. The C-MADP is configured by the CCM, based on the signaling exchange with the NCM and local policies at the client. The CCM configures the selection of delivery connections and the user-plane protocols to be used for uplink user data traffic based on the signaling messages exchanged with the NCM. The C-MADP entity handles the forwarding of user-plane data across multiple delivery connections and associated user-plane functions (e.g., encapsulation, fragmentation, concatenation, reordering, retransmissions).

The NCM and N-MADP can be either collocated or instantiated on different network nodes. The NCM can set up multiple N-MADP instances in the network. The NCM controls the selection of the N-MADP instance by the client and the rules for the distribution of user traffic across the N-MADP instances. This is beneficial in multiple deployment scenarios, like the following examples:

- Different N-MADP instances to handle different sets of clients for load balancing across clients.
- Network topologies where the N-MADP is hosted at the user-plane node at the access edge or in the core network, while the NCM is hosted at the access edge node.
- Access network technology architecture with an N-MADP instance at the core network node to manage traffic distribution across LTE and DSL networks, and an N-MADP instance at an access network node to manage traffic distribution across LTE and Wi-Fi networks.

- A single client can be configured to use multiple N-MADP instances. This is beneficial in addressing different application requirements. For example, separate N-MADP instances to handle traffic that is based on TCP and UDP transport.

Thus, the MAMS architecture flexibly addresses multiple network deployments.

7. MAMS Protocol Architecture

This section describes the protocol structure for the MAMS user-plane and control-plane functional elements.

7.1. MAMS Control-Plane Protocol

Figure 2 shows the default MAMS control-plane protocol stack. WebSocket [RFC6455] is used for transporting management and control messages between the NCM and the CCM.

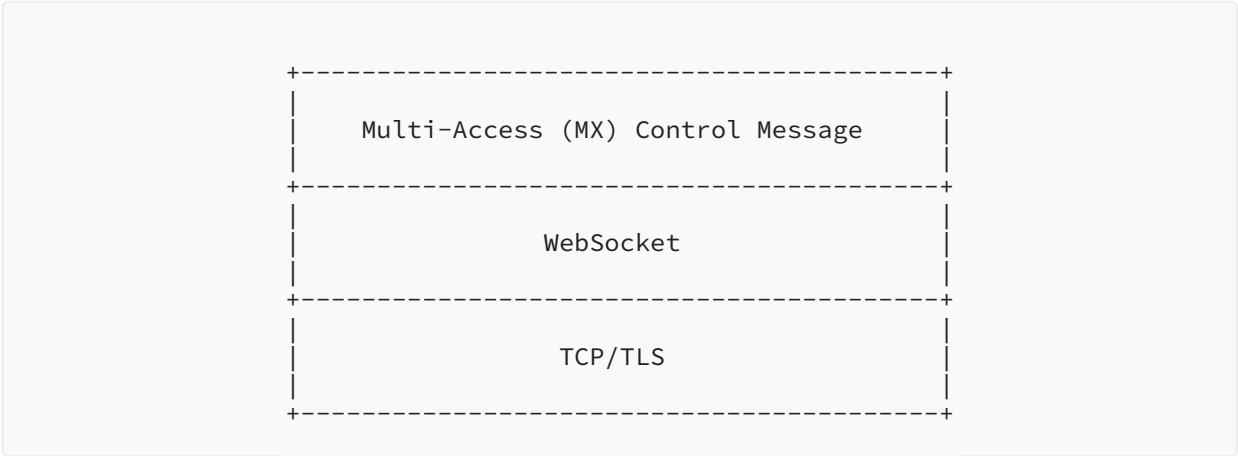

Figure 2: TCP-Based MAMS Control-Plane Protocol Stack

7.2. MAMS User-Plane Protocol

Figure 3 shows the MAMS user-plane protocol stack for transporting the user payload, e.g., an IP Protocol Data Unit (PDU).

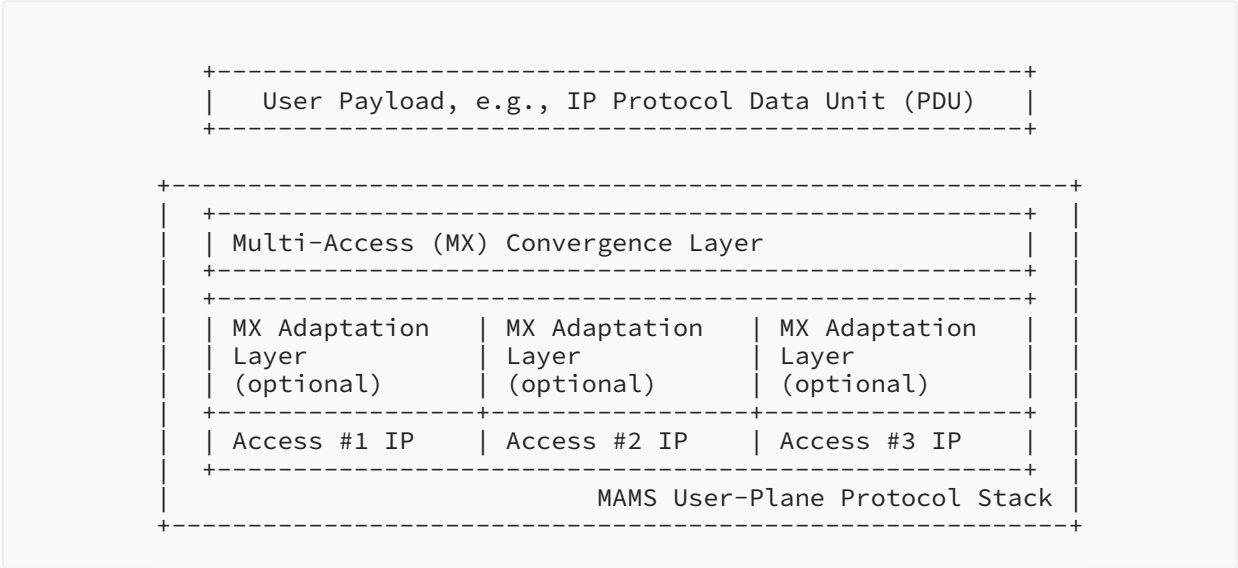

Figure 3: MAMS User-Plane Protocol Stack

The MAMS user-plane protocol consists of the following two layers:

- Multi-Access (MX) Convergence Layer: The MAMS framework configures the Convergence Layer to perform multi-access-specific tasks in the user plane. This layer performs such functions as access (path) selection, multi-link (path) aggregation, splitting/reordering, lossless switching, fragmentation, or concatenation. The MX Convergence Layer can be implemented by using existing user-plane protocols like MPTCP [RFC6824] or Multipath QUIC (MPQUIC) [QUIC-MULTIPATH], or by adapting encapsulating header/trailer schemes such as GRE [RFC2784] [RFC2890] or Generic Multi-Access (GMA) [INTAREA-GMA].
- Multi-Access (MX) Adaptation Layer: The MAMS framework configures the Adaptation Layer to address transport-network-related aspects such as reachability and security in the user plane. This layer performs functions to handle tunneling, network-layer security, and NAT. The MX Adaptation Layer can be implemented using IPsec, DTLS [RFC6347], or a Client NAT (Source NAT at the client with inverse mapping at the N-MADP [INTAREA-MAMS]). The MX Adaptation Layer is **OPTIONAL** and can be independently configured for each of the access links. For example, in a deployment with LTE (assumed secure) and Wi-Fi (assumed to not be secure), the MX Adaptation Layer can be omitted for the LTE link, but is configured with IPsec to secure the Wi-Fi link. Further details on the MAMS user plane are provided in [INTAREA-MAMS].

8. MAMS Control-Plane Procedures

8.1. Overview

The CCM and NCM exchange signaling messages to configure the user-plane functions via the C-MADP and the N-MADP at the client and the network, respectively. The means for the CCM to obtain the NCM credentials (Fully Qualified Domain Name (FQDN) or IP address) for sending the initial discovery messages are out of scope for this document. As an example, the client can

obtain the NCM credentials by using such methods as provisioning or DNS queries. Once the discovery process is successful, the (initial) NCM can update and assign additional NCM addresses, e.g., based on Mobile Country Code (MCC) / Mobile Network Code (MNC) tuple information received in the MX Discover message, for sending subsequent control-plane messages.

The CCM discovers and exchanges capabilities with the NCM. The NCM provides the credentials of the N-MADP endpoint and negotiates the parameters for the user plane with the CCM. The CCM configures the C-MADP to set up the user-plane path (e.g., MPTCP/UDP Proxy connection) with the N-MADP, based on the credentials (e.g., (MPTCP/UDP) Proxy IP address and port, associated core network path), and the parameters exchanged with the NCM. Further, the NCM and CCM exchange link status information to adapt traffic steering and user-plane treatment to dynamic network conditions. The key procedures are described in detail in the following subsections.

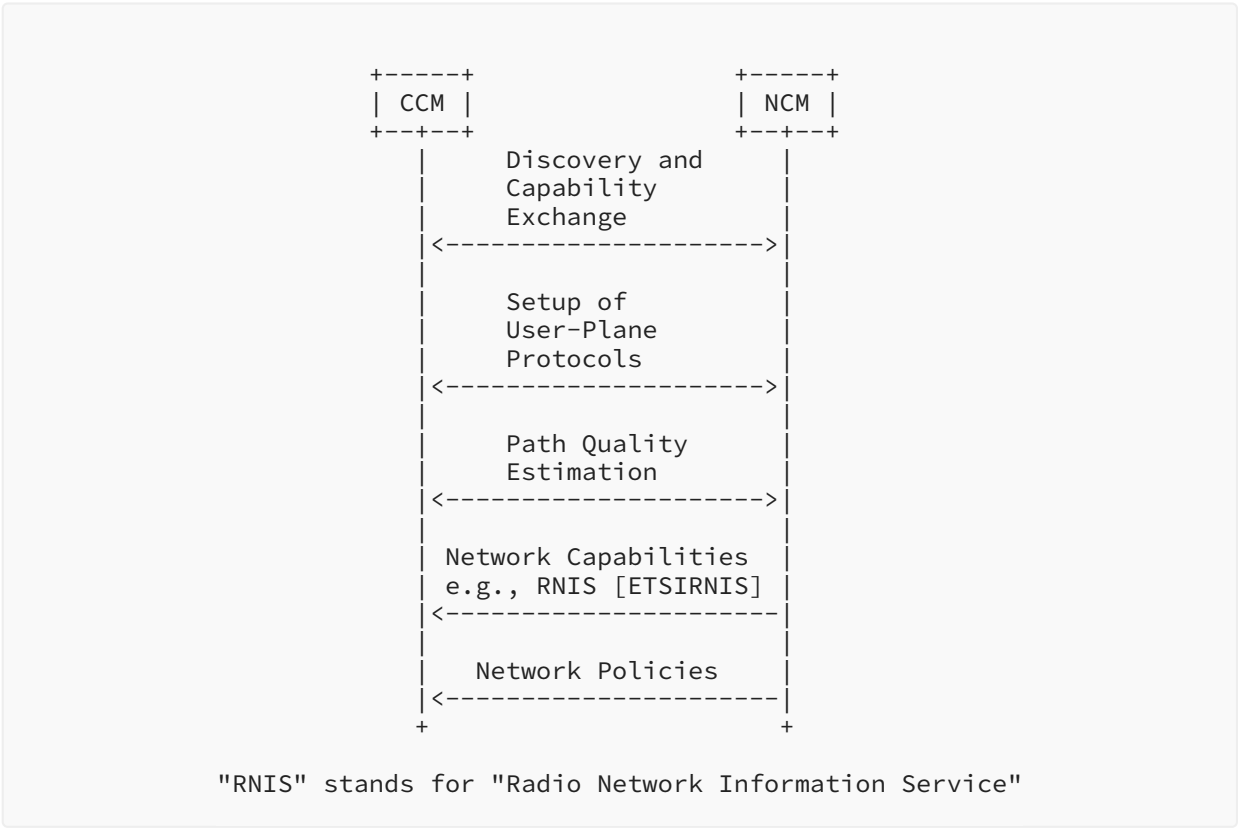

Figure 4: MAMS Control-Plane Procedures

8.2. Common Fields in MAMS Control Messages

Each MAMS control message consists of the following common fields:

- Version: Indicates the version of the MAMS control protocol.

- Message Type: Indicates the type of the message, e.g., MX Discover, MX Capability Request (REQ) / Response (RSP).
- Sequence Number: Auto-incremented integer to uniquely identify a particular message exchange, e.g., MX Capability Request/Response.

8.3. Common Procedures for MAMS Control Messages

This section describes the common procedures for MAMS control messages.

8.3.1. Message Timeout

After sending a MAMS control message, the MAMS control-plane peer (NCM or CCM) waits for a duration of MAMS_TIMEOUT ms before timing out in cases where a response was expected. The sender of the message will retransmit the message for MAMS_RETRY times before declaring failure if no response is received. A failure implies that the MAMS peer is dead or unreachable, and the sender reverts to native non-multi-access / single-path mode. The CCM may initiate the MAMS discovery procedure for re-establishing the MAMS session.

8.3.2. Keep-Alive Procedure

MAMS control-plane peers execute the keep-alive procedures to ensure that the other peers are reachable and to recover from dead-peer scenarios. Each MAMS control-plane endpoint maintains a Keep-Alive timer that is set for a duration of MAMS_KEEP_ALIVE_TIMEOUT. The Keep-Alive timer is reset whenever the peer receives a MAMS control message. When the Keep-Alive timer expires, an MX Keep-Alive Request is sent.

The values for MAMS_RETRY and MAMS_KEEP_ALIVE_TIMEOUT parameters used in keep-alive procedures are deployment dependent, and the means for obtaining them are out of scope for this document. As an example, the client and network can obtain the values using provisioning. On receipt of an MX Keep-Alive Request, the receiver responds with an MX Keep-Alive Response. If the sender does not receive a MAMS control message in response to MAMS_RETRY retries of the MX Keep-Alive Request, the MAMS peer declares that the peer is dead or unreachable. The CCM **MAY** initiate the MAMS discovery procedure for re-establishing the MAMS session.

Additionally, the CCM **SHALL** immediately send an MX Keep-Alive Request to the NCM whenever it detects a handover from one base station / access point to another. During this time, the client **SHALL** stop using MAMS user-plane functionality in the uplink direction until it receives an MX Keep-Alive Response from the NCM.

The MX Keep-Alive Request includes the following information:

- Reason: Can be timeout or handover. Handover shall be used by the CCM only on detection of a handover.
- Unique Session ID: See Section 8.4.
- Connection ID: If the reason is handover, the inclusion of this field is mandatory.
- Delivery Node ID: Identity of the node to which the client is attached. In the case of LTE, this is an E-UTRAN Cell Global Identifier (ECGI). In the case of Wi-Fi, this is an AP ID or a Media

Access Control (MAC) address. If the reason is "Handover", the inclusion of this field is mandatory.

8.4. Discovery and Capability Exchange

Figure 5 shows the MAMS discovery and capability exchange procedure.

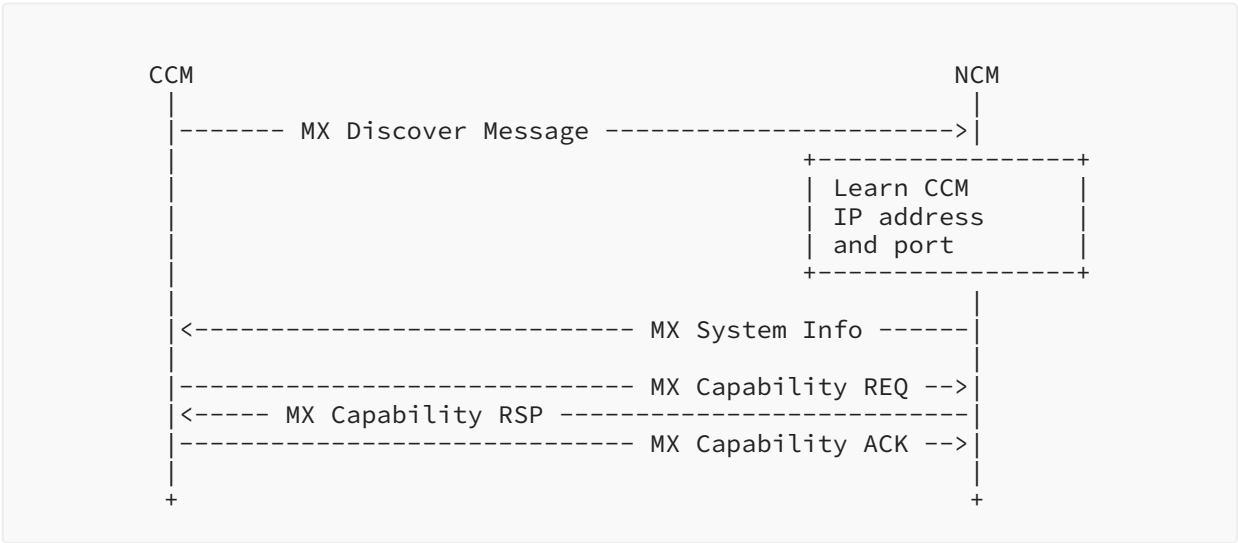

Figure 5: MAMS Control Procedure for Discovery and Capability Exchange

This procedure consists of the following key steps:

Step 1 (discovery): The CCM periodically sends an MX Discover message to a predefined (NCM) IP address/port until an MX System Info message is received in acknowledgment.

- The MX Discover message includes the following information:
  - MAMS Version.
  - Mobile Country Code (MCC) / Mobile Network Code (MNC) Tuple: Optional parameter to identify the operator network to which the client is subscribed, in conformance with the format specified in [ITU-E212].
- The MX System Info message includes the following information:
  - Number of Anchor Connections.

    For each anchor connection, the following parameters are included:

    - Connection ID: Unique identifier for the anchor connection.
    - Connection Type (e.g., Wi-Fi, 5G NR, MulteFire, LTE).
    - NCM Endpoint Address (for control-plane messages over this connection):
      - IP Address or FQDN

        - Port Number

Step 2 (capability exchange): The CCM learns the IP address and port from the MX System Info message. It then sends the MX Capability REQ message, which includes the following parameters:

- MX Feature Activation List: Indicates whether the corresponding feature is supported or not, e.g., lossless switching, fragmentation, concatenation, uplink aggregation, downlink aggregation, measurement, probing.
- Number of Anchor Connections (core networks).

  For each anchor connection, the following parameters are included:

  - Connection ID
  - Connection Type (e.g., Wi-Fi, 5G NR, MulteFire, LTE)

- Number of Delivery Connections (access links).

  For each delivery connection, the following parameters are included:

  - Connection ID
  - Connection Type (e.g., Wi-Fi, 5G NR, MulteFire, LTE)

- MX Convergence Method Support List:

  - GMA
  - MPTCP Proxy
  - GRE Aggregation Proxy
  - MPQUIC

- MX Adaptation Method Support List:

  - UDP without DTLS
  - UDP with DTLS
  - IPsec [RFC3948]
  - Client NAT

In response, the NCM creates a unique identity for the CCM session and sends the MX Capability Response, including the following information:

- MX Feature Activation List: Indicates whether the corresponding feature is enabled or not, e.g., lossless switching, fragmentation, concatenation, uplink aggregation, downlink aggregation, measurement, probing.
- Number of Anchor Connections (core networks):

  For each anchor connection, the following parameters are included:

  - Connection ID

- Connection Type (e.g., Wi-Fi, 5G NR, MulteFire, LTE)

- Number of Delivery Connections (access links):

  For each delivery connection, the following parameters are included:

  - Connection ID
  - Connection Type (e.g., Wi-Fi, 5G NR, MulteFire, LTE)

- MX Convergence Method Support List:

  - GMA
  - MPTCP Proxy
  - GRE Aggregation Proxy
  - MPQUIC

- MX Adaptation Method Support List:

  - UDP without DTLS
  - UDP with DTLS
  - IPsec [RFC3948]
  - Client NAT

- Unique Session ID: Unique session identifier for the CCM that set up the connection. If the session already exists, then the existing unique session identifier is returned.

  - NCM ID: Unique identity of the NCM in the operator network.
  - Session ID: Unique identity assigned to the CCM instance by this NCM instance.

In response to the MX Capability Response, the CCM sends a confirmation (or rejection) in the MX Capability Acknowledge. The MX Capability Acknowledge includes the following parameters:

- Unique Session ID: Same identifier as the identifier provided in the MX Capability Response.
- Acknowledgment: An indication of whether the client has accepted or rejected the capability exchange phase.

  - MX ACCEPT: The CCM accepts the capability set proposed by the NCM.
  - MX REJECT: The CCM rejects the capability set proposed by the NCM.

If the NCM receives an MX_REJECT, the current MAMS session will be terminated.

If the CCM can no longer continue with the current capabilities, it **SHOULD** send an MX Session Termination Request to terminate the MAMS session. In response, the NCM **SHOULD** send an MX Session Termination Response to confirm the termination.

8.5. User-Plane Configuration

Figure 6 shows the user-plane (UP) configuration procedure.

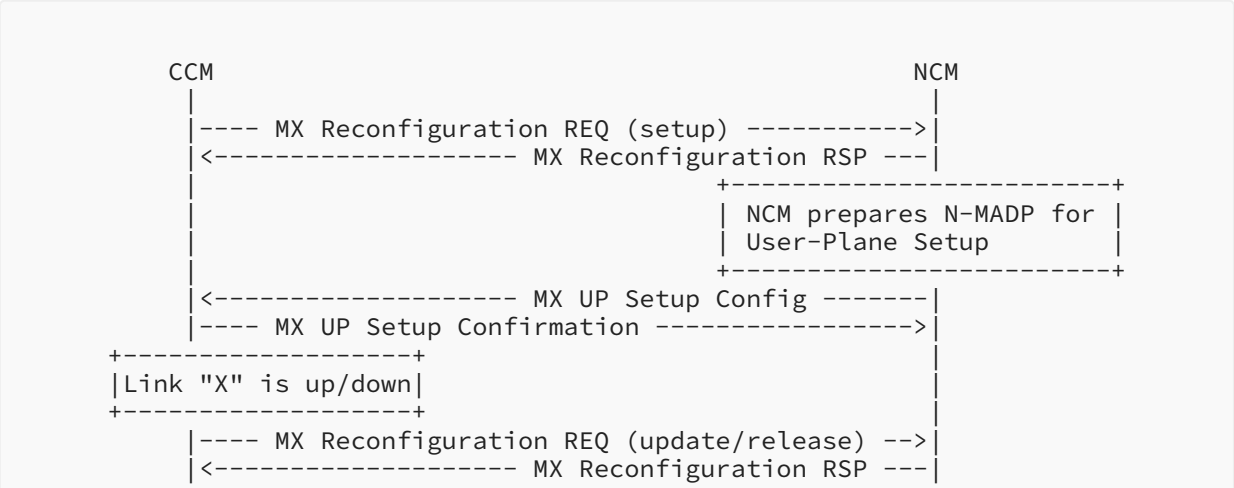

Figure 6: MAMS Control Procedure for User-Plane Configuration

This procedure consists of the following two key steps:

- Reconfiguration: The CCM informs the NCM about the changes to the client's connections - setup of a new connection, teardown of an existing connection, or update of parameters related to an existing connection. It consists of the client triggering the procedure by requesting an update to the connection configuration, and a response from the NCM.
- UP Setup: The NCM configures the user-plane protocols at the client and the network. The NCM initiates the UP setup by sending the MX UP Setup Configuration Request to the client, which confirms the set of mutually acceptable parameters by using the User Plane Setup Confirmation (CNF) message.

These steps are elaborated as follows.

Reconfiguration: When the client detects that the link is up/down or the IP address changes (e.g., via APIs provided by the client OS), the CCM sends an MX Reconfiguration Request to set up, update, or release the connection. The message **SHOULD** include the following information:

- Unique Session ID: Identity of the CCM at the NCM, created by the NCM during the capability exchange phase.
- Reconfiguration Action: Indicates the reconfiguration action (release, setup, or update).
- Connection ID: Identifies the connection for reconfiguration.

If the Reconfiguration Action is set to "setup" or "update", then the message includes the following parameters:

- IP address of the connection.
- SSID (Service Set Identifier of the Wi-Fi connection).
- MTU of the connection: The MTU of the delivery path that is calculated at the client for use by the NCM to configure fragmentation and concatenation procedures [INTAREA-MAMS] at the N-MADP.

- Delivery Node ID: Identity of the node to which the client is attached. In the case of LTE, this is an ECGI. In the case of Wi-Fi, this is an AP ID or a MAC address.

At the beginning of a connection setup, the CCM informs the NCM of the connection status using the MX Reconfiguration Request with the Reconfiguration Action set to "setup". The NCM acknowledges the connection setup status and exchanges parameters with the CCM for user-plane setup, as described below.

Setup of User-Plane Protocols: Based on the negotiated capabilities, the NCM sets up the user-plane (Adaptation Layer and Convergence Layer) protocols at the N-MADP and informs the CCM of the user-plane protocols to be set up at the client (C-MADP) and the parameters for the C-MADP to connect to the N-MADP.

The MX UP Setup Configuration Request is used to create one or more MADP instances, with each anchor connection having one or more configurations, namely MX Configurations. The MX UP Setup Configuration Request consists of the following parameters:

- Number of Anchor Connections (core networks).

  For each anchor connection, the following parameters are included:

  - Anchor Connection ID
  - Connection Type (e.g., Wi-Fi, 5G NR, MulteFire, LTE)
  - Number of Active MX Configurations (included only if more than one MX configuration is active for the anchor connection).

    For each active MX configuration, the following parameters are included:

    - MX Configuration ID (included if more than one MX configuration is present)
    - MX Convergence Method. One of the following:

      - GMA
      - MPTCP Proxy
      - GRE Aggregation Proxy
      - MPQUIC

    - MX Convergence Method Parameters:

      - Convergence Proxy IP Address
      - Convergence Proxy Port
      - Client Key

    - MX Convergence Control Parameters (included if any MX Control PDU types (e.g., Probe-REQ/ACK) are supported):

      - UDP port number for sending and receiving MX Control PDUs (e.g., Probe-REQ/ACK, Keep-Alive)

        - Convergence Proxy Port

    - Number of Delivery Connections.

      For each delivery connection, include the following:

        - Delivery Connection ID
        - Connection Type (e.g., Wi-Fi, 5G NR, MulteFire, LTE)
        - MX Adaptation Method. One of the following:

            - UDP without DTLS
            - UDP with DTLS
            - IPsec
            - Client NAT

        - MX Adaptation Method Parameters:

            - Tunnel Endpoint IP Address
            - Tunnel Endpoint Port
            - Shared Secret
            - Header Optimization (included only if the MX Convergence Method is GMA)

For example, when LTE and Wi-Fi are the two user-plane accesses, the NCM conveys to the CCM that IPsec needs to be set up as the MX Adaptation Layer over the Wi-Fi access, using the following parameters: IPsec endpoint IP address, and Pre-Shared Key. No Adaptation Layer is needed if it is considered secure with no NAT, or a Client NAT may be used over the LTE access.

Similarly, as an example of the MX Convergence Method, the configuration indicates the convergence method as the MPTCP proxy, along with parameters for a connection to the MPTCP proxy: namely the IP address and port of the MPTCP proxy for TCP applications.

Once the user-plane protocols are configured, the CCM informs the NCM of the status via the MX UP Setup Confirmation. The MX UP Setup Confirmation consists of the following parameters:

- Unique Session ID: Session identifier provided to the client in an MX Capability Response.
- MX Convergence Control Parameters (included if any MX Control PDU types (e.g., Probe-REQ/ACK, Keep-Alive) are supported):

    - UDP port number for sending and receiving MX Control PDUs (e.g., Probe-REQ/ACK, Keep-Alive)
    - MX Configuration ID (if an MX Configuration ID is specified in an MX UP Setup Configuration Request) to indicate the MX Configuration that will be used for probing)

- Client Adaptation-Layer Parameters:

    - Number of Delivery Connections.

For each delivery connection, include the following:

- Delivery Connection ID
- UDP port number: If UDP-based adaptation is in use, the UDP port on the C-MADP side

8.6. MAMS Path Quality Estimation

Path quality estimations can be done either passively or actively. Traffic measurements in the network can be performed passively by comparing the real-time data throughput of the client with the capacity available in the network. In special deployments where the NCM has interfaces with access nodes, direct interfaces can be used to gather information regarding path quality. For example, the utilization of the LTE access node (also known as Evolved Node B), to which the client is attached, could be used as data for the estimation of path quality without creating any extra traffic overhead. Active measurements by the client provide an alternative way to estimate path quality.

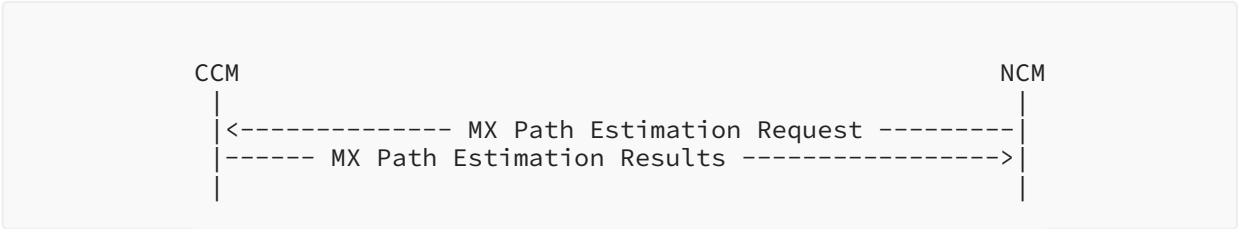

Figure 7: MAMS Control-Plane Procedure for Path Quality Estimation

The NCM sends the following configuration parameters in the MX Path Estimation Request to the CCM:

- Connection ID (of the delivery connection whose path quality needs to be estimated)
- Init Probe Test Duration (ms)
- Init Probe Test Rate (Mbps)
- Init Probe Size (bytes)
- Init Probe-ACK Required (0 -> No / 1 -> Yes)
- Active Probe Frequency (ms)
- Active Probe Size (bytes)
- Active Probe Test Duration (ms)
- Active Probe-ACK Required (0 -> No / 1 -> Yes)

The CCM configures the C-MADP for probe receipt based on these parameters and for collection of the statistics according to the following configuration.

- Unique Session ID: Session identifier provided to the client in an MX Capability Response.
- Init Probe Results Configuration:
  - Lost Probes (percent)
  - Probe Receiving Rate (packets per second)

- Active Probe Results Configuration:
  - Average Throughput in the last Probe Duration

The user-plane probing is divided into two phases: the Initialization phase and the Active phase.

- Initialization Phase: A network path that is not included by the N-MADP for transmission of user data is deemed to be in the Initialization phase. The user data may be transmitted over other available network paths.
- Active Phase: A network path that is included by the N-MADP for transmission of user data is deemed to be in the Active phase.

During the Initialization phase, the NCM configures the N-MADP to send an Init Probe-REQ message. The CCM collects the Init Probe statistics from the C-MADP and sends the MX Path Estimation Results message to the NCM per the Initialization Probe Results configuration.

During the Active phase, the NCM configures the N-MADP to send an Active Probe-REQ message. The C-MADP calculates the metrics as specified by the Active Probe Results configuration. The CCM collects the Active Probe statistics from the C-MADP and sends the MX Path Estimation Results message to the NCM per the Active Probe Results configuration.

The following subsections define the control PDU encoding for Keep-Alive and Probe-REQ/ACK messages to support path quality estimation.

8.6.1. MX Control PDU Definition

Control PDUs are sent as UDP messages between the C-MADP and the N-MADP to exchange control messages for keep-alive or path quality estimation. MX probe parameters are negotiated during the user-plane setup phase (MX UP Setup Configuration Request and MX UP Setup Confirmation). Figure 8 shows the MX Control PDU format with the following fields:

- Type (1 byte): The type of the MX Control message.
  - 0: Keep-Alive
  - 1: Probe-REQ/ACK
  - Others: Reserved
- CID (1 byte): The connection ID of the delivery connection for sending the MX Control message.
- MX Control Message (variable): The payload of the MX Control message.

The MX Control PDU is sent as a normal user-plane packet over the desired delivery connection whose quality and reachability need to be determined.

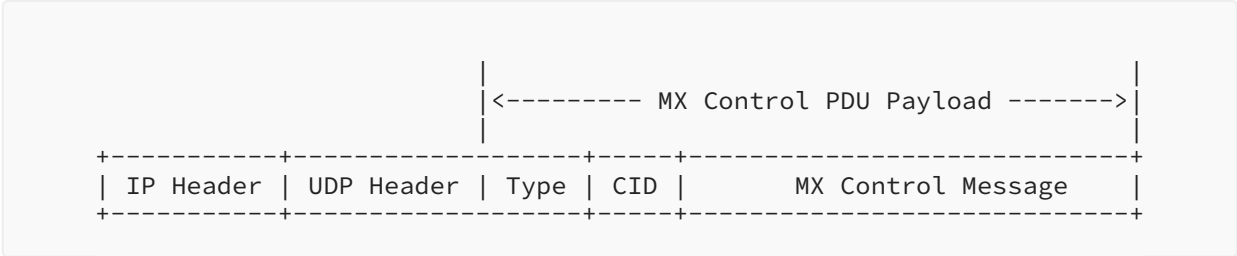

Figure 8: MX Control PDU Format

8.6.2. Keep-Alive Message

The "Type" field is set to "0" for Keep-Alive messages. The C-MADP may periodically send a Keep-Alive message over one or multiple delivery connections, especially if UDP tunneling is used as the adaptation method for the delivery connection with a NAT function on the path.

A Keep-Alive message is 2 bytes long and consists of the following field:

- Keep-Alive Sequence Number (2 bytes): The sequence number of the Keep-Alive message.

8.6.3. Probe-REQ/ACK Message

The "Type" field is set to "1" for Probe-REQ/ACK messages. The N-MADP may send the Probe-REQ message for path quality estimation. In response, the C-MADP may return the Probe-ACK message.

A Probe-REQ message consists of the following fields:

- Probing Sequence Number (2 bytes): The sequence number of the Probe REQ message.
- Probing Flag (1 byte):
  - Bit 0: A Probe-ACK flag to indicate whether the Probe-ACK message is expected (1) or not (0).
  - Bit 1: A Probe Type flag to indicate whether the Probe-REQ/ACK message was sent during the Initialization phase (0) when the network path is not included for transmission of user data, or during the Active phase (1) when the network path is included for transmission of user data.
  - Bit 2: A bit flag to indicate the presence of the Reverse Connection ID (R-CID) field.
  - Bits 3-7: Reserved.
- Reverse Connection ID (R-CID) (1 byte): The connection ID of the delivery connection for sending the Probe-ACK message on the reverse path.
- Padding (variable).

The "R-CID" field is only present if both Bit 0 and Bit 2 of the "Probing Flag" field are set to "1". Moreover, Bit 2 of the "Probing Flag" field **SHOULD** be set to "0" if Bit 0 is "0", indicating that the Probe-ACK message is not expected.

If the "R-CID" field is not present, but Bit 0 of the "Probing Flag" field is set to "1", the Probe-ACK message **SHOULD** be sent over the same delivery connection as the Probe-REQ message.

The "Padding" field is used to control the length of the Probe-REQ message.

The C-MADP **SHOULD** send the Probe-ACK message in response to a Probe-REQ message with the Probe-ACK flag set to "1".

A Probe-ACK message is 3 bytes long and consists of the following field:

- Probing Acknowledgment Number (2 bytes): The sequence number of the corresponding Probe-REQ message.

8.7. MAMS Traffic Steering

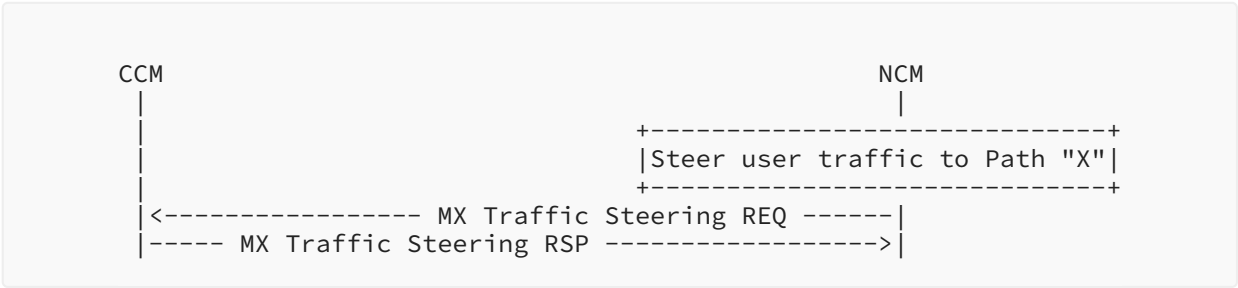

Figure 9: MAMS Traffic-Steering Procedure

The NCM sends an MX Traffic Steering Request to steer data traffic. It is also possible to send data traffic over multiple connections simultaneously, i.e., aggregation. The message includes the following information:

- Anchor Connection ID: Connection ID of the anchor connection.
- MX Configuration ID (if an MX Configuration ID is specified in an MX UP Setup Configuration Request).
- DL Connection ID List: List of DL delivery connections, provided as Connection IDs.
- UL Connection ID: Connection ID of the default UL delivery connection.
- For the number of specific UL traffic templates, the message includes the following:
  - Traffic Flow Template for identifying the UL traffic.
  - UL Connection ID List: List of UL delivery connections, provided as Connection IDs, to be used for sending the UL traffic.
- MX Feature Activation List. Each parameter indicates whether the corresponding feature is enabled or not: lossless switching, fragmentation, concatenation, uplink aggregation, downlink aggregation, measurement, probing.

In response, the CCM sends an MX Traffic Steering Response, including the following information:

- Unique Session ID: Session identifier provided to the client in an MX Capability Response.
- MX Feature Activation List. Each parameter indicates whether the corresponding feature is enabled or not: lossless switching, fragmentation, concatenation, uplink aggregation, downlink aggregation, measurement, probing.

8.8. MAMS Application MADP Association

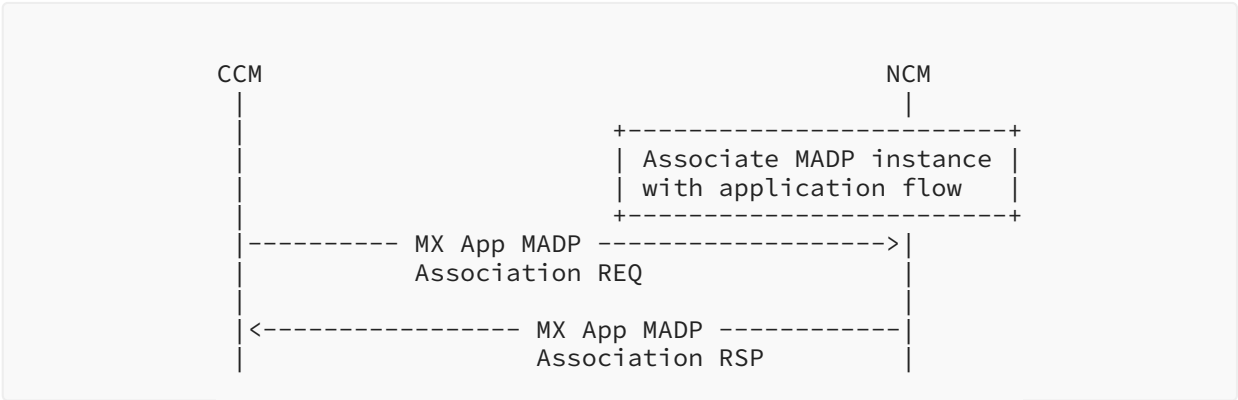

Figure 10: MAMS Application MADP Association Procedure

The CCM sends an MX Application MADP Association Request to request the association of a specific application flow with a specific MADP instance ID for the anchor connection with multiple active MX configurations. The MADP Instance ID is a tuple (Anchor Connection ID, MX Configuration ID). This provides the capability for the client to indicate the user-plane processing that needs to be associated with different application flows depending on the needs of those flows. The application flow is identified by its associated Traffic Flow Template.

The MX Application MADP Association Request includes the following information:

- Number of Application Flows.

  For each application flow, identified by the Traffic Flow Templates:

  - Anchor Connection ID
  - MX Configuration ID (if more than one MX configuration is associated with an anchor connection)
  - Traffic Flow Template for identifying the UL traffic
  - Traffic Flow Template for identifying the DL traffic

In response, the NCM sends an MX Application MADP Association Response, including the following information:

- Number of Application Flows.

For each application flow, identified by the Traffic Flow Templates:

- Status (Success or Failure)

8.9. MAMS Network ID Indication

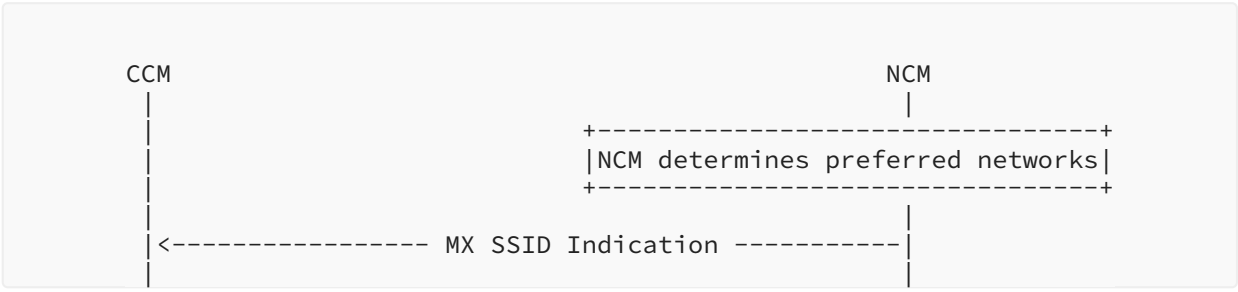

Figure 11: MAMS Network ID Indication Procedure

The NCM indicates the preferred network list to the CCM to guide the client regarding networks that it should connect to. To indicate preferred Wi-Fi networks, the NCM sends the list of WLANs, each represented by an SSID (Service Set Identifier)/BSSID (Basic Service Set Identifier)/HESSID (Homogeneous Extended Service Set Identifier) as defined in [IEEE-80211]), available in the MX SSID Indication.

8.10. MAMS Client Measurement Configuration and Reporting

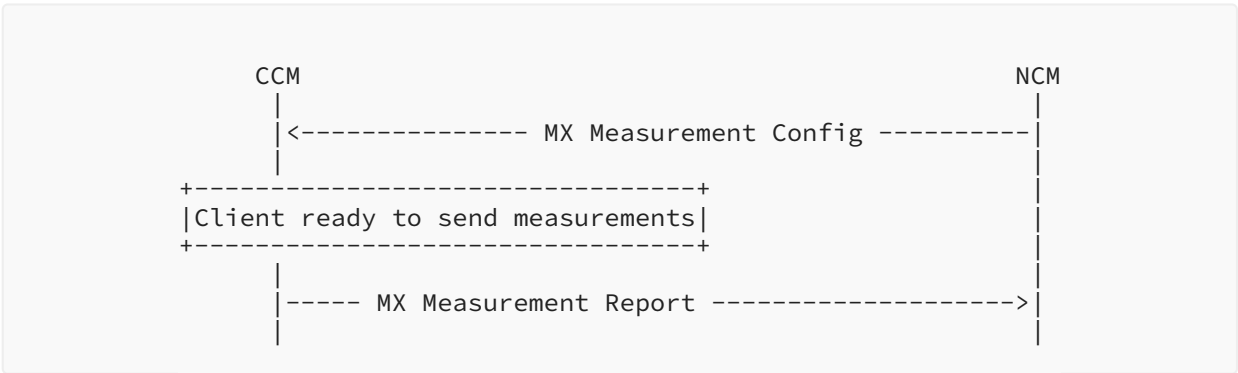

Figure 12: MAMS Client Measurement Configuration and Reporting Procedure

The NCM configures the CCM with the different parameters (e.g., radio link information), with the associated thresholds to be reported by the client. The MX Measurement Configuration message contains the following parameters for each delivery connection:

- Delivery Connection ID.
- Connection Type (e.g., Wi-Fi, 5G NR, MulteFire, LTE).
- If the connection type is Wi-Fi:
  - High and low thresholds for the sending of average Received Signal Strength Indicator (RSSI) of the Wi-Fi link.

- Periodicity, in ms, for sending the average RSSI of the Wi-Fi link.
- High and low thresholds for sending the loading of the WLAN system.
- Periodicity, in ms, for sending the loading of the WLAN system.
- High and low thresholds for sending the reverse link throughput on the Wi-Fi link.
- Periodicity, in ms, for sending the reverse link throughput on the Wi-Fi link.
- High and low thresholds for sending the forward link throughput on the Wi-Fi link.
- Periodicity, in ms, for sending the forward link throughput on the Wi-Fi link.
- High and low thresholds for sending the reverse link throughput (EstimatedThroughputOutbound as defined in [IEEE-80211]) on the Wi-Fi link.
- Periodicity, in ms, for sending the reverse link throughput (EstimatedThroughputOutbound as defined in [IEEE-80211]) on the Wi-Fi link.
- High and low thresholds for sending the forward link throughput (EstimatedThroughputInbound, as defined in [IEEE-80211]) on the Wi-Fi link.
- Periodicity, in ms, for sending the forward link throughput (EstimatedThroughputInbound, as defined in [IEEE-80211]) on the Wi-Fi link.

• If the connection type is LTE:

- High and low thresholds for sending the Reference Signal Received Power (RSRP) of the serving LTE link.
- Periodicity, in ms, for sending the RSRP of the serving LTE link.
- High and low thresholds for sending the RSRQ (Reference Signal Received Quality) of the serving LTE link.
- Periodicity, in ms, for sending the RSRP of the serving LTE link.
- High and low thresholds for sending the reverse link throughput on the serving LTE link.
- Periodicity, in ms, for sending the reverse link throughput on the serving LTE link.
- High and low thresholds, for sending the forward link throughput on the serving LTE link.
- Periodicity, in ms, for sending the forward link throughput on the serving LTE link.

• If the connection type is 5G NR:

- High and low thresholds for sending the RSRP of the serving NR link.
- Periodicity, in ms, for sending the RSRP of the serving NR link.
- High and low thresholds for sending the RSRQ of the serving NR link.
- Periodicity, in ms, for sending the RSRP of the serving NR link.
- High and low thresholds for sending the reverse link throughput on the serving NR link.
- Periodicity, in ms, for sending the reverse link throughput on the serving NR link.
- High and low thresholds for sending the forward link throughput on the serving NR link.
- Periodicity, in ms, for sending the forward link throughput on the serving NR link.

The MX Measurement Report contains the following parameters:

• Unique Session ID: Session identifier provided to the client in an MX Capability Response.

- For each delivery connection, include the following:
  - Delivery Connection ID
  - Connection Type (e.g., Wi-Fi, 5G NR, MulteFire, LTE)
  - Delivery Node ID (ECGI in the case of LTE. In the case of Wi-Fi, this is an AP ID or a MAC address.)
  - If the connection type is Wi-Fi:
    - Average RSSI of the Wi-Fi link.
    - Loading of the WLAN system.
    - Reverse link throughput on the Wi-Fi link.
    - Forward link throughput on the Wi-Fi link.
    - Estimated reverse link throughput on the Wi-Fi link (EstimatedThroughputOutbound as defined in [IEEE-80211]).
    - Estimated forward link throughput on the Wi-Fi link (EstimatedThroughputInbound, as defined in [IEEE-80211]).
  - If the connection type is LTE:
    - RSRP of the serving LTE link.
    - RSRQ of the serving LTE link.
    - Reverse link throughput on the serving LTE link.
    - Forward link throughput on the serving LTE link.
  - If the connection type is 5G NR:
    - RSRP of the serving NR link.
    - RSRQ of the serving NR link.
    - Reverse link throughput on the serving NR link.
    - Forward link throughput on the serving NR link.

8.11. MAMS Session Termination Procedure

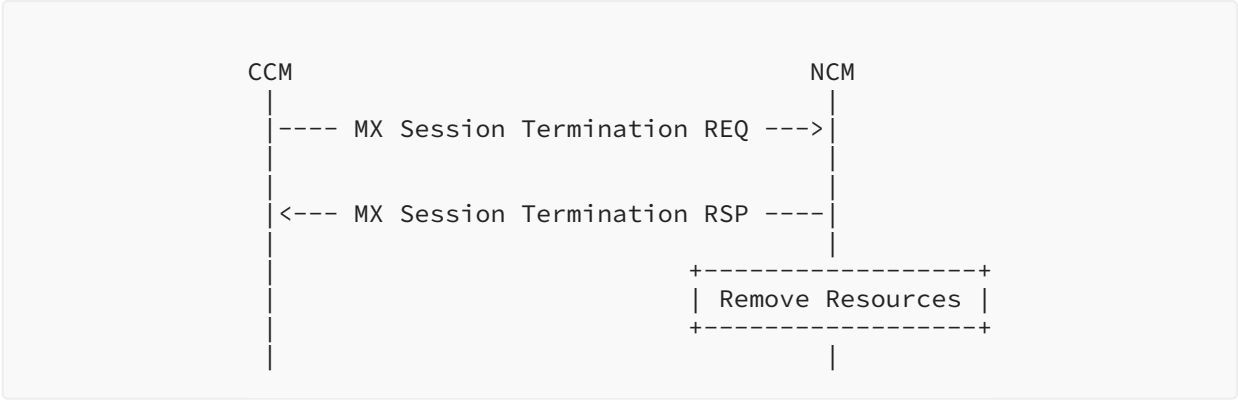

Figure 13: MAMS Session Termination Procedure - Initiated by Client

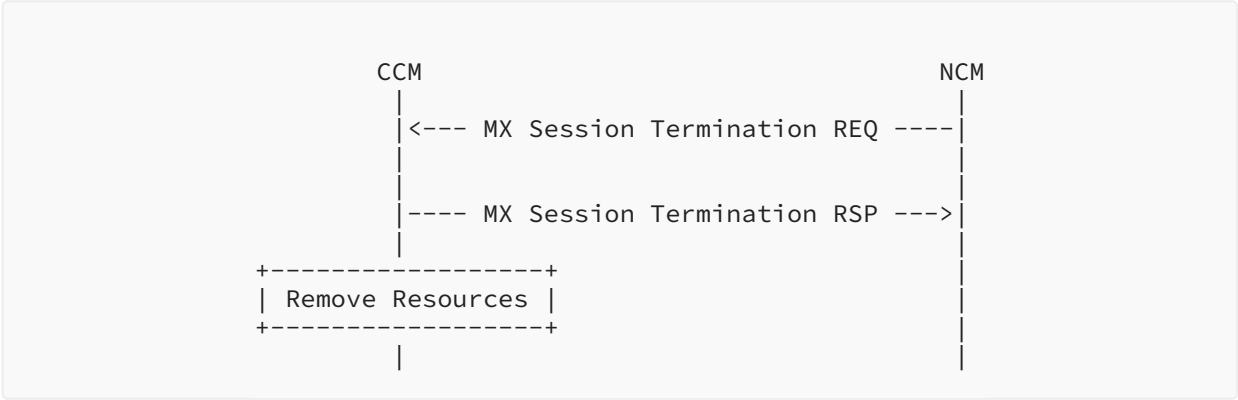

Figure 14: MAMS Session Termination Procedure - Initiated by Network

At any point in MAMS processing, if the CCM or NCM is no longer able to support the MAMS functions, then either of them can initiate a termination procedure by sending an MX Session Termination Request to the peer. The peer **SHALL** acknowledge the termination by sending an MX Session Termination Response message. After the session is disconnected, the CCM **SHALL** start a new procedure with an MX Discover message. An MX Session Termination Request shall contain a Unique Session ID and the reason for the termination. Possible reasons for termination are:

- Normal Release
- No Response from Peer
- Internal Error

8.12. MAMS Network Analytics Request Procedure

Figure 15: MAMS Network Analytics Request Procedure

The CCM sends the MX Network Analytics Request to the NCM to request information related to such network parameters as bandwidth, latency, jitter, and signal quality, based on the application of analytics at the network (utilizing the received path measurements and client measurement reporting).

The MX Network Analytics Request consists of the following parameters:

- Link Quality Indicators. One or more of the following:
  - Bandwidth
  - Jitter
  - Latency
  - Signal Quality

The NCM sends the MX Network Analytics Response to convey analytics information that might be of interest to the CCM. This message will include network parameters with their predicted likelihoods.

The MX Network Analytics Response consists of the following parameters:

- Number of Delivery Connections.

  For each delivery connection, include the following:

  - Access Link Identifier:
    - Connection Type
    - Connection ID
  - Link Quality Indicator:
    - Bandwidth:
      - Predicted Value (Mbps)
      - Likelihood (percent)
      - Prediction Validity (Validity Time, in seconds)

- Jitter:
  - Predicted Value (in seconds)
  - Likelihood (percent)
  - Prediction Validity (Validity Time, in seconds)
- Latency:
  - Predicted Value (in seconds)
  - Likelihood (percent)
  - Prediction Validity (Validity Time, in seconds)
- Signal Quality:
  - If delivery connection type is LTE, LTE_RSRP Predicted Value in decibel-milliwatts (dBm)
  - If delivery connection type is LTE, LTE_RSRQ Predicted Value (dBm)
  - If delivery connection type is 5G NR, NR_RSRP Predicted Value (dBm)
  - If delivery connection type is 5G NR, NR_RSRQ Predicted Value (dBm)
  - If delivery connection type is Wi-Fi, WLAN_RSSI Predicted Value (dBm)
  - Likelihood (percent)
  - Prediction Validity (Validity Time, in seconds)

9. Generic MAMS Signaling Flow

Figure 16 illustrates the MAMS signaling mechanism for negotiation of network paths and flow protocols between the client and the network. In this example scenario, the client is connected to two networks (LTE and Wi-Fi).

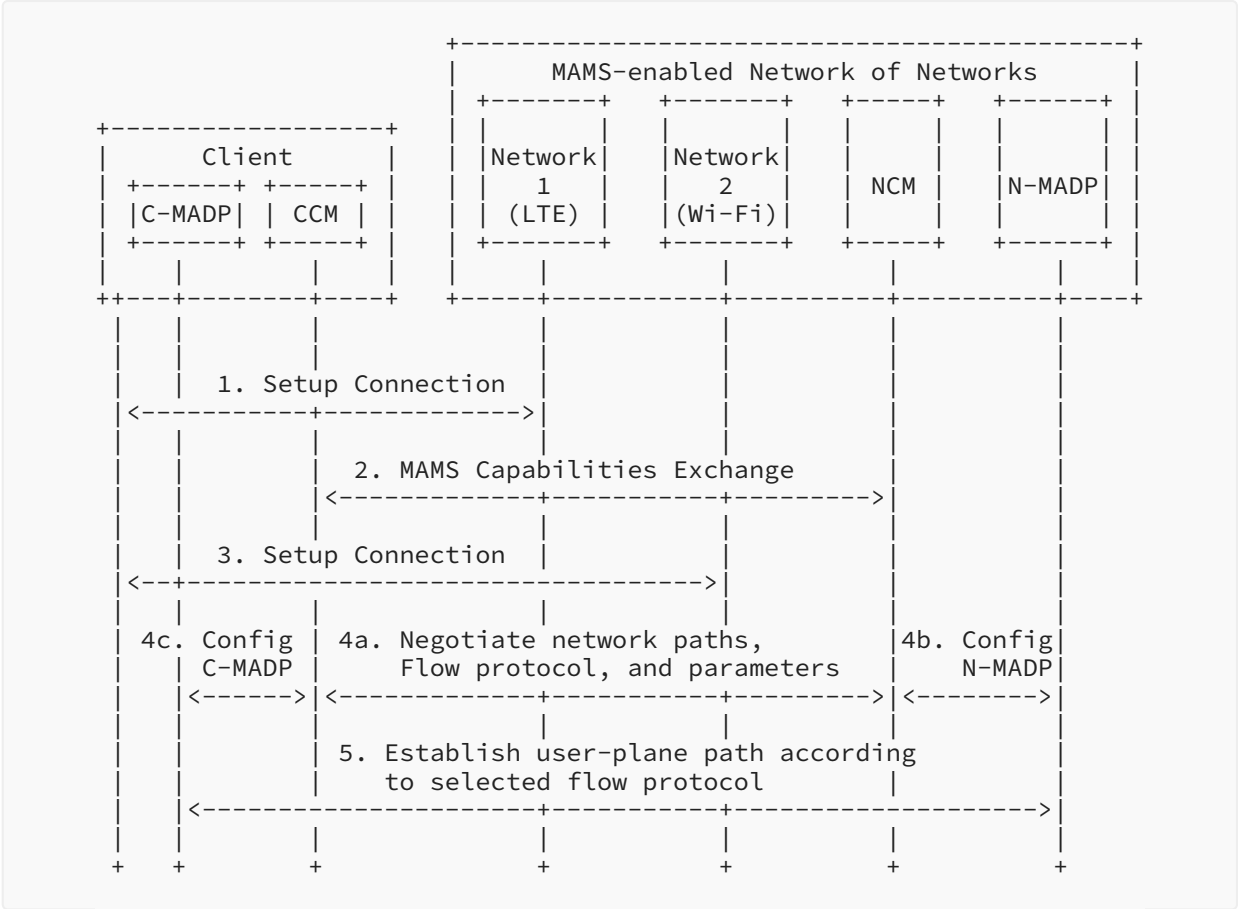

Figure 16: MAMS Call Flow

1. The client connects to Network 1 and gets an IP address assigned by Network 1.
2. The CCM communicates with the NCM functional element via the Network 1 connection and exchanges capabilities and parameters for MAMS operation. Note: The NCM credentials (e.g., the NCM's IP address) can be made known to the client by provisioning.
3. The client sets up the connection with Network 2 and gets an IP address assigned by Network 2.
4. The CCM and NCM negotiate capabilities and parameters for establishing network paths. The negotiated capabilities and parameters are then used to configure user-plane functions, i.e., the N-MADP at the network and the C-MADP at the client.

   4a. The CCM and NCM negotiate network paths, flow routing and aggregation protocols, and related parameters.

   4b. The NCM communicates with the N-MADP to exchange and configure flow aggregation protocols, policies, and parameters in alignment with those negotiated with the CCM.

   4c. The CCM communicates with the C-MADP to exchange and configure flow aggregation protocols, policies, and parameters in alignment with those negotiated with the NCM.

5. The C-MADP and N-MADP establish the user-plane paths, e.g., using Internet Key Exchange Protocol (IKE) [RFC7296] signaling, based on the negotiated flow aggregation protocols and parameters specified by the NCM.

The CCM and NCM can further exchange messages containing access link measurements for link maintenance by the NCM. The NCM evaluates the link conditions in the UL and DL across LTE and Wi-Fi, based on link measurements reported by the CCM and/or link-probing techniques, and determines the policy for UL and DL user data distribution. The NCM and CCM also negotiate application-level policies for categorizing applications, e.g., based on the Differentiated Services Code Point (DSCP), destination IP address, and determination of which available network path needs to be used for transporting data of that category of applications. The NCM configures the N-MADP, and the CCM configures the C-MADP, based on the negotiated application policies. The CCM may apply local application policies, in addition to the application policy conveyed by the NCM.

10. Relationship to IETF Technologies

The MAMS framework leverages technologies developed in the IETF (such as MPTCP and GRE) and enables a control-plane framework to negotiate the use of these protocols between the client and the network. It also addresses the limitations in scope of other multihoming protocols. For example, the IKEv2 Mobility and Multihoming Protocol (MOBIKE [RFC4555]) scope indicates that it is limited to multihoming between IPsec clients (tunnel mode IPsec Security Associations) and does not support load balancing. To address this limitation regarding how the multihoming scenario is handled, the MAMS framework supports load balancing with the simultaneous use of multiple access paths by negotiating the use of protocols like MPTCP. Unlike MOBIKE, which only applies to endpoints connected with an IPsec tunnel mode Security Association, the MAMS framework allows the flexibility to use a wide range of tunneling protocols in the Adaptation Layer.

11. Applying MAMS Control Procedures with MPTCP Proxy as User Plane

If the NCM determines that the N-MADP is to be instantiated with MPTCP as the MX Convergence Protocol, it exchanges the MPTCP capability support in the discovery and capability exchange procedures. An MPTCP proxy (e.g., see [TCPM-CONVERTERS]) is configured to be the N-MADP instance. The NCM then provides the credentials of the MPTCP Proxy instance, along with related parameters, to the CCM. The CCM configures the C-MADP with these parameters to connect to this MPTCP proxy instance.

Figure 17 illustrates the user-plane protocol layering when MPTCP is configured to be the "MX Convergence Layer" protocol. MPTCP manages traffic distribution and aggregation over multiple delivery connections.

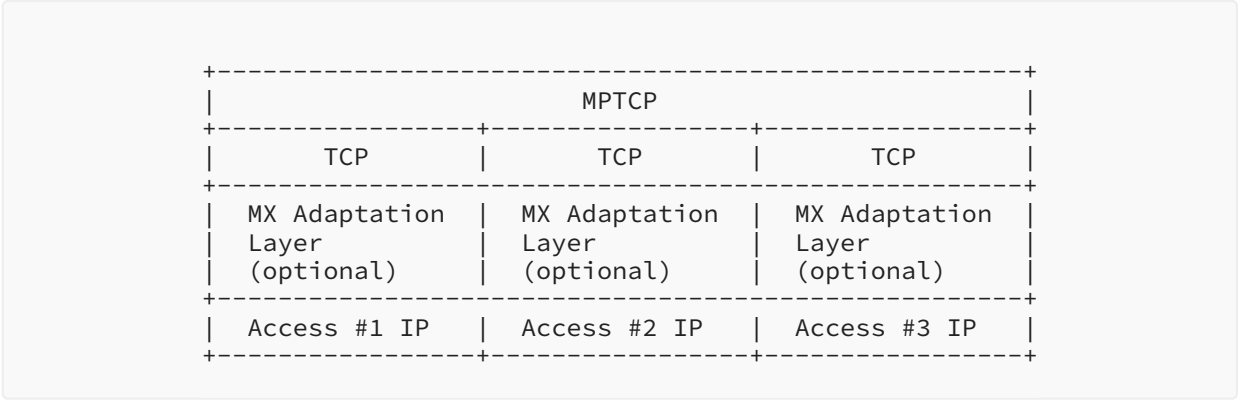

Figure 17: MAMS User-Plane Protocol Stack with MPTCP as MX Convergence Layer

The client (C-MADP) sets up an MPTCP connection with the N-MADP to begin with. The MAMS control procedures are then applied to do the following:

- Connect to the appropriate MPTCP network endpoint, e.g., the MPTCP proxy (illustrated in Figure 18).
- Control the addition of a second TCP subflow after the Wi-Fi connection is established and is deemed good (illustrated in Figure 19).
- Control the behavior of the MPTCP scheduler, e.g., by using only the LTE subflow in the UL and both the LTE and Wi-Fi subflows in the DL (illustrated in Figure 20).
- Provide faster response to Wi-Fi link degradation by proactively deleting a TCP subflow over Wi-Fi when poor link conditions are reported, maintaining optimum performance (illustrated in Figure 21).

Figure 18 shows the call flow describing MAMS control procedures applied to configure the user plane and dynamic optimal path selection in a scenario with the MPTCP proxy as the convergence protocol in the user plane.

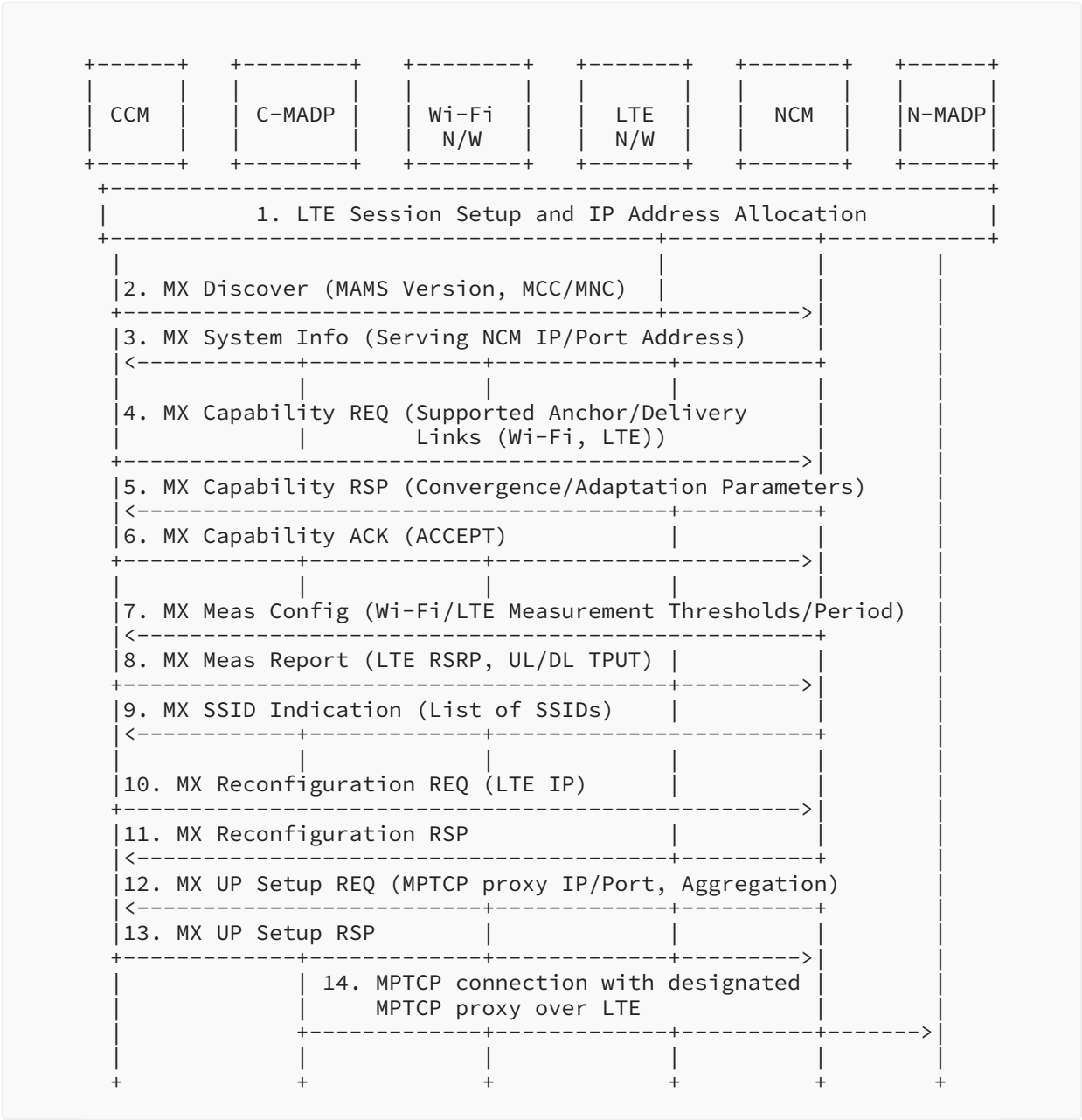

Figure 18: MAMS-Assisted MPTCP Proxy as User Plane - Initial Setup with LTE Leg

The salient steps described in the call flow are as follows. The client connects to the LTE network and obtains an IP address (assume that LTE is the first connection). It then initiates the NCM discovery procedures and exchanges capabilities, including the support for MPTCP as the convergence protocol at both the network and the client.

The CCM provides the LTE connection parameters to the NCM. The NCM provides the parameters like MPTCP proxy IP address/port, and MPTCP Client Key for configuring the Convergence Layer. This is useful if the N-MADP is reachable, via a different IP address or/and port, from different access networks. The current MPTCP signaling can't identify or differentiate the MPTCP proxy IP

address and port from multiple access networks. The client uses the MPTCP Client Key during the subflow creation, and this enables the N-MADP to uniquely identify the client, even if a NAT is present. The N-MADP can then inform the NCM of the subflow creation and parameters related to creating additional subflows. Since LTE is the only connection, the user-plane traffic flows over the single TCP subflow over the LTE connection. Optionally, the NCM provides assistance information to the client on the neighboring/preferred Wi-Fi networks that it can associate with.

Figure 19 describes the steps where the client establishes a Wi-Fi connection. The CCM informs the NCM of the Wi-Fi connection, along with such parameters as the Wi-Fi IP address or the SSID. The NCM determines that the Wi-Fi connection needs to be secured, configures the Adaptation Layer to use IPsec, and provides the required parameters to the CCM. In addition, the NCM provides the information for configuring the Convergence Layer (e.g., MPTCP proxy IP address) and provides the MX Traffic Steering Request to indicate that the client **SHOULD** use only the LTE access. The NCM may do this, for example, on determining from the measurements that the Wi-Fi link is not consistently good enough. As the Wi-Fi link conditions improve, the NCM sends an MX Traffic Steering Request to use Wi-Fi access as well. This triggers the client to establish the TCP subflow over the Wi-Fi link with the MPTCP proxy.

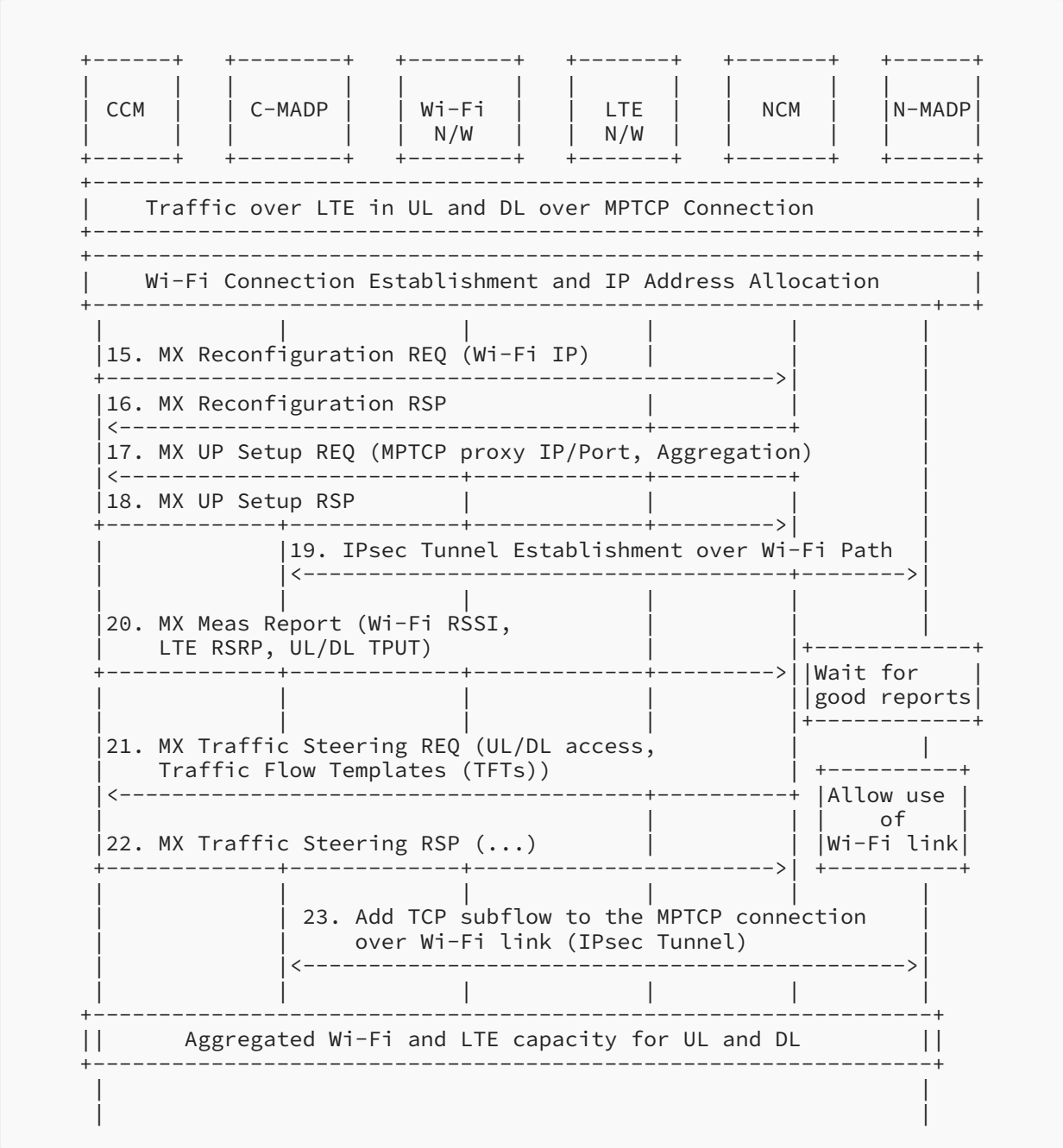

Figure 19: MAMS-Assisted MPTCP Proxy as User Plane - Add Wi-Fi Leg

Figure 20 describes the steps where the client reports that Wi-Fi link conditions degrade in UL. The MAMS control plane is used to continuously monitor the access link conditions on Wi-Fi and LTE connections. The NCM may at some point determine an increase in UL traffic on the Wi-Fi network, and trigger the client to use only LTE in the UL via a MX Traffic Steering Request to improve UL performance.

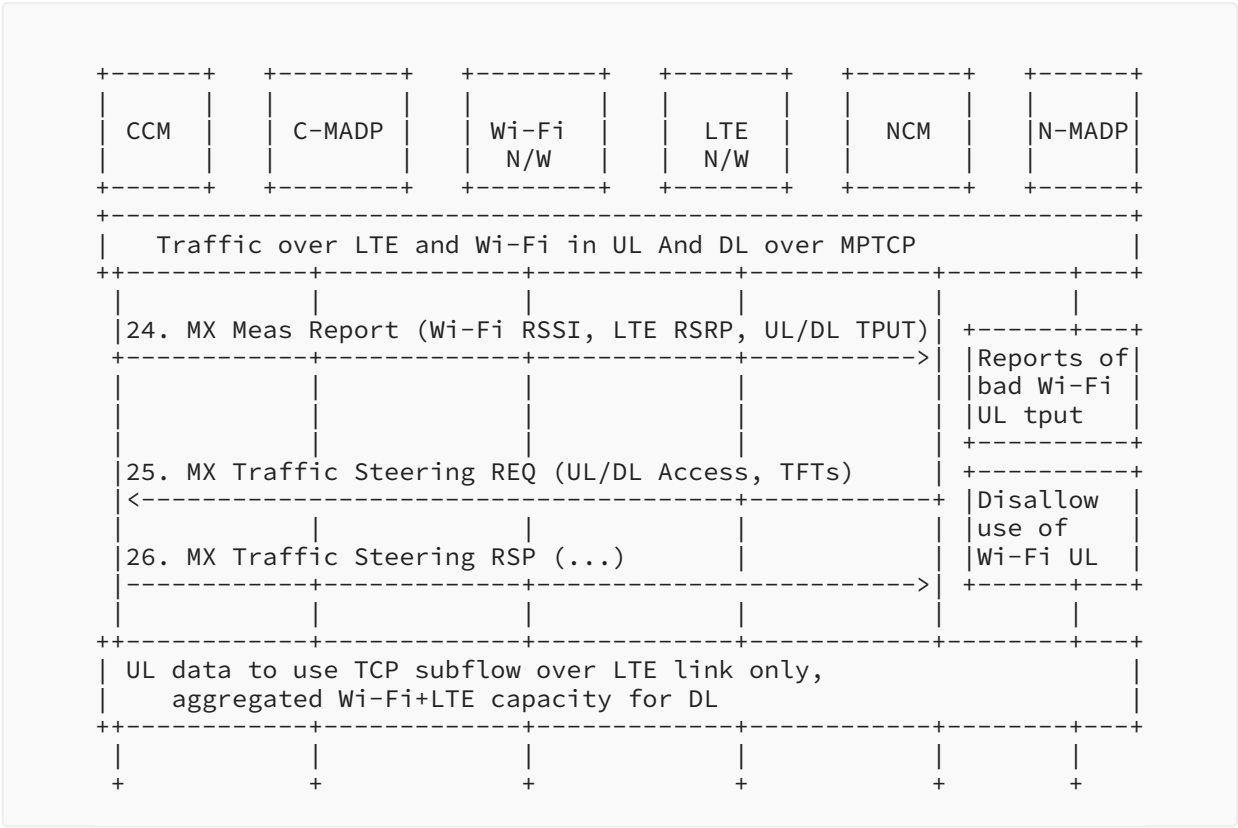

Figure 20: MAMS-Assisted MPTCP Proxy as User Plane - Wi-Fi UL Degrades

Figure 21 describes the steps where the client reports that Wi-Fi link conditions have degraded in both the UL and DL. As the Wi-Fi link conditions deteriorate further, the NCM may decide to send a MX Traffic Steering Request that instructs the client to stop using Wi-Fi and to use only the LTE access in both the UL and DL. This condition may be maintained until the NCM determines, based on reported measurements, that the Wi-Fi link has again become usable.

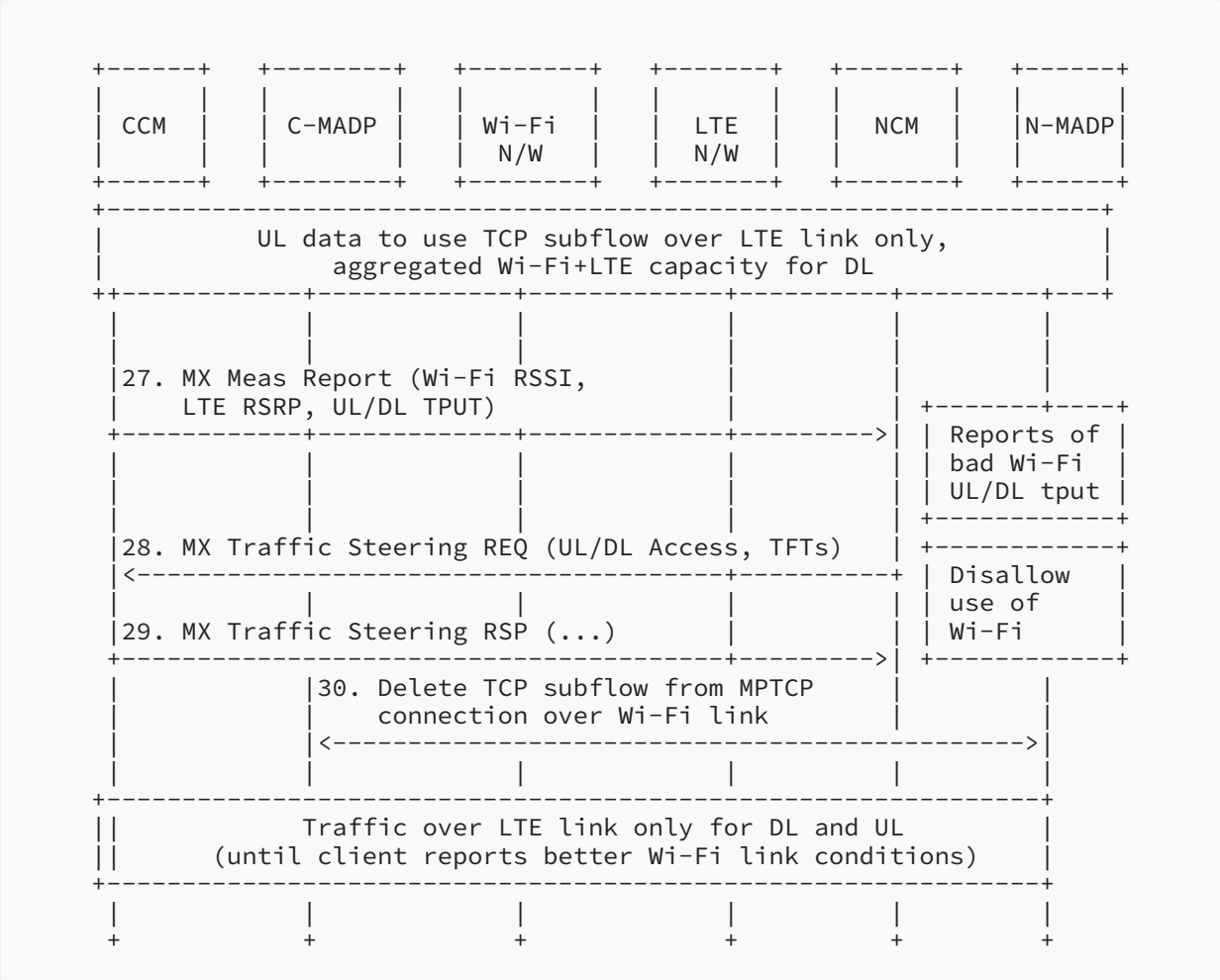

Figure 21: MAMS-Assisted MPTCP Proxy as User Plane - Part 4

12. Applying MAMS Control Procedures for Network-Assisted Traffic Steering When There Is No Convergence Layer

Figure 22 shows the call flow describing MAMS control procedures applied for dynamic optimal path selection in a scenario where Convergence and Adaptation Layer protocols are omitted. This scenario indicates the applicability of a solution for only the MAMS control plane.

In the capability exchange messages, the NCM and CCM negotiate that Convergence-Layer and Adaptation-Layer protocols are not needed (or supported). The CCM informs the NCM of the availability of the LTE and Wi-Fi links. The NCM dynamically determines the access links (Wi-Fi or LTE) to be used based on the reported measurements of link quality.

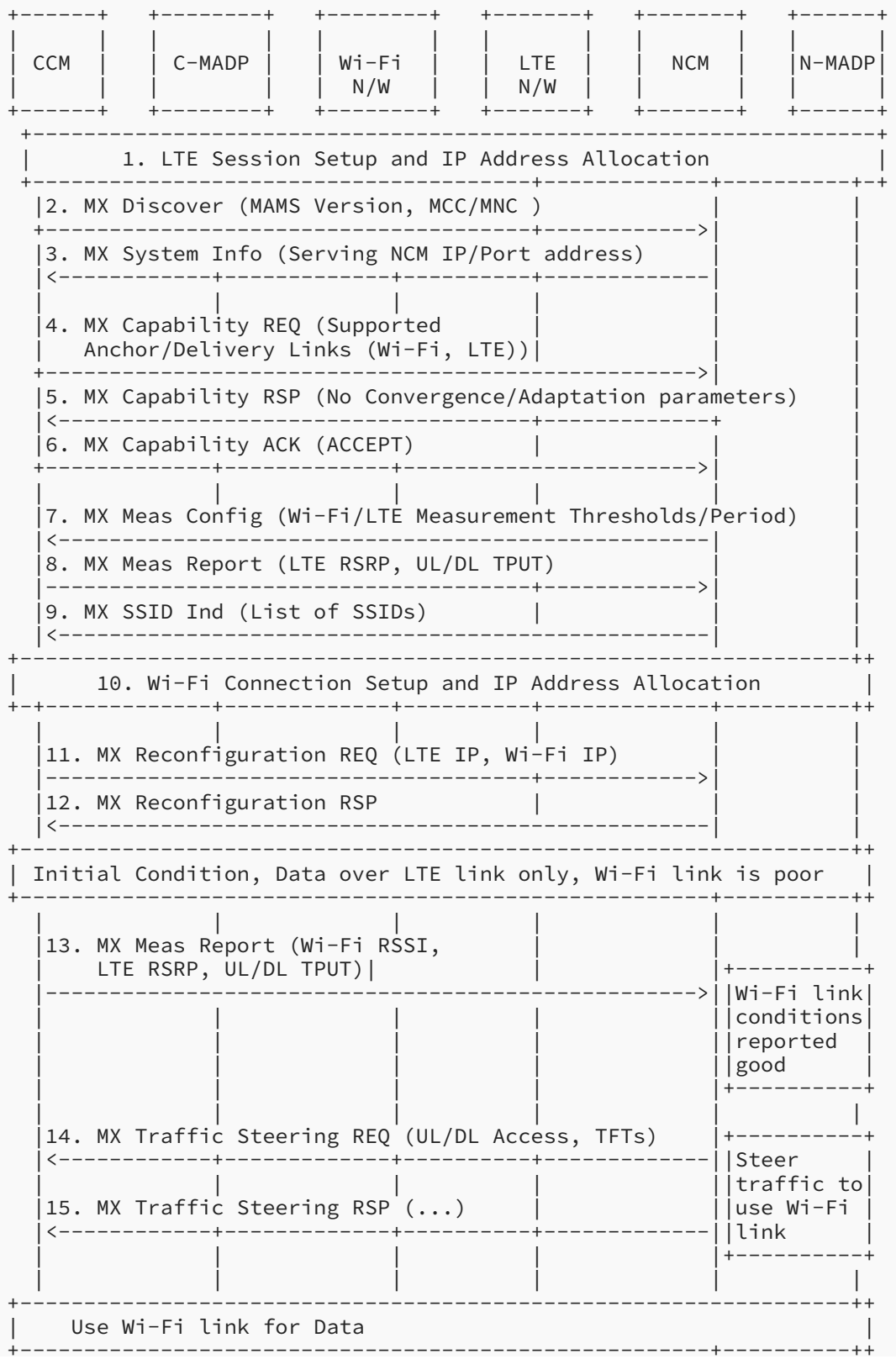
CCM
C-MADP
Wi-Fi N/W
LTE N/W
NCM
N-MADP
1. LTE Session Setup and IP Address Allocation
2. MX Discover (MAMS Version, MCC/MNC)
3. MX System Info (Serving NCM IP/Port address)
4. MX Capability REQ (Supported Anchor/Delivery Links (Wi-Fi, LTE))
5. MX Capability RSP (No Convergence/Adaptation parameters)
6. MX Capability ACK (ACCEPT)
7. MX Meas Config (Wi-Fi/LTE Measurement Thresholds/Period)
8. MX Meas Report (LTE RSRP, UL/DL TPUT)
9. MX SSID Ind (List of SSIDs)
10. Wi-Fi Connection Setup and IP Address Allocation
11. MX Reconfiguration REQ (LTE IP, Wi-Fi IP)
12. MX Reconfiguration RSP
Initial Condition, Data over LTE link only, Wi-Fi link is poor
13. MX Meas Report (Wi-Fi RSSI, LTE RSRP, UL/DL TPUT)
Wi-Fi link conditions reported good
14. MX Traffic Steering REQ (UL/DL Access, TFTs)
Steer traffic to use Wi-Fi link
15. MX Traffic Steering RSP (...)
Use Wi-Fi link for Data

Figure 22: MAMS with No Convergence Layer

13. Coexistence of MX Adaptation and MX Convergence Layers

The MAMS user plane supports multiple instances and combinations of protocols to be used at the MX Adaptation and the Convergence Layer.

For example, one instance of the MX Convergence Layer can be MPTCP Proxy and another instance can be GMA. The MX Adaptation for each can be either a UDP tunnel or IPsec. IPsec may be set up when the network path needs to be secured, e.g., to protect the TCP subflow traversing the network path between the client and the MPTCP proxy.

Each instance of the MAMS user plane, i.e., the combination of MX Convergence-Layer and MX Adaptation-Layer protocols, can coexist simultaneously and independently handle different traffic types.

14. Security Considerations

14.1. MAMS Control-Plane Security

The NCM functional element is hosted on a network node that is assumed to be within a secure network, e.g., within the operator's network, and is assumed to be protected against hijack attacks.

For deployment scenarios where the client is configured (e.g., by the network operator) to use a specific network path for exchanging control-plane messages, and if the network path is assumed to be secure, MAMS control messages will rely on security provided by the underlying network.

For deployment scenarios where the security of the network path cannot be assumed, NCM and CCM implementations **MUST** support the "wss" URI scheme [RFC6455] and Transport Layer Security (TLS) [RFC8446] to secure the exchange of control-plane messages between the NCM and the CCM.

For deployment scenarios where client authentication is desired, the WebSocket server can use any client authentication mechanisms available to a generic HTTP server, such as cookies, HTTP authentication, or TLS authentication.

14.2. MAMS User-Plane Security

User data in the MAMS framework relies on the security of the underlying network transport paths. When this security cannot be assumed, the NCM configures the use of protocols (e.g., IPsec [RFC4301] [RFC3948]) in the MX Adaptation Layer, for security.

15. Implementation Considerations

The MAMS architecture builds on commonly available functions in clients that can be used to deliver software updates over popular client operating systems, thereby enabling rapid deployment and addressing the large base of deployed clients.

16. Applicability to Multi-Access Edge Computing

Multi-access Edge Computing (MEC), previously known as Mobile Edge Computing, is an access-edge cloud platform being considered at the European Telecommunications Standards Institute (ETSI) [ETSIMEC], whose initial focus was to improve the QoE by leveraging intelligence at the cellular (e.g., 3GPP technologies like LTE) access edge, and the scope is now being extended to support access technologies beyond 3GPP. The applicability of the framework described in this document to the MEC platform has been evaluated and tested in different network configurations by the authors.

The NCM can be hosted on a MEC cloud server that is located in the user-plane path at the edge of the multi-technology access network. The NCM and CCM can negotiate the network path combinations based on an application's needs and the necessary user-plane protocols to be used across the multiple paths. The network conditions reported by the CCM to the NCM can be complemented by a Radio Analytics application [ETSIRNIS] residing at the MEC cloud server to configure the uplink and downlink access paths according to changing radio and congestion conditions.

The user-plane functional element, N-MADP, can either be collocated with the NCM at the MEC cloud server (e.g., MEC-hosted applications) or placed at a separate network element like a common user-plane gateway across the multiple networks.

Also, even in scenarios where an N-MADP is not deployed, the NCM can be used to augment the traffic-steering decisions at the client.

The aim of these enhancements is to improve the end user's QoE by leveraging the best network path based on an application's needs and network conditions, and building on the advantages of significantly reduced latency and the dynamic and real-time exposure of radio network information available at the MEC.

17. Related Work in Other Industry and Standards Forums

The MAMS framework described in this document has been incorporated or is proposed for incorporation as a solution to address multi-access integration in multiple industry forums and standards. This section describes the related work in other industry forums and the standards organizations.

Wireless Broadband Alliance industry partners have published a white paper that describes the applicability of different technologies for multi-access integration to different deployments as part of their "Unlicensed Integration with 5G Networks" project [WBAUnl5G]. The white paper includes the MAMS framework described in this document as a technology for integrating unlicensed (Wi-Fi) networks with 5G networks above the 5G core network.

The 3GPP is developing a technical report as part of its work item Study on Access Traffic Steering, Switching, and Splitting (ATSSS). That report, TR 23.793 [ATSSS], contains a number of potential solutions; Solution 1 in [ATSSS] utilizes a separate control plane for the flexible negotiation of user-plane protocols and path measurements in a way that is similar to the MAMS architecture described in this document.

The Small Cell Forum (SCF) [SCFTECH5G] plans to develop a white paper as part of its work item on LTE/5G and Wi-Fi. There is a proposal to include MAMS in this white paper.

The ETSI Multi-access Edge Computing Phase 2 technical work is examining many aspects of this work, including use cases for optimizing QoE and resource utilization. The MAMS architecture and procedures outlined in this document are included in the ETSI's use cases and requirements document [ETSIMAMS].

18. IANA Considerations

This document has no IANA actions.

19. References

19.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <https://www.rfc-editor.org/info/rfc4301>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

19.2. Informative References

[ANDSF] 3rd Generation Partnership Project, "Access Network Discovery and Selection Function (ANDSF) Management Object (MO)", 3GPP TS 24.312 version 15.0.0, Technical Specification Group Core Network and Terminals, June 2018, <https://www.3gpp.org/ftp//Specs/archive/24_series/24.312/24312-f00.zip>.

[ATSSS] 3rd Generation Partnership Project, "Study on access traffic steering, switch and splitting support in the 5G System (5GS) architecture", Work in Progress, 3GPP TR 23.793 v16.0.0, December 2018, <https://www.3gpp.org/ftp/Specs/archive/23_series/23.793/>.

[ETSIMAMS] European Telecommunications Standards Institute, "Multi-access Edge Computing (MEC); Phase 2: Use Cases and Requirements", ETSI GS MEC 002 v2.1.1, October 2018, <https://www.etsi.org/deliver/etsi_gs/MEC/001_099/002/02.01.01_60/gs_MEC002v020101p.pdf>.

[ETSIMEC] European Telecommunications Standards Institute, "Multi-access Edge Computing (MEC)", <https://www.etsi.org/technologies/multi-access-edge-computing>.

[ETSIRNIS] European Telecommunications Standards Institute, "Mobile Edge Computing (MEC) Radio Network Information API", ETSI GS MEC 012 v1.1.1, July 2017, <https://www.etsi.org/deliver/etsi_gs/MEC/001_099/012/01.01.01_60/gs_MEC012v010101p.pdf>.

[IEEE-80211] IEEE, "IEEE Standard for Information technology-Telecommunications and information exchange between systems - Local and metropolitan area networks-Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications", IEEE 802.11-2016, <https://ieeexplore.ieee.org/document/7786995>.

[INTAREA-GMA] Zhu, J. and S. Kanugovi, "Generic Multi-Access (GMA) Convergence Encapsulation Protocols", Work in Progress, Internet-Draft, draft-zhu-intarea-gma-05, 16 December 2019, <https://tools.ietf.org/html/draft-zhu-intarea-gma-05>.

[INTAREA-MAMS] Zhu, J., Seo, S., Kanugovi, S., and S. Peng, "User-Plane Protocols for Multiple Access Management Service", Work in Progress, Internet-Draft, draft-zhu-intarea-mams-user-protocol-09, 4 March 2020, <https://tools.ietf.org/html/draft-zhu-intarea-mams-user-protocol-09>.

[ITU-E212] International Telecommunication Union, "The international identification plan for public networks and subscriptions", ITU-T Recommendation E.212, September 2016, <https://www.itu.int/rec/T-REC-E.212-201609-I/en>.

[QUIC-MULTIPATH] Coninck, Q. and O. Bonaventure, "Multipath Extensions for QUIC (MP-QUIC)", Work in Progress, Internet-Draft, draft-deconinck-quic-multipath-04, 5 March 2020, <https://tools.ietf.org/html/draft-deconinck-quic-multipath-04>.

[RFC2784] Farinacci, D., Li, T., Hanks, S., Meyer, D., and P. Traina, "Generic Routing Encapsulation (GRE)", RFC 2784, DOI 10.17487/RFC2784, March 2000, <https://www.rfc-editor.org/info/rfc2784>.

[RFC2890] Dommety, G., "Key and Sequence Number Extensions to GRE", RFC 2890, DOI 10.17487/RFC2890, September 2000, <https://www.rfc-editor.org/info/rfc2890>.

[RFC3948] Huttunen, A., Swander, B., Volpe, V., DiBurro, L., and M. Stenberg, "UDP Encapsulation of IPsec ESP Packets", RFC 3948, DOI 10.17487/RFC3948, January 2005, <https://www.rfc-editor.org/info/rfc3948>.

[RFC4555] Eronen, P., "IKEv2 Mobility and Multihoming Protocol (MOBIKE)", RFC 4555, DOI 10.17487/RFC4555, June 2006, <https://www.rfc-editor.org/info/rfc4555>.

[RFC4960] Stewart, R., Ed., "Stream Control Transmission Protocol", RFC 4960, DOI 10.17487/RFC4960, September 2007, <https://www.rfc-editor.org/info/rfc4960>.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <https://www.rfc-editor.org/info/rfc6347>.

[RFC6455] Fette, I. and A. Melnikov, "The WebSocket Protocol", RFC 6455, DOI 10.17487/RFC6455, December 2011, <https://www.rfc-editor.org/info/rfc6455>.

[RFC6824] Ford, A., Raiciu, C., Handley, M., and O. Bonaventure, "TCP Extensions for Multipath Operation with Multiple Addresses", RFC 6824, DOI 10.17487/RFC6824, January 2013, <https://www.rfc-editor.org/info/rfc6824>.

[RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, DOI 10.17487/RFC7231, June 2014, <https://www.rfc-editor.org/info/rfc7231>.

[RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <https://www.rfc-editor.org/info/rfc7296>.

[RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <https://www.rfc-editor.org/info/rfc8259>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[SCFTECH5G] Small Cell Forum, "Small Cell Forum", <https://scf.io/>.

[ServDesc3GPP] 3rd Generation Partnership Project, "General Packet Radio Service (GPRS); Service description; Stage 2", 3GPP TS 23.060 version 16.0.0, Technical Specification Group Services and System Aspects, March 2019, <https://www.3gpp.org/ftp/Specs/archive/23_series/23.060/23060-g00.zip>.

[TCPM-CONVERTERS] Bonaventure, O., Boucadair, M., Gundavelli, S., Seo, S., and B. Hesmans, "0-RTT TCP Convert Protocol", Work in Progress, Internet-Draft, draft-ietf-tcpm-converters-19, 22 March 2020, <https://tools.ietf.org/html/draft-ietf-tcpm-converters-19>.

[WBAUnl5G] Wireless Broadband Alliance, "Unlicensed Integration with 5G Networks", <https://wballiance.com/resource/unlicensed-integration-with-5g-networks/>.

Appendix A. MAMS Control-Plane Optimization over Secure Connections

This appendix is informative, and provides indicative information about how MAMS operates.

If the connection between the CCM and the NCM over which the MAMS control-plane messages are transported is assumed to be secure, UDP is used as the transport for management and control messages between the NCM and the CCM (see Figure 23).

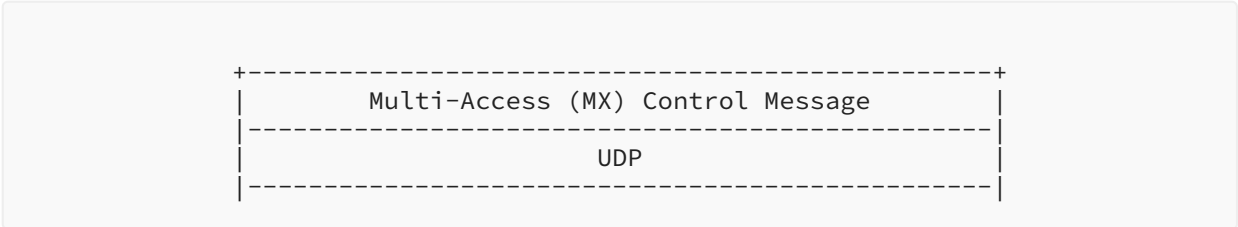

Figure 23: UDP-Based MAMS Control-Plane Protocol Stack

Appendix B. MAMS Application Interface

This appendix describes the MAMS Application Interface. It does not provide normative text for the definition of the MAMS framework or protocols, but offers additional information that may be used to construct a system based on the MAMS framework.

B.1. Overall Design

The CCM hosts an HTTPS server for applications to communicate and request services. This document assumes, from a security point of view, that all CCMs and the communicating application instances are hosted in a single administrative domain.

The content of messages is described in JavaScript Object Notation (JSON) format. They offer RESTful APIs for communication.

The exact mechanism regarding how the application knows about the endpoint of the CCM is out of scope for this document. This mechanism may instead be provided as part of the application settings.

B.2. Notation

The documentation of APIs is provided in the OpenAPI format, using Swagger v2.0. See Appendix D.

B.3. Error Indication

For every API, there could be an error response if the objective of the API could not be met; see [RFC7231].

B.4. CCM APIs

The following subsections describe the APIs exposed by the CCM to the applications.

B.4.1. GET Capabilities

The CCM provides an HTTPS GET interface as "/ccm/v1.0/capabilities" for the application to query the capabilities supported by the CCM instance.

Figure 24: CCM API - GET Procedures

The CCM shall provide information regarding its capabilities as follows:

- Supported Features: One or more of the "Feature Name" values, as defined in the MX Feature Activation List parameter of the MX Capability Request (Appendix C.2.5).
- Supported Connections: Supported connection types and connection IDs.
- Supported MX Adaptation Layers: List of MX Adaptation Layer protocols supported by the N-MADP instance, along with the connection types where these are supported and their respective parameters.
- Supported MX Convergence Layers: List of supported MX Convergence Layer protocols, along with the parameters associated with the respective convergence technique.

B.4.2. Posting Application Requirements

The CCM provides an HTTPS POST interface as "/ccm/v1.0/app_requirements" for the application to post the needs of the application data streams to the CCM instance.

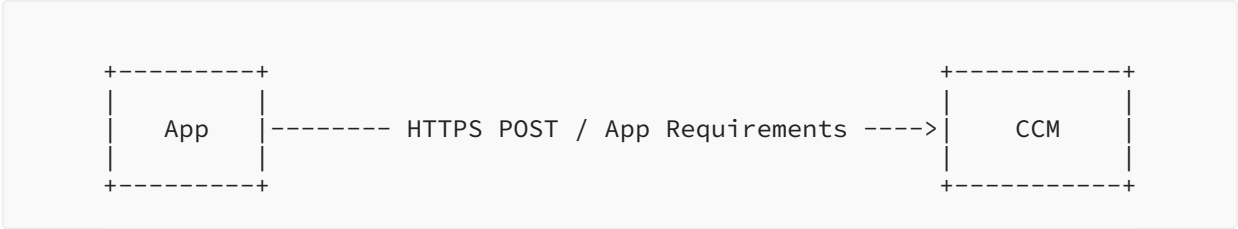

Figure 25: CCM API - POST Procedures

The CCM shall provide for the application to post the following requirements for its different data streams:

- Number of Data Stream Types.
- For each data stream type, specify the following parameters for the link, which are preferred by the application:
  - Protocol Type: Transport-layer protocol associated with the application data stream packets.
  - Port Range: Supported connection types and connection IDs.
  - Traffic QoS: Quality of service parameters, as follows:
    - Bandwidth
    - Latency
    - Jitter

B.4.3. Getting Predictive Link Parameters

The CCM provides an HTTPS GET interface as "/ccm/v1.0/predictive_link_params" for the application to get the predicted link parameters from the CCM instance.

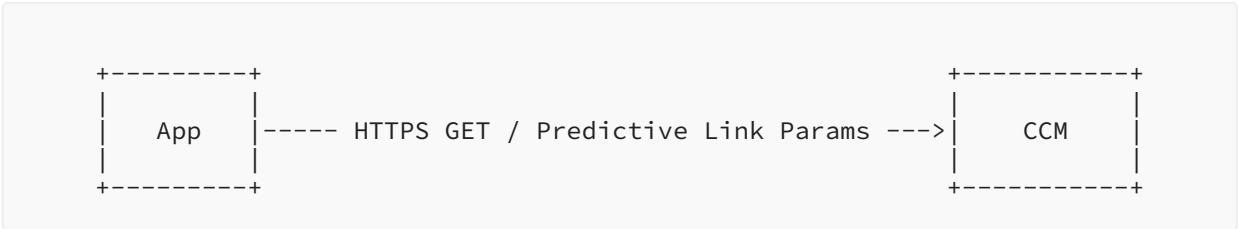

Figure 26: CCM API - Getting Predictive Link Parameters

The CCM asks the NCM for link parameters via the MAMS Network Analytics Request Procedure (Section 8.12) and includes the information in response to the API invocation.

- Number of Delivery Connections.

For each delivery connection, include the following:

- Access Link Identifier:
  - Connection Type
  - Connection ID
- Link Quality Indicator
  - Bandwidth:
    - Predicted Value (Mbps)
    - Likelihood (percent)
    - Prediction Validity (Validity Time, in seconds)
  - Jitter:
    - Predicted Value (in seconds)
    - Likelihood (percent)
    - Prediction Validity (Validity Time, in seconds)
  - Latency:
    - Predicted Value (in seconds)
    - Likelihood (percent)
    - Prediction Validity (Validity Time, in seconds)
  - Signal Quality
    - If delivery connection type is LTE, LTE_RSRP Predicted Value (dBm)
    - If delivery connection type is LTE, LTE_RSRQ Predicted Value (dBm)
    - If delivery connection type is 5G NR, NR_RSRP Predicted Value (dBm)
    - If delivery connection type is 5G NR, NR_RSRQ Predicted Value (dBm)
    - If delivery connection type is Wi-Fi, WLAN_RSSI Predicted Value (dBm)
    - Likelihood (percent)
    - Prediction Validity (Validity Time, in seconds)

Appendix C. MAMS Control-Plane Messages Described Using JSON

MAMS control-plane messages are exchanged between the CCM and the NCM. This non-normative appendix describes the format and content of messages using JSON [RFC8259].

C.1. Protocol Specification: General Processing

C.1.1. Notation

This document uses JSONString, JSONNumber, and JSONBool to indicate the JSON string, number, and boolean types, respectively.

This document uses an adaptation of the C-style struct notation to describe JSON objects. A JSON object consists of name/value pairs. This document refers to each pair as a field. In some contexts, this document also refers to a field as an attribute. The name of a field/attribute may be referred to as the key. An optional field is enclosed by "[ ]". In the definitions, the JSON names of the fields are case sensitive. An array is indicated by two numbers in angle brackets, <m..n>, where m indicates the minimal number of values and n is the maximum. When this document uses * for n, it means no upper bound.

For example, the text below describes a new type Type4, with three fields: "name1", "name2", and "name3", respectively. The "name3" field is optional, and the "name2" field is an array of at least one value.

```
object { Type1 name1; Type2 name2 <1..*>; [Type3 name3;] } Type4;
```

This document uses subtyping to denote that one type is derived from another type. The example below denotes that TypeDerived is derived from TypeBase. TypeDerived includes all fields defined in TypeBase. If TypeBase does not have a "name1" field, TypeDerived will have a new field called "name1". If TypeBase already has a field called "name1" but with a different type, TypeDerived will have a field called "name1" with the type defined in TypeDerived (i.e., Type1 in the example).

```
object { Type1 name1; } TypeDerived : TypeBase;
```

Note that, despite the notation, no standard, machine-readable interface definition or schema is provided in this document. Extension documents may describe these as necessary.

For compatibility with publishing requirements, line breaks have been inserted inside long JSON strings, with the following continuation lines indented. To form the valid JSON example, any line breaks inside a string must be replaced with a space and any other white space after the line break removed.

C.1.2. Discovery Procedure

C.1.2.1. MX Discover Message

This message is the first message sent by the CCM to discover the presence of NCM in the network. It contains only the base information as described in Appendix C.2.1 with message_type set as mx_discover.

The representation of the message is as follows:

```
object {
    [JSONString MCC_MNC_Tuple;]
 } MXDiscover : MXBase;
```

C.1.3. System Information Procedure

C.1.3.1. MX System Info Message

This message is sent by the NCM to the CCM to inform the endpoints that the NCM supports MAMS functionality. In addition to the base information (Appendix C.2.1), it contains the following information:

(a) NCM Connections (Appendix C.2.3).

The representation of the message is as follows:

```
object {
   NCMConnections ncm_connections;
} MXSystemInfo : MXBase;
```

C.1.4. Capability Exchange Procedure

C.1.4.1. MX Capability Request

This message is sent by the CCM to the NCM to indicate the capabilities of the CCM instance available to the NCM indicated in the System Info message earlier. In addition to the base information (Appendix C.2.1), it contains the following information:

(a) Features and their activation status: See Appendix C.2.5.

(b) Number of Anchor Connections: The number of anchor connections (toward the core) supported by the NCM.

(c) Anchor connections: See Appendix C.2.6.

(d) Number of Delivery Connections: The number of delivery connections (toward the access) supported by the NCM.

(e) Delivery connections: See Appendix C.2.7.

(f) Convergence methods: See Appendix C.2.9.

(g) Adaptation methods: See Appendix C.2.10.

The representation of the message is as follows:

```
object {
   FeaturesActive feature_active;
   JSONNumber num_anchor_connections;
   AnchorConnections anchor_connections;
   JSONNumber num_delivery_connections;
   DeliveryConnections delivery_connections;
   ConvergenceMethods convergence_methods;
   AdaptationMethods adaptation_methods
} MXCapabilityReq : MXBase;
```

C.1.4.2. MX Capability Response

This message is sent by the NCM to the CCM to indicate the capabilities of the NCM instance and unique session identifier for the CCM. In addition to the base information (Appendix C.2.1), it contains the following information:

(a) Features and their activation status: See Appendix C.2.5.

(b) Number of Anchor Connections: The number of anchor connections (toward the core) supported by the NCM.

(c) Anchor connections: See Appendix C.2.6.

(d) Number of Delivery Connections: The number of delivery connections (toward the access) supported by the NCM.

(e) Delivery connections: See Appendix C.2.7.

(f) Convergence methods: See Appendix C.2.9.

(g) Adaptation methods: See Appendix C.2.10.

(h) Unique Session ID: This uniquely identifies the session between the CCM and the NCM in a network. See Appendix C.2.2.

The representation of the message is as follows:

```
object {
   FeaturesActive feature_active;
   JSONNumber num_anchor_connections;
   AnchorConnections anchor_connections;
   JSONNumber num_delivery_connections;
   DeliveryConnections delivery_connections;
   ConvergenceMethods convergence_methods;
   AdaptationMethods adaptation_methods
   UniqueSessionId unique_session_id;
} MXCapabilityRsp : MXBase;
```

C.1.4.3. MX Capability Acknowledge

This message is sent by the CCM to the NCM to indicate acceptance of capabilities advertised by the NCM in an earlier MX Capability Response message. In addition to the base information (Appendix C.2.1), it contains the following information:

(a) Unique Session ID: Same identifier as the identifier provided in the MX Capability Response. See Appendix C.2.2.

(b) Capability Acknowledgment: Indicates either acceptance or rejection of the capabilities sent by the CCM. Can use either "MX_ACCEPT" or "MX_REJECT" as acceptable values.

The representation of the message is as follows:

```
object {
   UniqueSessionId unique_session_id;
   JSONString capability_ack;
} MXCapabilityAck : MXBase;
```

C.1.5. User-Plane Configuration Procedure

C.1.5.1. MX User-Plane Configuration Request

This message is sent by the NCM to the CCM to configure the user plane for MAMS. In addition to the base information (Appendix C.2.1), it contains the following information:

(a) Number of Anchor Connections: The number of anchor connections supported by the NCM.

(b) Setup of anchor connections: See Appendix C.2.11.

The representation of the message is as follows:

```
object {
   JSONNumber num_anchor_connections;
   SetupAnchorConns anchor_connections;
} MXUPSetupConfigReq : MXBase;
```

C.1.5.2. MX User-Plane Configuration Confirmation

This message is the confirmation of the user-plane setup message sent from the CCM after successfully configuring the user plane on the client. This message contains the following information:

(a) Unique Session ID: Same identifier as the identifier provided in the MX Capability Response. See Appendix C.2.2.

(b) MX probe parameters (included if probing is supported).

   (1) Probe Port: UDP port for accepting probe message.

(2) Anchor connection ID: Identifier of the anchor connection to be used for probe function. Provided in the MX UP Setup Configuration Request.

(3) MX Configuration ID: This parameter is included only if the MX Configuration ID parameter is available from the user-plane setup configuration. It indicates the MX configuration ID of the anchor connection to be used for probe function.

(c) The following information is required for each delivery connection:

(1) Connection ID: Delivery connection ID supported by the client.

(2) Client Adaptation-Layer Parameters: If the UDP Adaptation Layer is in use, then the UDP port to be used on the C-MADP side.

The representation of the message is as follows:

```
object {
   UniqueSessionId unique_session_id;
   [ProbeParam probe_param;]
   JSONNumber num_delivery_conn;
   ClientParam client_params <1...*>;
} MXUPSetupConfigCnf : MXBase;
```

Where ProbeParam is defined as follows:

```
object {
   JSONNumber probe_port;
   JSONNumber anchor_conn_id;
   [JSONNumber mx_configuration_id;]
} ProbeParam;
```

Where ClientParam is defined as follows:

```
object {
   JSONNumber connection_id;
   [AdaptationParam adapt_param;]
} ClientParam;
```

Where AdaptationParam is defined as follows:

```
object {
   JSONNumber udp_adapt_port;
} AdaptationParam;
```

C.1.6. Reconfiguration Procedure

C.1.6.1. MX Reconfiguration Request

This message is sent by the CCM to the NCM in the case of reconfiguration of any of the connections from the client's side. In addition to the base information (Appendix C.2.1), it contains the following information:

(a) Unique Session ID: Identifier for the CCM-NCM association Appendix C.2.2.

(b) Reconfiguration Action: The reconfiguration action type can be one of "setup", "release", or "update".

(c) Connection ID: Connection ID for which the reconfiguration is taking place.

(d) IP address: Included if Reconfiguration Action is either "setup" or "update".

(e) SSID: If the connection type is Wi-Fi, then this parameter contains the SSID to which the client has attached.

(f) MTU of the connection: The MTU of the delivery path that is calculated at the client for use by the NCM to configure fragmentation and concatenation procedures at the N-MADP.

(g) Connection Status: This parameter indicates whether the connection is currently "disabled", "enabled", or "connected". Default: "connected".

(h) Delivery Node ID: Identity of the node to which the client is attached. In the case of LTE, this is an ECGI. In the case of Wi-Fi, this is an AP ID or a MAC address.

The representation of the message is as follows:

```
object {
   UniqueSessionId unique_session_id;
   JSONString reconf_action;
   JSONNumber connection_id;
   JSONString ip_address;
   JSONString ssid;
   JSONNumber mtu_size;
   JSONString connection_status;
   [JSONString delivery_node_id;]
} MXReconfReq : MXBase;
```

C.1.6.2. MX Reconfiguration Response

This message is sent by the NCM to the CCM as a confirmation of the received MX Reconfiguration Request and contains only the base information (as defined in Appendix C.2.1).

The representation of the message is as follows:

```
object {
} MXReconfRsp : MXBase;
```

C.1.7. Path Estimation Procedure

C.1.7.1. MX Path Estimation Request

This message is sent by the NCM toward the CCM to configure the CCM to send MX Path Estimation Results. In addition to the base information (Appendix C.2.1), it contains the following information:

(a) Connection ID: ID of the connection for which the path estimation report is required.
(b) Init Probe Test Duration: Duration of initial probe test, in milliseconds.
(c) Init Probe Test Rate: Initial testing rate, in megabits per second.
(d) Init Probe Size: Size of each packet for initial probe, in bytes.
(e) Init Probe-ACK: If an acknowledgment for probe is required. (Possible values: "yes", "no")
(f) Active Probe Frequency: Frequency, in milliseconds, at which the active probes shall be sent.
(g) Active Probe Size: Size of the active probe, in bytes.
(h) Active Probe Duration: Duration, in seconds, for which the active probe shall be performed.
(i) Active Probe-ACK: If an acknowledgment for probe is required. (Possible values: "yes", "no")

The representation of the message is as follows:

```
object {
   JSONNumber connection_id;
   JSONNumber init_probe_test_duration_ms;
   JSONNumber init_probe_test_rate_Mbps;
   JSONNumber init_probe_size_bytes;
   JSONString init_probe_ack_req;
   JSONNumber active_probe_freq_ms;
   JSONNumber active_probe_size_bytes;
   JSONNumber active_probe_duration_sec;
   JSONString active_probe_ack_req;
} MXPathEstReq : MXBase;
```

C.1.7.2. MX Path Estimation Results

This message is sent by the CCM to the NCM to report on the probe estimation configured by the NCM. In addition to the base information (Appendix C.2.1), it contains the following information:

(a) Unique Session ID: Same identifier as the identifier provided in the MX Capability Response. See Appendix C.2.2.
(b) Connection ID: ID of the connection for which the MX Path Estimation Results message is required.
(c) Init Probe Results: See Appendix C.2.12.
(d) Active Probe Results: See Appendix C.2.13.

The representation of the message is as follows:

```
object {
   JSONNumber connection_id;
   UniqueSessionId unique_session_id;
   [InitProbeResults init_probe_results;]
   [ActiveProbeResults active_probe_results;]
} MXPathEstResults : MXBase;
```

C.1.8. Traffic-Steering Procedure

C.1.8.1. MX Traffic Steering Request

This message is sent by the NCM to the CCM to enable traffic steering on the delivery side in uplink and downlink configurations. In addition to the base information (Appendix C.2.1), it contains the following information:

(a) Connection ID: Anchor connection number for which the traffic steering is being defined.

(b) MX Configuration ID: MX configuration for which the traffic steering is being defined.

(c) Downlink Delivery: See Appendix C.2.14.

(d) Default UL Delivery: The default delivery connection for the uplink. All traffic should be delivered on this connection in the uplink direction, and the Traffic Flow Template (TFT) filter should be applied only for the traffic mentioned in Uplink Delivery.

(e) Uplink Delivery: See Appendix C.2.15.

(f) Features and their activation status: See Appendix C.2.5.

The representation of the message is as follows:

```
object {
   JSONNumber connection_id;
   [JSONNumber mx_configuration_id;]
   DLDelivery downlink_delivery;
   JSONNumber default_uplink_delivery;
   ULDelivery uplink_delivery;
   FeaturesActive feature_active;
} MXTrafficSteeringReq : MXBase;
```

C.1.8.2. MX Traffic Steering Response

This message is a response to an MX Traffic Steering Request from the CCM to the NCM. In addition to the base information (Appendix C.2.1), it contains the following information:

(a) Unique Session ID: Same identifier as the identifier provided in the MX Capability Response. See Appendix C.2.2.

(b) Features and their activation status: See Appendix C.2.5.

The representation of the message is as follows:

```
object {
   UniqueSessionId unique_session_id;
   FeaturesActive feature_active;
} MXTrafficSteeringResp : MXBase;
```

C.1.9. MAMS Application MADP Association

C.1.9.1. MX Application MADP Association Request

This message is sent by the CCM to the NCM to select MADP instances provided earlier in the MX UP Setup Configuration Request, based on requirements for the applications.

In addition to the base information (Appendix C.2.1), it contains the following:

(a) Unique Session ID: This uniquely identifies the session between the CCM and the NCM in a network. See Appendix C.2.2.

(b) A list of MX Application MADP Associations, with each entry as follows:

(1) Connection ID: Represents the anchor connection number of the MADP instance.

(2) MX Configuration ID: Identifies the MX configuration of the MADP instance.

(3) Traffic Flow Template Uplink: Traffic Flow Template, as defined in Appendix C.2.16, to be used in the uplink direction.

(4) Traffic Flow Template Downlink: Traffic Flow Template, as defined in Appendix C.2.16, to be used in the downlink direction.

The representation of the message is as follows:

```
object {
   UniqueSessionId unique_session_id;
   MXAppMADPAssoc app_madp_assoc_list <1..*>;
} MXAppMADPAssocReq : MXBase;
```

Where each measurement MXAppMADPAssoc is represented by the following:

```
object {
   JSONNumber connection_id;
   JSONNumber mx_configuration_id
   TrafficFlowTemplate tft_ul_list <1..*>;
   TrafficFlowTemplate tft_dl_list <1..*>;
} MXAppMADPAssoc;
```

C.1.9.2. MX Application MADP Association Response

This message is sent by the NCM to the CCM to confirm the selected MADP instances provided in the MX Application MADP Association Request by the CCM.

In addition to the base information (Appendix C.2.1), it contains information if the request has been successful.

The representation of the message is as follows:

```
object {
   JSONBool is_success;
} MXAppMADPAssocResp : MXBase;
```

C.1.10. MX SSID Indication

This message is sent by the NCM to the CCM to indicate the list of allowed SSIDs that are supported by the MAMS entity on the network side. It contains the list of SSIDs.

Each SSID consists of the type of SSID (which can be one of the following: SSID, BSSID, or HESSID) and the SSID itself.

The representation of the message is as follows:

```
object {
   SSID ssid_list <1..*>;
} MXSSIDIndication : MXBase;
```

Where each SSID is defined as follows:

```
object {
   JSONString ssid_type;
   JSONString ssid;
} SSID;
```

C.1.11. Measurements

C.1.11.1. MX Measurement Configuration

This message is sent from the NCM to the CCM to configure the period measurement reporting at the CCM. The message contains a list of measurement configurations, with each element containing the following information:

(a) Connection ID: Connection ID of the delivery connection for which the reporting is being configured.

(b) Connection Type: Connection type for which the reporting is being configured. Can be "LTE", "Wi-Fi", "5G_NR".

(c) Measurement Report Configuration: Actual report configuration based on the Connection Type, as defined in Appendix C.2.17.

The representation of the message is as follows:

```
object {
   MeasReportConf measurement_configuration <1..*>;
} MXMeasReportConf : MXBase;
```

Where each measurement MeasReportConf is represented by the following:

```
object {
   JSONNumber connection_id;
   JSONString connection_type;
   MeasReportConfs meas_rep_conf <1..*>;
} MeasReportConf;
```

C.1.11.2. MX Measurement Report

This message is periodically sent by the CCM to the NCM after measurement configuration. In addition to the base information, it contains the following information:

(a) Unique Session ID: Same identifier as the identifier provided in the MX Capability Response. Described in Appendix C.2.2.

(b) Measurement report for each delivery connection is measured by the client as defined in Appendix C.2.18.

The representation of the message is as follows:

```
object {
   UniqueSessionId unique_session_id;
   MXMeasRep measurement_reports <1..*>;
} MXMeasurementReport : MXBase;
```

C.1.12. Keep-Alive

C.1.12.1. MX Keep-Alive Request

An MX Keep-Alive Request can be sent from either the NCM or the CCM on expiry of the Keep-Alive timer or a handover event. The peer shall respond to this request with an MX Keep-Alive Response. In the case of no response from the peer, the MAMS connection shall be assumed to be broken, and the CCM shall establish a new connection by sending MX Discover messages.

In addition to the base information, it contains the following information:

(a) Keep-Alive Reason: Reason for sending this message, can be "Timeout" or "Handover".

(b) Unique Session ID: Identifier for the CCM-NCM association Appendix C.2.2.

(c) Connection ID: Connection ID for which handover is detected, if the reason is "Handover".

(d) Delivery Node ID: The target delivery node ID (ECGI or Wi-Fi AP ID/MAC address) to which the handover is executed.

The representation of the message is as follows:

```
object {
   JSONString keep_alive_reason;
   UniqueSessionId unique_session_id;
   JSONNumber connection_id;
   JSONString delivery_node_id;
} MXKeepAliveReq : MXBase;
```

C.1.12.2. MX Keep-Alive Response

On receiving an MX Keep-Alive Request from a peer, the NCM/CCM shall immediately respond with an MX Keep-Alive Response on the same delivery path from where the request arrived. In addition to the base information, it contains the unique session identifier for the CCM-NCM association (defined in Appendix C.2.2)

The representation of the message is as follows:

```
object {
   UniqueSessionId unique_session_id;
} MXKeepAliveResp : MXBase;
```

C.1.13. Session Termination Procedure

C.1.13.1. MX Session Termination Request

In the event where the NCM or CCM can no longer handle MAMS for any reason, it can send an MX Session Termination Request to the peer. In addition to the base information, it contains a Unique Session ID and the reason for the termination; this can be "MX_NORMAL_RELEASE", "MX_NO_RESPONSE", or "INTERNAL_ERROR".

The representation of the message is as follows:

```
object {
   UniqueSessionId unique_session_id;
   JSONString reason;
} MXSessionTerminationReq : MXBase;
```

C.1.13.2. MX Session Termination Response

On receipt of an MX Session Termination Request from a peer, the NCM/CCM shall respond with MX Session Termination Response on the same delivery path where the request arrived and clean up the MAMS-related resources and settings. The CCM shall reinitiate a new session with MX Discover messages.

The representation of the message is as follows:

```
object {
   UniqueSessionId unique_session_id;
} MXSessionTerminationResp : MXBase;
```

C.1.14. Network Analytics

C.1.14.1. MX Network Analytics Request

This message is sent by the CCM to the NCM to request parameters like bandwidth, jitter, latency, and signal quality predicted by the network analytics function. In addition to the base information, it contains the following parameter:

(a) Unique Session ID: Same identifier as the identifier provided in the MX Capability Response. Described in Appendix C.2.2.

(b) Parameter List: List of parameters in which the CCM is interested: one or more of "bandwidth", "jitter", "latency", and "signal_quality".

The representation of the message is as follows:

```
object {
   UniqueSessionId unique_session_id;
   JSONString params <1..*>;
} MXNetAnalyticsReq : MXBase;
```

Where the params object can take one or more of the following values:

```
"bandwidth"
"jitter"
"latency"
"signal_quality"
```

C.1.14.2. MX Network Analytics Response

This message is sent by the NCM to the CCM in response to the MX Network Analytics Request. For each delivery connection that the client has, the NCM reports the requested parameter predictions and their respective likelihoods (between 1 and 100 percent).

In addition to the base information, it contains the following parameters:

(a) Number of Delivery Connections: The number of delivery connections that are currently configured for the client.

(b) The following information is provided for each delivery connection:

   (1) Connection ID: Connection ID of the delivery connection for which the parameters are being predicted.

   (2) Connection Type: Type of connection. Can be "Wi-Fi", "5G_NR", "MulteFire", or "LTE".

   (3)

List of Parameters for which Prediction is requested, where each of the predicted parameters consists of the following:

(a) Parameter Name: Name of the parameter being predicted. Can be one of "bandwidth", "jitter", "latency", or "signal_quality".

(b) Additional Parameter: If Parameter name is "signal_quality", then this qualifies the quality parameter like "lte_rsrp", "lte_rsrq", "nr_rsrp", "nr_rsrq", or "wifi_rssi".

(c) Predicted Value: Provides the predicted value of the parameter and, if applicable, the additional parameter.

(d) Likelihood: Provides a stochastic likelihood of the predicted value.

(e) Validity Time: The time duration for which the predictions are valid.

The representation of the message is as follows:

```
object {
   MXAnalyticsList param_list <1..*>;
} MXNetAnalyticsResp : MXBase;
```

Where MXAnalyticsList is defined as follows:

```
object {
   JSONNumber connection_id;
   JSONString connection_type;
   ParamPredictions predictions <1..*>;
} MXAnalyticsList;
```

Where each ParamPredictions item is defined as:

```
object {
   JSONString param_name;
   [JSONString additional_param;]
   JSONNumber prediction;
   JSONNumber likelihood;
   JSONNumber validity_time;
} ParamPredictions;
```

C.2. Protocol Specification: Data Types

C.2.1. MXBase

This is the base information that every message between the CCM and NCM exchanges shall have as mandatory information. It contains the following information:

(a) Version: Version of MAMS used.

(b) Message Type: Message type being sent, where the following are considered valid values:

```
"mx_discover"
"mx_system_info"
"mx_capability_req"
"mx_capability_rsp"
"mx_capability_ack"
"mx_up_setup_conf_req"
"mx_up_setup_cnf"
"mx_reconf_req"
"mx_reconf_rsp"
"mx_path_est_req"
"mx_path_est_results"
"mx_traffic_steering_req"
"mx_traffic_steering_rsp"
"mx_ssid_indication"
"mx_keep_alive_req"
"mx_keep_alive_rsp"
"mx_measurement_conf"
"mx_measurement_report"
"mx_session_termination_req"
"mx_session_termination_rsp"
"mx_app_madp_assoc_req"
"mx_app_madp_assoc_rsp"
"mx_network_analytics_req"
"mx_network_analytics_rsp"
```

(c) Sequence Number: Sequence number to uniquely identify a particular message exchange, e.g., MX Capability Request/Response/Acknowledge.

The representation of this data type is as follows:

```
object {
   JSONString version;
   JSONString message_type;
   JSONNumber sequence_num;
} MXBase;
```

C.2.2. Unique Session ID

This data type represents the unique session ID between a CCM and NCM entity. It contains an NCM ID that is unique in the network and a session ID that is allocated by the NCM for that session. On receipt of the MX Discover message, if the session exists, then the old session ID is returned in the MX System Info message; otherwise, the NCM allocates a new session ID for the CCM and sends the new ID in the MX System Info message.

The representation of this data type is as follows:

```
object {
   JSONNumber ncm_id;
   JSONNumber session_id;
} UniqueSessionId;
```

C.2.3. NCM Connections

This data type represents the connection available at the NCM for MAMS connectivity toward the client. It contains a list of NCM connections available, where each connection has the following information:

(a) Connection Information: See Appendix C.2.4.

(b) NCM Endpoint Information: Contains the IP address and port exposed by the NCM endpoint for the CCM.

The representation of this data type is as follows:

```
object {
   NCMConnection items <1..*>;
} NCMConnections;
```

where NCMConnection is defined as:

```
object {
   NCMEndPoint ncm_end_point;
} NCMConnection : ConnectionInfo;
```

where NCMEndPoint is defined as:

```
object {
   JSONString ip_address;
   JSONNumber port;
} NCMEndPoint;
```

C.2.4. Connection Information

This data type provides the mapping of connection ID and connection type. It contains the following information:

(a) Connection ID: Unique number identifying the connection.

(b) Connection Type: Type of connection can be "Wi-Fi", "5G_NR", "MulteFire", or "LTE".

The representation of this data type is as follows:

```
object {
   JSONNumber connection_id;
   JSONString connection_type;
} ConnectionInfo;
```

C.2.5. Features and Their Activation Status

This data type provides the list of all features with their activation status. Each feature status contains the following:

(a) Feature Name: The name of the feature can be one of the following:

```
"lossless_switching"
"fragmentation"
"concatenation"
"uplink_aggregation"
"downlink_aggregation"
"measurement"
```

(b) Active status: Activation status of the feature: "true" means that the feature is active, and "false" means that the feature is inactive.

The representation of this data type is as follows:

```
object {
   FeatureInfo items <1..*>;
} FeaturesActive;
```

where FeatureInfo is defined as:

```
object {
   JSONString feature_name;
   JSONBool active;
} FeatureInfo;
```

C.2.6. Anchor Connections

This data type contains the list of Connection Information items (Appendix C.2.4) that are supported on the anchor (core) side.

The representation of this data type is as follows:

```
object {
   ConnectionInfo items <1..*>;
} AnchorConnections;
```

C.2.7. Delivery Connections

This data type contains the list of Connection Information (Appendix C.2.4) that are supported on the delivery (access) side.

The representation of this data type is as follows:

```
object {
   ConnectionInfo items <1..*>;
} DeliveryConnections;
```

C.2.8. Method Support

This data type provides the support for a particular convergence or adaptation method. It consists of the following:

(a) Method: Name of the method.

(b) Supported: Whether the method listed above is supported or not. Possible values are "true" and "false".

The representation of this data type is as follows:

```
object {
   JSONString method;
   JSONBool supported;
} MethodSupport;
```

C.2.9. Convergence Methods

This data type contains the list of all convergence methods and their support status. The possible convergence methods are:

```
"GMA"
"MPTCP_Proxy"
"GRE_Aggregation_Proxy"
"MPQUIC"
```

The representation of this data type is as follows:

```
object {
   MethodSupport items <1..*>;
} ConvergenceMethods;
```

C.2.10. Adaptation Methods

This data type contains the list of all adaptation methods and their support status. The possible adaptation methods are:

```
"UDP_without_DTLS"
"UDP_with_DTLS"
"IPsec"
"Client_NAT"
```

The representation of this data type is as follows:

```
object {
   MethodSupport items <1..*>;
} AdaptationMethods;
```

C.2.11. Setup of Anchor Connections

This data type represents the setup configuration for each anchor connection that is required on the client's side. It contains the following information, in addition to the connection ID and type of the anchor connection:

(a) Number of Active MX Configurations: If more than one active configuration is present for this anchor, then this identifies the number of such connections.

(b) The following convergence parameters are provided for each active configuration:

   (1) MX Configuration ID: Present if there are multiple active configurations. Identifies the configuration for this MADP instance ID.

   (2) Convergence Method: Convergence method selected. Has to be one of the supported convergence methods listed in Appendix C.2.9.

   (3) Convergence Method Parameters: Described in Appendix C.2.11.1

   (4) Number of Delivery Connections: The number of delivery connections (access side) that are supported for this anchor connection.

   (5) Setup of delivery connections: Described in Appendix C.2.11.2.

The representation of this data type is as follows:

```
object {
   SetupAnchorConn items <1..*>;
} SetupAnchorConns;
```

Where each anchor connection configuration is defined as follows:

```
object {
   [JSONNumber num_active_mx_conf;]
   ConvergenceConfig convergence_config
} SetupAnchorConn : ConnectionInfo;
```

where each Convergence configuration is defined as follows:

```
object {
   [JSONNumber mx_configuration_id;]
   JSONString convergence_method;
   ConvergenceMethodParam convergence_method_params;
   JSONNumber num_delivery_connections;
   SetupDeliveryConns delivery_connections;
} ConvergenceConfig;
```

C.2.11.1. Convergence Method Parameters

This data type represents the parameters used for the convergence method and contains the following:

(a) Proxy IP: IP address of the proxy that is provided by the selected convergence method.

(b) Proxy Port: Port of the proxy that is provided by the selected convergence method.

The representation of this data type is as follows:

```
object {
   JSONString proxy_ip;
   JSONString proxy_port;
   JSONString client_key;
} ConvergenceMethodParam;
```

C.2.11.2. Setup Delivery Connections

This is the list of delivery connections and their parameters to be configured on the client. Each delivery connection defined by its connection information (Appendix C.2.4) optionally contains the following:

(a) Adaptation Method: Selected adaptation method name. This shall be one of the methods listed in Appendix C.2.10.

(b) Adaptation Method Parameters: Depending on the adaptation method, one or more of the following parameters shall be provided.

   (1) Tunnel IP address

   (2) Tunnel Port number

   (3) Shared Secret

   (4) MX header optimization: If the adaptation method is UDP_without_DTLS or UDP_with_DTLS, and convergence is GMA, then this flag represents whether or not

the checksum field and the length field in the IP header of an MX PDU should be recalculated by the MX Convergence Layer. The possible values are "true" and "false". If it is "true", both fields remain unchanged; otherwise, both fields should be recalculated. If this field is not present, then the default of "false" should be considered.

The representation of this data type is as follows:

```
object {
   SetupDeliveryConn items <1..*>;
} SetupDeliveryConns;
```

where each "SetupDeliveryConn" consists of the following:

```
object {
  [JSONString adaptation_method;]
  [AdaptationMethodParam adaptation_method_param;]
} SetupDeliveryConn : ConnectionInfo;
```

where AdaptationMethodParam is defined as:

```
object {
   JSONString tunnel_ip_addr;
   JSONString tunnel_end_port;
   JSONString shared_secret;
   [JSONBool mx_header_optimization;]
} AdaptationMethodParam;
```

C.2.12. Init Probe Results

This data type provides the results of the init probe request made by the NCM. It consists of the following information:

(a) Lost Probes: Percentage of probes lost.

(b) Probe Delay: Average delay of probe message, in microseconds.

(c) Probe Rate: Probe rate achieved, in megabits per second.

The representation of this data type is as follows:

```
object {
   JSONNumber lost_probes_percentage;
   JSONNumber probe_rate_Mbps;
} InitProbeResults;
```

C.2.13. Active Probe Results

This data type provides the results of the active probe request made by the NCM. It consists of the following information:

(a) Average Probe Throughput: Average active probe throughput achieved, in megabits per second.

The representation of this data type is as follows:

```
object {
   JSONNumber avg_tput_last_probe_duration_Mbps;
} ActiveProbeResults;
```

C.2.14. Downlink Delivery

This data type represents the list of connections that are enabled on the delivery side to be used in the downlink direction.

The representation of this data type is as follows:

```
object {
   JSONNumber connection_id <1..*>;
} DLDelivery;
```

C.2.15. Uplink Delivery

This data type represents the list of connections and parameters enabled for the delivery side to be used in the uplink direction.

The uplink delivery consists of multiple uplink delivery entities, where each entity consists of a Traffic Flow Template (TFT) (Appendix C.2.16) and a list of connection IDs in the uplink, where traffic qualifying for such a Traffic Flow Template can be redirected.

The representation of this data type is as follows:

```
object {
   ULDeliveryEntity ul_del <1..*>;
} ULDelivery;
```

Where each uplink delivery entity consists of the following data type:

```
object {
   TrafficFlowTemplate ul_tft <1..*>;
   JSONNumber connection_id <1..*>;
} ULDeliveryEntity;
```

C.2.16. Traffic Flow Template

The Traffic Flow Template generally follows the guidelines specified in [ServDesc3GPP].

The Traffic Flow Template in MAMS consists of one or more of the following:

(a) Remote Address and Mask: IP address and subnet for remote addresses represented in Classless Inter-Domain Routing (CIDR) notation. Default: "0.0.0.0/0".

(b) Local Address and Mask: IP address and subnet for local addresses represented in CIDR notation. Default: "0.0.0.0/0"

(c) Protocol Type: IP protocol number of the payload being carried by an IP packet (e.g., UDP, TCP). Default: 255.

(d) Local Port Range: Range of ports for local ports for which the Traffic Flow Template is applicable. Default: Start=0, End=65535.

(e) Remote Port Range: Range of ports for remote ports for which the Traffic Flow Template is applicable. Default: Start=0, End=65535.

(f) Traffic Class: Represented by Type of Service in IPv4 and Traffic Class in IPv6. Default: 255

(g) Flow Label: Flow label for IPv6, applicable only for IPv6 protocol type. Default: 0.

The representation of this data type is as follows:

```
object {
   JSONString remote_addr_mask;
   JSONString local_addr_mask;
   JSONNumber protocol_type;
   PortRange local_port_range;
   PortRange remote_port_range;
   JSONNumber traffic_class;
   JSONNumber flow_label;
} TrafficFlowTemplate;
```

Where the port range is defined as follows:

```
object {
   JSONNumber start;
   JSONNumber end;
} PortRange;
```

C.2.17. Measurement Report Configuration

This data type represents the configuration done by the NCM toward the CCM for reporting measurement events.

(a) Measurement Report Parameter: Parameter that shall be measured and reported. This is dependent on the connection type:

  (1) For the connection type of "Wi-Fi", the allowed measurement type parameters are "WLAN_RSSI", "WLAN_LOAD", "UL_TPUT", "DL_TPUT", "EST_UL_TPUT", and "EST_DL_TPUT".
  (2) For the connection type of "LTE", the allowed measurement type parameters are "LTE_RSRP", "LTE_RSRQ", "UL_TPUT", and "DL_TPUT".
  (3) For the connection type of "5G_NR", the allowed measurement type parameters are "NR_RSRP", "NR_RSRQ", "UL_TPUT", and "DL_TPUT".

(b) Threshold: High and low threshold for reporting.

(c) Period: Period for reporting, in milliseconds.

The representation of this data type is as follows:

```
object {
   JSONString meas_rep_param;
   Threshold meas_threshold;
   JSONNumber meas_period;
} MeasReportConfs;
```

Where "Threshold" is defined as follows:

```
object {
   JSONNumber high;
   JSONNumber low;
} Threshold;
```

C.2.18. Measurement Report

This data type represents the measurements reported by the CCM for each access network measured. This type contains the connection information, the Delivery Node ID that identifies either the cell (ECGI) or the Wi-Fi Access Point ID or MAC address (or equivalent identifier in other technologies), and the actual measurement performed by the CCM in the last measurement period.

The representation of this data type is as follows:

```
object {
   JSONNumber connection_id;
   JSONString connection_type;
   JSONString delivery_node_id;
   Measurement measurements <1..*>;
} MXMeasRep;
```

Where Measurement is defined as the key-value pair of the measurement type and value. The exact measurement type parameter reported for a given connection depends on its Connection Type. The measurement type parameters, for each Connection Type, are specified in Appendix C.2.17.

```
object {
   JSONString measurement_type;
   JSONNumber measurement_value;
} Measurement;
```

C.3. Schemas in JSON

C.3.1. MX Base Schema

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "definitions": {
    "message_type_def": {
      "enum": [
        "mx_discover",
        "mx_system_info",
        "mx_capability_req",
        "mx_capability_rsp",
        "mx_capability_ack",
        "mx_up_setup_conf_req",
        "mx_up_setup_cnf",
        "mx_reconf_req",
        "mx_reconf_rsp",
        "mx_path_est_req",
        "mx_path_est_results",
        "mx_traffic_steering_req",
        "mx_traffic_steering_rsp",
        "mx_ssid_indication",
        "mx_keep_alive_req",
        "mx_keep_alive_rsp",
        "mx_measurement_conf",
        "mx_measurement_report",
        "mx_session_termination_req",
        "mx_session_termination_rsp",
        "mx_app_madp_assoc_req",
        "mx_app_madp_assoc_rsp",
        "mx_network_analytics_req",
        "mx_network_analytics_rsp"
      ],
      "type": "string"
    },
    "sequence_num_def": {
      "minimum": 1,
      "type": "integer"
    },
    "version_def": {
      "type": "string"
    }
   },
  "id": "https://example.com/mams/mx_base_def.json"
}
```

C.3.2. MX Definitions

```
{
    "$schema": "https://json-schema.org/draft-04/schema#",
    "definitions": {
        "adapt_method": {
            "enum": [
                "UDP_without_DTLS",
                "UDP_with_DTLS",
                "IPsec",
                "Client_NAT"
            ],
            "type": "string"
        },
        "conv_method": {
            "enum": [
                "GMA",
                "MPTCP_Proxy",
                "GRE_Aggregation_Proxy",
                "MPQUIC"
            ],
            "type": "string"
        },
        "supported": {
            "type": "boolean"
        },
        "active": {
            "type": "boolean"
        },
        "connection_id": {
            "type": "integer"
        },
        "feature_name": {
            "enum": [
                "lossless_switching",
                "fragmentation",
                "concatenation",
                "uplink_aggregation",
                "downlink_aggregation",
                "measurement"
                "probing"
            ],
            "type": "string"
        },
        "connection_type": {
            "enum": [
                "Wi-Fi",
                "5G_NR",
                "MulteFire",
                "LTE"
            ],
            "type": "string"
        },
        "ip_address": {
                "type": "string"
        },
        "port": {
            "maximum": 65535,
            "minimum": 1,
```

```
            "type": "integer"
        },
        "adaptation_method": {
            "allOf" : [
                { "$ref": "#/definitions/adapt_method" },
                { "$ref": "#/definitions/supported" }
            ]
        },
        "connection": {
            "allOf" : [
                { "$ref": "#/definitions/connection_id" },
                { "$ref": "#/definitions/connection_type" }
            ]
        },
        "convergence_method": {
            "allOf": [
                { "$ref": "#/definitions/conv_method" },
                { "$ref": "#/definitions/supported" }
            ]
        },
        "feature_status": {
            "allOf": [
                { "$ref": "#/definitions/feature_name" },
                { "$ref": "#/definitions/active" }
            ]
        },
        "ncm_end_point": {
            "allOf" : [
                { "$ref" : "#/definitions/ip_address" },
                { "$ref" : "#/definitions/port" }
            ]
        },
        "capability_acknowledgment" : {
            "enum" : [
                "MX_ACCEPT",
                "MX_REJECT"
            ],
            "type" : "string"
        },
        "threshold" : {
            "high" : {
                "type" : "integer"
            },
            "low" : {
                "type" : "integer"
            },
            "type" : "object"
        },
        "meas_report_param" : {
            "enum" : [
                    "WLAN_RSSI",
                    "WLAN_LOAD",
                    "LTE_RSRP",
                    "LTE_RSRQ",
                    "UL_TPUT",
                    "DL_TPUT",
                    "EST_UL_TPUT",
                    "EST_DL_TPUT",
```

```
                    "NR_RSRP",
                    "NR_RSRQ"
                ],
                "type" : "string"
            },
            "meas_report_conf" : {
                "meas_rep_param" : {
                    "$ref" : "#definitions/meas_report_param"
                },
                "meas_threshold" : {
                    "$ref" : "#definitions/threshold"
                },
                "meas_period_ms" : {
                    "type" : "integer"
                },
                "type" : "object"
            },
            "ssid_types" : {
                "enum" : [
                    "ssid",
                    "bssid",
                    "hessid"
                ],
                "type" : "string"
            },
            "ip_addr_mask" : {
                "type" : "string",
                "default" : "0.0.0.0/0"
            },
            "port_range" : {
                "start" : {
                    "type" : "integer",
                    "default" : 0
                },
                "end" : {
                    "type" : "integer",
                    "default" : 65535
                }
            },
            "traffic_flow_template" : {
                "remote_addr_mask" : {
                    "$ref" : "#definitions/ip_addr_mask" },
                "local_addr_mask" : {
                    "$ref" : "#definitions/ip_addr_mask" },
                "protocol_type" : {
                    "type" : "integer",
                    "minimum" : 0,
                    "maximum" : 255
                },
                "local_port_range" : {
                    "$ref" : "#definitions/port_range" },
                "remote_port_range" : {
                    "$ref" : "#definitions/port_range" },
                "traffic_class" : {
                    "type" : "integer",
                    "default" : 255
                },
                "flow_label" : {
```

```
                "type" : "integer",
                "default" : 0
            }
        },
        "delivery_node_id" : {
            "type" : "string"
        },
        "unique_session_id" : {
            "type" : "object",
            "ncm_id" : {
              "type" : "integer"
            },
            "session_id" : {
              "type" : "integer"
            }
        },
        "keep_alive_reason" : {
            "enum" : [
                "Timeout",
                "Handover"
            ],
            "type" : "string"
        },
        "connection_status" : {
            "enum" : [
                "disabled",
                "enabled",
                "connected"
            ],
            "type" : "string",
            "default" : "connected"
        },
        "adaptation_param" : {
           "udp_adapt_port" : {
             "type" : "integer"
           }
        },
        "probe_param" : {
            "probe_port" : {
                "type" : "integer"
            },
            "anchor_conn_id" : {
                "type" : "integer"
            },
            "mx_configuration_id" : {
                "type" : "integer"
            }
    },
        "client_param" : {
            "connection_id" : {
                "type" : "integer"
            },
            "adapt_param" : {
                "type" : {"$ref" : "#definitions/adaptation_param" }
            }
        }
    },
            "adapt_param": {
```

```
            "tunnel_ip_addr": {
                "type": "string"
            },
            "tunnel_end_port": {
                "type": "integer"
            },
            "shared_secret": {
                "type": "string"
            },
            "mx_header_optimization": {
                "type": "boolean",
                "default": false
            }
        },
        "delivery_connection": {
            "connection_id": {
                "$ref": "#definitions/connection_id"
            },
            "connection_type": {
                "$ref": "#definitions/connection_type"
            },
            "adaptation_method": {
                "$ref": "#definitions/adapt_method"
            },
            "adaptation_method_param": {
                "$ref": "#definitions/adapt_param"
            }
        },
        "app_madp_assoc": {
            "anchor_conn_id" : {
                "type" : "integer"
            },
            "mx_configuration_id" : {
                "type" : "integer"
            }

            "ul_tft_list": {
                "items": {
                    "$ref": "#definitions/traffic_flow_template"
                },
                "type": "array"
            },
            "dl_tft_list": {
                "items": {
                    "$ref": "#definitions/traffic_flow_template"
                },
                "type": "array"
            }
        },
        "predict_param_name": {
            "enum": [
                "validity_time",
                "bandwidth",
                "jitter",
                "latency",
                "signal_quality"
            ],
            "type": "string"
```

```
        },
        "predict_add_param_name": {
            "enum": [
                "WLAN_RSSI",
                "WLAN_LOAD",
                "LTE_RSRP",
                "LTE_RSRQ",
                "NR_RSRP",
                "NR_RSRQ"
            ],
            "type": "string"
        },
    "id": "https://example.com/mams/definitions.json"
}
```

C.3.3. MX Discover

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "additionalProperties": false,
  "id": "https://example.com/mams/mx_discover.json",
  "properties": {
    "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
    "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
    "version": {"$ref": "mx_base_def.json#/version_def"}
  },
  "type": "object"
}
```

C.3.4. MX System Info

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "additionalProperties": false,
  "id": "https://example.com/mams/mx_system_info.json",
  "properties": {
    "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
    "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
    "version": {"$ref": "mx_base_def.json#/version_def"},
    "ncm_connections": {
      "type": "array",
      "items": [
        {"$ref": "definitions.json#/connection"},
        {"$ref": "definitions.json#/ncm_end_point"}
      ]
    }
  },
  "type": "object"
}
```

C.3.5. MX Capability Request

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "additionalProperties": false,
  "id": "https://example.com/mams/mx_capability_req.json",
  "properties": {
    "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
    "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
    "version": {"$ref": "mx_base_def.json#/version_def"},
    "adaptation_methods": {
      "items": {"$ref": "definitions.json#/adaptation_method"},
      "type": "array"
    },
    "anchor_connections": {
      "items": {"$ref": "definitions.json#/connection"},
      "type": "array"
    },
    "convergence_methods": {
      "items": {"$ref": "definitions.json#/convergence_method"},
      "type": "array"
    },
    "delivery_connections": {
      "items": {"$ref": "definitions.json#/connection"},
      "type": "array"
    },
    "feature_active": {
      "items": {"$ref": "definitions.json#/feature_status"},
      "type": "array"
    },
    "num_anchor_connections": {
      "type": "integer"
    },
    "num_delivery_connections": {
      "type": "integer"
    }
  },
  "type": "object"
}
```

C.3.6. MX Capability Response

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "additionalProperties": false,
  "id": "https://example.com/mams/mx_capability_rsp.json",
  "properties": {
    "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
    "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
    "version": {"$ref": "mx_base_def.json#/version_def"},
    "adaptation_methods": {
      "items": {"$ref": "definitions.json#/adaptation_method"},
      "type": "array"
    },
    "anchor_connections": {
      "items": {"$ref": "definitions.json#/connection"},
      "type": "array"
    },
    "convergence_methods": {
      "items": {"$ref": "definitions.json#/convergence_method"},
      "type": "array"
    },
    "delivery_connections": {
      "items": {"$ref": "definitions.json#/connection"},
      "type": "array"
    },
    "feature_active": {
      "items": {"$ref": "definitions.json#/feature_status"},
      "type": "array"
    },
    "num_anchor_connections": {
      "type": "integer"
    },
    "num_delivery_connections": {
      "type": "integer"
    },
    "unique_session_id": {
      "$ref": "definitions.json#/unique_session_id"
    }
  },
  "type": "object"
}
```

C.3.7. MX Capability Acknowledge

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "definitions": {},
  "id": "https://example.com/mams/mx_capability_ack.json",
  "properties": {
     "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
     "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
     "version": {"$ref": "mx_base_def.json#/version_def"},
     "unique_session_id": {
       "$ref": "definitions.json#/unique_session_id"},
     "capability_ack": {
       "$ref": "definitions.json#/capability_acknowledgment"}
  },
  "type": "object"
}
```

C.3.8. MX Reconfiguration Request

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "definitions": {},
  "id": "https://example.com/mams/mx_reconf_req.json",
  "properties": {
    "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
    "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
    "version": {"$ref": "mx_base_def.json#/version_def"},
    "unique_session_id": {
      "$ref": "definitions.json#/unique_session_id"
    },
    "connection_id": {"$ref": "definitions.json#/connection_id"},
    "ip_address": {"$ref": "definitions.json#/ip_address"},
    "mtu_size": {
      "maximum": 65535,
      "minimum": 1,
      "type": "integer"
    },
    "ssid": {
      "type": "string"
    },
    "reconf_action": {
      "enum": [
        "release",
        "setup",
        "update"
      ],
      "id": "/properties/reconf_action",
      "type": "string"
    },
    "connection_status": {
       "$ref": "definitions.json#/connection_status"},
    "delivery_node_id": {
       "$ref": "definitions.json#/delivery_node_id"}
  },
  "type": "object"
}
```

C.3.9. MX Reconfiguration Response

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "definitions": {},
  "id": "https://example.com/mams/mx_reconf_rsp.json",
  "properties": {
    "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
    "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
    "version": {"$ref": "mx_base_def.json#/version_def"}
  },
  "type": "object"
}
```

C.3.10. MX UP Setup Configuration Request

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "additionalProperties": false,
  "definitions": {
    "convergence_configuration": {
      "mx_configuration_id": {"type": "integer"},
      "convergence_method": {
        "$ref": "definitions.json#/conv_method"},
      "convergence_method_params": {
        "properties": {
          "proxy_ip": {"$ref": "definitions.json#/ip_address"},
          "proxy_port": {"$ref": "definitions.json#/port"},
          "client_key": {"$ref": "definitions.json#/client_key"}
        },
        "type": "object"
      },
      "num_delivery_connections": {
        "type": "integer"
      },
      "delivery_connections": {
        "items": {"$ref": "definitions.json#/delivery_connection"},
        "type": "array"
      }
    }
  },
  "id": "https://example.com/mams/mx_up_setup_conf_req.json",
  "properties": {
    "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
    "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
    "version": {"$ref": "mx_base_def.json#/version_def"},
    "num_anchor_connections": {
      "type": "integer"
    },
    "anchor_connections": {
      "items": {
        "properties": {
          "connection_id": {
            "$ref": "definitions.json#/connection_id"},
          "connection_type": {
            "$ref": "definitions.json#/connection_type"},
          "num_active_mx_conf": {"type": "integer"},
          "convergence_config": {
            "items": {
              "$ref": "definitions/convergence_configuration"},
            "type": "array"
          }
        },
        "type": "object"
      },
      "type": "array"
    }
  },
  "type": "object"
}
```

C.3.11. MX UP Setup Confirmation

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "definitions": {},
  "id": "https://example.com/mams/mx_up_setup_cnf.json",
  "properties": {
    "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
    "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
    "version": {"$ref": "mx_base_def.json#/version_def"},
    "unique_session_id": {
      "$ref": "definitions.json#/unique_session_id"},
    "probe_param": {"$ref": "definitions.json#/probe_param"},
    "num_delivery_conn": {
      "type": "integer"
    },
    "client_params": {
      "type": "array",
      "items": [
        {"$ref": "definitions.json#/client_param"}
      ]
    }
  },
  "type": "object"
}
```

C.3.12. MX Traffic Steering Request

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "definitions": {
    "conn_list": {
      "items": {"$ref": "definitions.json#/connection_id"},
      "type": "array"
    },
    "ul_delivery": {
      "ul_tft": {
        "$ref": "definitions.json#/traffic_flow_template"},
      "connection_list": {"$ref": "#definitions/conn_list"}
    }
  },
  "id": "https://example.com/mams/mx_traffic_steering_req.json",
  "properties": {
    "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
    "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
    "version": {"$ref": "mx_base_def.json#/version_def"},
    "connection_id": {"$ref": "definitions.json#/connection_id"},
    "mx_configuration_id": {"type": "integer"},
    "downlink_delivery": {
      "items": {"$ref": "definitions.json#/connection_id"},
      "type": "array"
    },
    "feature_active": {
      "items": {"$ref": "definitions.json#/feature_status"},
      "type": "array"
    },
    "default_uplink_delivery": {
      "type": "integer"
    },
    "uplink_delivery": {
      "items": {"$ref": "#definitions/ul_delivery"},
      "type": "array"
    }
  },
  "type": "object"
}
```

C.3.13. MX Traffic Steering Response

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "definitions": {},
  "id": "https://example.com/example.json",
  "properties": {
    "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
    "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
    "version": {"$ref": "mx_base_def.json#/version_def"},
    "unique_session_id": {
      "$ref": "definitions.json#/unique_session_id"},
    "feature_active": {
      "items": {"$ref": "definitions.json#/feature_status"},
      "type": "array"
    }
  },
  "type": "object"
}
```

C.3.14. MX Application MADP Association Request

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "definitions": {},
  "id": "https://example.com/example.json",
  "properties": {
    "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
    "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
    "version": {"$ref": "mx_base_def.json#/version_def"},
    "unique_session_id": {
      "$ref": "definitions.json#/unique_session_id"},
    "app_madp_assoc_list": {
      "items": {
        "$ref": "definitions.json#/app_madp_assoc"
       },
      "type": "array"
    }
  },
  "type": "object"
}
```

C.3.15. MX Application MADP Association Response

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "definitions": {},
  "id": "https://example.com/example.json",
  "properties": {
    "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
    "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
    "version": {"$ref": "mx_base_def.json#/version_def"},
    "unique_session_id": {
      "$ref": "definitions.json#/unique_session_id"},
    "is_success": {
      "type": "boolean"
    }
  },
  "type": "object"
}
```

C.3.16. MX Path Estimation Request

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "definitions": {},
  "id": "https://example.com/mams/mx_path_est_req.json",
  "properties": {
    "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
    "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
    "version": {"$ref": "mx_base_def.json#/version_def"},
    "active_probe_ack_req": {
      "enum": [
        "no",
        "yes"
      ],
      "type": "string"
    },
    "active_probe_freq_ms": {
      "maximum": 10000,
      "minimum": 100,
      "type": "integer"
    },
    "active_probe_size_bytes": {
      "maximum": 1500,
      "minimum": 100,
      "type": "integer"
    },
    "active_probe_duration_sec": {
      "maximum": 100,
      "minimum": 10,
      "type":  "integer"
        },
    "connection_id": {"$ref": "definitions#/connection_id"},
    "init_probe_ack_req": {
      "enum": [
        "no",
        "yes"
      ],
      "type": "string"
    },
    "init_probe_size_bytes": {
      "maximum": 1500,
      "minimum": 100,
      "type": "integer"
    },
    "init_probe_test_duration_ms": {
      "maximum": 10000,
      "minimum": 100,
      "type": "integer"
    },
    "init_probe_test_rate_Mbps": {
      "maximum": 100,
      "minimum": 1,
      "type": "integer"
    }
  },
  "type": "object"
}
```

C.3.17. MX Path Estimation Results

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "definitions": {},
  "id": "https://example.com/mams/mx_path_est_results.json",
  "properties": {
    "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
    "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
    "version": {"$ref": "mx_base_def.json#/version_def"},
    "unique_session_id": {
              "$ref": "definitions.json#/unique_session_id"},
    "active_probe_results": {
      "properties": {
        "avg_tput_last_probe_duration_Mbps": {
          "maximum":100,
          "minimum": 1,
          "type": "number"
        }
      },
      "type": "object"
    },
    "connection_id": {"$ref": "definitions.json#/connection_id"},
    "init_probe_results": {
      "properties": {
        "lost_probes_percentage": {
          "maximum": 100,
          "minimum": 1,
          "type": "integer"
        },
       "probe_rate_Mbps": {
          "maximum": 100,
          "minimum": 1,
          "type": "number"
        }
      },
      "type": "object"
    }
  },
  "type": "object"
}
```

C.3.18. MX SSID Indication

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "definitions": {},
  "id": "https://example.com/mams/mx_ssid_indication.json",
  "properties": {
    "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
    "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
    "version": {"$ref": "mx_base_def.json#/version_def"},
    "ssid_list": {
      "items": {
        "properties": {
          "ssid_type": {
            "$ref": "definitions.json#/ssid_types"},
          "ssid_id": {
            "type": "integer"
            }
          }
      },
      "type": "array"
    }
  },
  "type": "object"
}
```

C.3.19. MX Measurement Configuration

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "additionalProperties": false,
  "definitions": {
    "meas_conf": {
      "connection_id" : {
        "$ref": "definitions.json#/connection_id"},
      "connection_type": {
        "$ref": "definitions.json#/connection_type"},
      "meas_rep_conf": {
        "items": {
          "$ref": "definitions.json#/meas_report_conf"},
        "type": "array"
      }
    }
  },
  "id": "https://example.com/mams/mx_measurement_conf.json",
  "properties": {
    "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
    "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
    "version": {"$ref": "mx_base_def.json#/version_def"},
    "measurement_configuration": {
      "items": {"$ref": "#definitions/meas_conf"},
      "type": "array"
    }
  },
  "type": "object"
}
```

C.3.20. MX Measurement Report

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "definitions": {},
  "id": "https://example.com/mams/mx_measurement_report.json",
  "properties": {
    "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
    "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
    "version": {"$ref": "mx_base_def.json#/version_def"},
    "unique_session_id": {
      "$ref": "definitions.json#/unique_session_id"},
    "measurement_reports": {
      "items": {
        "properties": {
          "connection_id": {
            "$ref": "definitions.json#/connection_id"},
          "connection_type": {
            "$ref": "definitions.json#/connection_type"},
          "delivery_node_id": {
            "$ref": "definitions.json#/delivery_node_id"},
          "measurements": {
            "items": {
              "properties": {
                "measurement_type": {
                  "$ref": "definitions.json#/meas_report_param"},
                "measurement_value": {
                  "type": "integer"
                }
              },
              "type": "object"
            },
            "type": "array"
          }
        },
        "type": "object"
      },
      "type": "array"
    }
  },
  "type": "object"
}
```

C.3.21. MX Keep-Alive Request

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "additionalProperties": false,
  "id": "https://example.com/mams/mx_keep_alive_req.json",
  "properties": {
    "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
    "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
    "version": {"$ref": "mx_base_def.json#/version_def"},
    "keep_alive_reason": {
      "$ref": "definitions.json#/keep_alive_reason"},
    "unique_session_id": {
      "$ref": "definitions.json#/unique_session_id"},
    "connection_id": {
      "$ref": "definitions.json#/connection_id"},
    "delivery_node_id": {
      "$ref": "definitions.json#/connection_id"}
  },
  "type": "object"
}
```

C.3.22. MX Keep-Alive Response

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "additionalProperties": false,
  "id": "https://example.com/mams/mx_keep_alive_rsp.json",
  "properties": {
    "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
    "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
    "version": {"$ref": "mx_base_def.json#/version_def"},
    "unique_session_id": {
      "$ref": "definitions.json#/unique_session_id"}
  },
  "type": "object"
}
```

C.3.23. MX Session Termination Request

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "additionalProperties": false,
  "id": "https://example.com/mams/mx_keep_alive_req.json",
  "properties": {
    "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
    "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
    "version": {"$ref": "mx_base_def.json#/version_def"},
    "unique_session_id": {
      "$ref": "definitions.json#/unique_session_id"},
    "reason": {
      "enum": [
        "MX_NORMAL_RELEASE",
        "MX_NO_RESPONSE",
        "INTERNAL_ERROR"
      ],
      "type": "string"
    }
  },
  "type": "object"
}
```

C.3.24. MX Session Termination Response

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "additionalProperties": false,
  "id": "https://example.com/mams/mx_session_termination_rsp.json",
  "properties": {
    "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
    "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
    "version": {"$ref": "mx_base_def.json#/version_def"},
    "unique_session_id": {
      "$ref": "definitions.json#/unique_session_id"}
  },
  "type": "object"
}
```

C.3.25. MX Network Analytics Request

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "additionalProperties": false,
  "id": "https://example.com/mams/mx_network_analytics_req.json",
  "properties": {
    "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
    "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
    "version": {"$ref": "mx_base_def.json#/version_def"},
    "unique_session_id": {
      "$ref": "definitions.json#/unique_session_id"},
    "params": {
      "items": {
        "$ref": "definitions.json#/predict_param_name"},
      "type": "array"
    }
  },
  "type": "object"
}
```

C.3.26. MX Network Analytics Response

```
{
  "$schema": "https://json-schema.org/draft-04/schema#",
  "additionalProperties": false,
  "definitions": {
    "ParamPredictions": {
      "param_name": {
        "$ref": "definitions.json#/predict_param_name"},
      "additional_param": {
        "$ref": "definitions.json#/predict_add_param_name"},
      "prediction": {"type": "integer"},
      "likelihood": {"type": "integer"},
      "validity_time": {"type": "integer"}

    },
    "MXAnalyticsList": {
      "connection_id": {
        "$ref": "definitions.json#/connection_id"},
      "connection_type": {
        "$ref": "definitions.json#/connection_type"},
      "predictions": {
        "items": {
          "$ref": "#definitions/ParamPredictions"},
          "type": "array"
      }
    }
  },
  "id": "https://example.com/mams/mx_network_analytics_rsp.json",
  "properties": {
    "message_type": {"$ref": "mx_base_def.json#/message_type_def"},
    "sequence_num": {"$ref": "mx_base_def.json#/sequence_num_def"},
    "version": {"$ref": "mx_base_def.json#/version_def"},
    "param_list": {
      "items": {
        "$ref": "#definitions/MXAnalyticsList"},
        "type": "array"}
  },
  "type": "object"
}
```

C.4. Examples in JSON

C.4.1. MX Discover

```
{
  "version" : "1.0",
  "message_type" : "mx_discover",
  "sequence_num" : 1
}
```

C.4.2. MX System Info

```
{
  "version" : "1.0",
  "message_type" : "mx_system_info",
  "sequence_num" : 2,
  "ncm_connections" : [
    {
      "connection_id" : 0,
      "connection_type" : "LTE",
      "ncm_end_point" : {
        "ip_address" : "192.168.1.10",
        "port" : 1234
      }
    },
    {
      "connection_id" : 1,
      "connection_type" : "Wi-Fi",
      "ncm_end_point" : {
        "ip_address" : "192.168.1.10",
        "port" : 1234
      }
    }
  ]
}
```

C.4.3. MX Capability Request

```
{
  "version" : "1.0",
  "message_type" : "mx_capability_req",
  "sequence_num" : 3,
  "feature_active" : [
    {
      "feature_name" : "lossless_switching",
      "active" : true
    },
    {
      "feature_name" : "fragmentation",
      "active" : false
    }
  ],
  "num_anchor_connections" : 2,
  "anchor_connections" : [
    {
      "connection_id" : 0,
      "connection_type" : "LTE"
    },
    {
      "connection_id" : 1,
      "connection_type" : "Wi-Fi"
    }
  ],
  "num_delivery_connections" : 2,
  "delivery_connections" : [
    {
      "connection_id" : 0,
      "connection_type" : "LTE"
    },
    {
      "connection_id" : 1,
      "connection_type" : "Wi-Fi"
    }
  ],
  "convergence_methods" : [
    {
      "method" : "GMA",
      "supported" : true
    },
    {
      "method" : "MPTCP_Proxy",
      "supported" : false
    }
  ],
  "adaptation_methods" : [
    {
      "method" : "UDP_without_DTLS",
      "supported" : false
    },
    {
      "method" : "UDP_with_DTLS",
      "supported" : false
    },
    {
      "method" : "IPsec",
```

```
      "supported" : true
    },
    {
      "method" : "Client_NAT",
      "supported" : false
    }
  ]
}
```

C.4.4. MX Capability Response

```
{
  "version" : "1.0",
  "message_type" : "mx_capability_rsp",
  "sequence_num" : 3,
  "feature_active" : [
    {
      "feature_name" : "lossless_switching",
      "active" : true
    },
    {
      "feature_name" : "fragmentation",
      "active" : false
    }
  ],
  "num_anchor_connections" : 2,
  "anchor_connections" : [
    {
      "connection_id" : 0,
      "connection_type" : "LTE"
    },
    {
      "connection_id" : 1,
      "connection_type" : "Wi-Fi"
    }
  ],
  "num_delivery_connections" : 2,
  "delivery_connections" : [
    {
      "connection_id" : 0,
      "connection_type" : "LTE"
    },
    {
      "connection_id" : 1,
      "connection_type" : "Wi-Fi"
    }
  ],
  "convergence_methods" : [
    {
      "method" : "GMA",
      "supported" : true
    },
    {
      "method" : "MPTCP_Proxy",
      "supported" : false
    }
  ],
  "adaptation_methods" : [
    {
      "method" : "UDP_without_DTLS",
      "supported" : false
    },
    {
      "method" : "UDP_with_DTLS",
      "supported" : false
    },
    {
      "method" : "IPsec",
```

```
      "supported" : true
    },
    {
      "method" : "Client_NAT",
      "supported" : false
    }
  ],
  "unique_session_id" : {
    "ncm_id" : 110,
    "session_id" : 1111
  }
}
```

C.4.5. MX Capability Acknowledge

```
{
  "version" : "1.0",
  "message_type" : "mx_capability_ack",
  "sequence_num" : 3,
  "unique_session_id" : {
    "ncm_id" : 110,
    "session_id" : 1111
  },
  "capability_ack" : "MX_ACCEPT"
}
```

C.4.6. MX Reconfiguration Request

```
{
  "version" : "1.0",
  "message_type" : "mx_reconf_req",
  "sequence_num" : 4,
  "unique_session_id" : {
    "ncm_id" : 110,
    "session_id" : 1111
  },
  "reconf_action" : "setup",
  "connection_id" : 0,
  "ip_address" : "192.168.110.1",
  "ssid" : "SSID_1",
  "mtu_size" : 1300,
  "connection_status" : "connected",
  "delivery_node_id" : "2A12C"
}
```

C.4.7. MX Reconfiguration Response

```
{
  "version" : "1.0",
  "message_type" : "mx_reconf_rsp",
  "sequence_num" : 4
}
```

C.4.8. MX UP Setup Configuration Request

```
{
  "version": "1.0",
  "message_type": "mx_up_setup_conf_req",
  "sequence_num": 5,
  "num_anchor_connections": 2,
  "anchor_connections": [{
    "connection_id": 1,
    "connection_type": "Wi-Fi",
    "num_active_mx_conf" : 2,
    "convergence_config" : [
    {
      "mx_configuration_id" : 1,
      "convergence_method": "GMA",
      "convergence_method_params": {},
      "num_delivery_connections": 2,
      "delivery_connections": [{
        "connection_id": 0,
        "connection_type": "LTE",
        "adaptation_method": "UDP_without_DTLS",
        "adaptation_method_param": {
          "tunnel_ip_addr": "6.6.6.6",
          "tunnel_end_port": 9999,
          "mx_header_optimization": true
        }
      },
      {
        "connection_id": 1,
        "connection_type": "Wi-Fi"
      }
      ]
    },
    {
      "mx_configuration_id" : 2,
      "convergence_method": "GMA",
      "convergence_method_params": {},
      "num_delivery_connections": 1,
      "delivery_connections": [{
        "connection_id": 0,
        "connection_type": "LTE",
        "adaptation_method": "UDP_without_DTLS",
        "adaptation_method_param": {
          "tunnel_ip_addr": "6.6.6.6",
          "tunnel_end_port": 8877
        }
      }
      ]
    }
    ]
  },
  {
    "connection_id": 0,
    "connection_type": "LTE",
    "udp_port": 8888,
    "num_delivery_connections": 2,
    "delivery_connections": [{
      "connection_id": 0,
      "connection_type": "LTE"
```

```
    },
    {
      "connection_id": 1,
      "connection_type": "Wi-Fi",
      "adaptation_method": "UDP_without_DTLS",
      "adaptation_method_param": {
        "tunnel_ip_addr": "192.168.3.3",
        "tunnel_end_port": "6000"
      }
    }
    ]
  }
  ]
}
```

C.4.9. MX UP Setup Confirmation

```
{
  "version" : "1.0",
  "message_type" : "mx_up_setup_cnf",
  "sequence_num" : 5,
  "unique_session_id" : {
    "ncm_id" : 110,
    "session_id" : 1111
  },
  "probe_param" : {
    "probe_port" : 48700,
    "anchor_conn_id" : 0,
    "mx_configuration_id" : 1
  },
  "num_delivery_conn" : 2,
  "client_params" : [
    {
      "connection_id" : 0,
      "adapt_param" : {
        "udp_adapt_port" : 51000
      }
    },
    {
      "connection_id" : 1,
      "adapt_param" : {
        "udp_adapt_port" : 52000
      }
    }
  ]
}
```

C.4.10. MX Traffic Steering Request

```
{
  "version" : "1.0",
  "message_type" : "mx_traffic_steering_req",
  "sequence_num" : 6,
  "connection_id" : 0,
  "mx_configuration_id" : 1,
  "downlink_delivery" : [
    {
      "connection_id" : 0
    },
    {
      "connection_id" : 1
    }
  ],
  "default_uplink_delivery" : 0,
  "uplink_delivery" : [
    {
      "ul_tft" : {
         "remote_addr_mask" : "10.10.0.0/24",
         "local_addr_mask" : "192.168.0.0/24",
         "protocol_type" :  6,
         "local_port_range" : {
           "start" : 100,
           "end" : 1000
         },
         "remote_port_range" : {
           "start" : 100,
           "end" : 1000
         },
         "traffic_class" : 20,
         "flow_label" : 100
      },
      "conn_list" : [
         {
           "connection_id" : 1
         }
      ]
    },
    {
      "ul_tft" : {
         "remote_addr_mask" : "10.10.0.0/24",
         "local_addr_mask" : "192.168.0.0/24",
         "protocol_type" :  6,
         "local_port_range" : {
           "start" : 2000,
           "end" : 2000
         },
         "remote_port_range" : {
           "start" : 100,
           "end" : 1000
         },
         "traffic_class" : 20,
         "flow_label" : 50
      },
      "conn_list" : [
         {
          "connection_id" : 1
```

```
          }
        ]
      }
    ],
    "feature_active" : [
      {
        "feature_name" : "dl_aggregation",
        "active" : true
      },
      {
        "feature_name" : "ul_aggregation",
        "active" : false
      }
    ]
}
```

C.4.11. MX Traffic Steering Response

```
{
  "version": "1.0",
  "message_type": "mx_traffic_steering_rsp",
  "sequence_num": 6,
  "unique_session_id": {
    "ncm_id": 110,
    "session_id": 1111
  },
  "feature_active": [{
    "feature_name": "lossless_switching",
    "active": true
  },
  {
    "feature_name": "fragmentation",
    "active": false
  }
  ]
}
```

C.4.12. MX Application MADP Association Request

```
{
  "version": "1.0",
  "message_type": "mx_app_madp_assoc_req",
  "sequence_num": 6,
  "unique_session_id": {
    "ncm_id": 110,
    "session_id": 1111
  },
  "app_madp_assoc_list": [{
    "connection_id" : 0,
    "mx_configuration_id" : 1,
    "ul_tft_list": [{
      "protocol_type": 17,
      "local_port_range": {
        "start": 8888,
        "end": 8888
      }
      }],
        "dl_tft_list": [{
          "protocol_type": 17,
          "remote_port_range": {
            "start": 8888,
            "end": 8888
          }
        }]
  }
  ]
}
```

C.4.13. MX Application MADP Association Response

```
{
  "version": "1.0",
  "message_type": "mx_app_madp_assoc_rsp",
  "sequence_num": 6,
  "is_success": true
}
```

C.4.14. MX Path Estimation Request

```
{
  "version" : "1.0",
  "message_type" : "mx_path_est_req",
  "sequence_num" : 7,
  "connection_id" : 0,
  "init_probe_test_duration_ms" : 100,
  "init_probe_test_rate_Mbps" : 10,
  "init_probe_size_bytes" : 1000,
  "init_probe_ack_req" : "yes",
  "active_probe_freq_ms" : 10000,
  "active_probe_size_bytes" : 1000,
  "active_probe_duration_sec" : 10,
  "active_probe_ack_req" : "no"
}
```

C.4.15. MX Path Estimation Results

```
{
  "version" : "1.0",
  "message_type" : "mx_path_est_results",
  "sequence_num" : 8,
  "unique_session_id" : {
    "ncm_id" : 110,
    "session_id" : 1111
  },
  "connection_id" : 0,
  "init_probe_results" : {
    "lost_probes_percentage" : 1,
    "probe_rate_Mbps" : 9.9
  },
  "active_probe_results" : {
    "avg_tput_last_probe_duration_Mbps" : 9.8
  }
}
```

C.4.16. MX SSID Indication

```
{
  "version" : "1.0",
  "message_type" : "mx_ssid_indication",
  "sequence_num" : 9,
  "ssid_list" : [
    {
      "ssid_type" : "ssid",
      "ssid_id" : "SSID_1"
    },
    {
      "ssid_type" : "bssid",
      "ssid_id" : "xxx-yyy"
    }
  ]
}
```

C.4.17. MX Measurement Configuration

```
{
  "version" : "1.0",
  "message_type" : "mx_measurement_conf",
  "sequence_num" : 10,
  "measurement_configuration" : [
  {
    "connection_id" : 0,
    "connection_type" : "Wi-Fi",
    "meas_rep_conf"  : [
      {
        "meas_rep_param" : "WLAN_RSSI",
        "meas_threshold" : {
          "high" : -10,
          "low" : -15
        },
        "meas_period_ms" : 500
      },
      {
        "meas_rep_param" : "WLAN_LOAD",
        "meas_threshold" : {
          "high" : -10,
          "low" : -15
        },
        "meas_period_ms" : 500
      },
      {
        "meas_rep_param" : "EST_UL_TPUT",
        "meas_threshold" : {
          "high" : 100,
          "low" : 30
        },
        "meas_period_ms" : 500
       }
     ]
  },
  {
    "connection_id" : 1,
    "connection_type" : "LTE",
    "meas_rep_conf"  : [
      {
        "meas_rep_param" : "LTE_RSRP",
        "meas_threshold" : {
          "high" : -10,
          "low" : -15
        },
        "meas_period_ms" : 500
      },
      {
        "meas_rep_param" : "LTE_RSRQ",
        "meas_threshold" : {
          "high" : -10,
          "low" : -15
        },
        "meas_period_ms" : 500
      }
    ]
  }
```

```
  ]
}
```

C.4.18. MX Measurement Report

```
{
  "version" : "1.0",
  "message_type" : "mx_measurement_report",
  "sequence_num" : 11,
  "unique_session_id" : {
    "ncm_id" : 110,
    "session_id" : 1111
  },
  "measurement_reports" : [
    {
      "connection_id" : 0,
      "connection_type" : "Wi-Fi",
      "delivery_node_id" : "2021A",
      "measurements"  : [
        {
          "measurement_type" : "WLAN_RSSI",
          "measurement_value" : -12
        },
        {
          "measurement_type" : "UL_TPUT",
          "measurement_value" : 10
        },
        {
          "measurement_type" : "EST_UL_TPUT",
          "measurement_value" : 20
        }
      ]
    },
    {
      "connection_id" : 1,
      "connection_type" : "LTE",
      "delivery_node_id" : "12323",
      "measurements"  : [
        {
          "measurement_type" : "LTE_RSRP",
          "measurement_value" : -12

         },
         {
           "measurement_type" : "LTE_RSRQ",
           "measurement_value" : -12

         }
      ]
    }
  ]
}
```

C.4.19. MX Keep-Alive Request

```
{
  "version" : "1.0",
  "message_type" : "mx_keep_alive_req",
  "sequence_num" : 12,
  "keep_alive_reason" : "Handover",
  "unique_session_id" : {
    "ncm_id" : 110,
    "session_id" : 1111
  },
  "connection_id" : 0,
  "delivery_node_id" : "2021A"
}
```

C.4.20. MX Keep-Alive Response

```
{
  "version" : "1.0",
  "message_type" : "mx_keep_alive_rsp",
  "sequence_num" : 12,
  "unique_session_id" : {
    "ncm_id" : 110,
    "session_id" : 1111
  }
}
```

C.4.21. MX Session Termination Request

```
{
  "version" : "1.0",
  "message_type" : "mx_session_termination_req",
  "sequence_num" : 13,
  "unique_session_id" : {
    "ncm_id" : 110,
    "session_id" : 1111
  },
  "reason" : "MX_NORMAL_RELEASE"
}
```

C.4.22. MX Session Termination Response

```
{
  "version" : "1.0",
  "message_type" : "mx_session_termination_rsp",
  "sequence_num" : 13,
  "unique_session_id" : {
    "ncm_id" : 110,
    "session_id" : 1111
  }
}
```

C.4.23. MX Network Analytics Request

```
{
  "version" : "1.0",
  "message_type" : "mx_network_analytics_req",
  "sequence_num" : 20,
  "unique_session_id" : {
    "ncm_id" : 110,
    "session_id" : 1111
  },
  "params" : [
    "jitter",
    "latency"
  ]
}
```

C.4.24. MX Network Analytics Response

```
{
  "version": "1.0",
  "message_type": "mx_network_analytics_rsp",
  "sequence_num": 20,
  "param_list": [{
    "connection_id": 1,
    "connection_type": "Wi-Fi",
    "predictions": [{
      "param_name": "jitter",
      "prediction": 100,
      "likelihood": 50,
      "validity_time": 10
    },
    {
      "param_name": "latency",
      "prediction": 19,
      "likelihood": 40,
      "validity_time": 10
    }
    ]
  },
  {
    "connection_id": 2,
    "connection_type": "LTE",
    "predictions": [{
      "param_name": "jitter",
      "prediction": 10,
      "likelihood": 80,
      "validity_time": 10
    },
    {
      "param_name": "latency",
      "prediction": 4,
      "likelihood": 60,
      "validity_time": 10
    }
    ]
  }
  ]
}
```

Appendix D. Definition of APIs Provided by the CCM to the Applications at the Client

This section provides an example implementation of the APIs exposed by the CCM to the applications on the client, documented with OpenAPI using Swagger 2.0.

```
{
  "swagger": "2.0",
  "info": {
    "version": "1.0.0",
    "title": "Client Connection Manager (CCM)",
    "description": "API provided by the CCM towards the application
                    on a MAMS client."
  },
  "host": "MAMS.ietf.org",
  "basePath": "/ccm/v1.0",
  "schemes": [
    "https"
  ],
  "consumes": [
    "application/json"
  ],
  "produces": [
    "application/json"
  ],
  "paths": {
    "/capabilities": {
      "get": {
        "description": "This API can be used by an application to
                        request the capabilities of the CCM.",
        "produces": [
          "application/json",
          "text/html"
        ],
        "responses": {
          "200": {
            "description": "OK",
            "schema": {
              "$ref": "#/definitions/capability"
            }
          },
          "default": {
            "description": "unexpected error",
            "schema": {
              "$ref": "#/definitions/errorModel"
            }
          }
        }
      }
    },
    "/app_requirements": {
      "post": {
        "description": "This API is used by the N-MADP to report
                        any types of MAMS user-specific errors to
                        the NCM.",
        "produces": [
          "application/json",
          "text/html"
        ],
        "parameters": [
          {
            "name": "app-requirements",
            "in": "body",
```

```
            "required": true,
            "schema": {
              "$ref": "#/definitions/app-requirements"
            }
          }
        ],
        "responses": {
          "200": {
            "description": "OK"
          },
          "default": {
            "description": "unexpected error",
            "schema": {
              "$ref": "#/definitions/errorModel"
            }
          }
        }
      }
    },
    "/predictive_link_params": {
      "get": {
        "description": "This API is used by applications to get the
                        information about predicted parameters for
                        each delivery connection.",
        "produces": [
          "application/json",
          "text/html"
        ],
        "responses": {
          "200": {
            "description": "OK",
            "schema": {
              "$ref": "#/definitions/link-params"
            }
          },
          "default": {
            "description": "unexpected error",
            "schema": {
              "$ref": "#/definitions/errorModel"
            }
          }
        }
      }
    }
  },
  "definitions": {
    "connection-id": {
      "type": "integer",
      "format": "uint8"
    },
    "connection-type": {
      "enum": [
        "Wi-Fi",
        "5G_NR",
        "MulteFire",
        "LTE"
      ],
      "type": "string"
```

```
    },
    "features": {
      "enum": [
        "lossless_switching",
        "fragmentation",
        "concatenation",
        "uplink_aggregation",
        "downlink_aggregation",
        "measurement"
        "probing"
      ],
      "type": "string"
    },
    "adaptation-methods": {
      "enum": [
        "UDP_without_DTLS",
        "UDP_with_DTLS",
        "IPsec",
        "Client_NAT"
      ],
      "type": "string"
    },
    "convergence-methods": {
      "enum": [
        "GMA",
        "MPTCP_Proxy",
        "GRE_Aggregation_Proxy",
        "MPQUIC"
      ],
      "type": "string"
    },
    "connection": {
      "type": "object",
      "properties": {
        "conn-id": {
          "$ref": "#/definitions/connection-id"
        },
        "conn-type": {
          "$ref": "#/definitions/connection-type"
        }
      }
    },
    "convergence-parameters": {
      "type": "object",
      "properties": {
        "conv-param-name": {
          "type": "string"
        },
        "conv-param-value": {
          "type": "string"
        }
      }
    },
    "convergence-details": {
      "type": "object",
      "properties": {
        "conv-method": {
          "$ref": "#/definitions/convergence-methods"
```

```
        },
        "conv-params": {
          "type": "array",
          "items": {
            "$ref": "#/definitions/convergence-parameters"
          }
        }
      }
    },
    "capability": {
      "type": "object",
      "properties": {
        "connections": {
          "type": "array",
          "items": {
            "$ref": "#/definitions/connection"
          }
        },
        "features": {
          "type": "array",
          "items": {
            "$ref": "#/definitions/features"
          }
        },
        "adapt-methods": {
          "type": "array",
          "items": {
            "$ref": "#/definitions/adaptation-methods"
          }
        },
        "conv-methods": {
          "type": "array",
          "items": {
            "$ref": "#/definitions/convergence-details"
          }
        }
      }
    },
    "qos-param-name": {
      "enum": [
        "jitter",
        "latency",
        "bandwidth"
      ],
      "type": "string"
    },
    "qos-param": {
      "type": "object",
      "properties": {
        "qos-param-name": {
          "$ref": "#/definitions/qos-param-name"
        },
        "qos-param-value": {
          "type": "integer"
        }
      }
    },
    "port-range": {
```

```
        "type": "object",
        "properties": {
          "start": {
            "type": "integer"
          },
          "end": {
            "type": "integer"
          }
        }
      },
      "protocol-type": {
        "type": "integer"
      },
      "stream-features": {
        "type": "object",
        "properties": {
          "proto": {
            "$ref": "#/definitions/protocol-type"
          },
          "port-range": {
            "$ref": "#/definitions/port-range"
          },
          "traffic-qos": {
            "$ref": "#/definitions/qos-param"
          }
        }
      },
      "app-requirements": {
        "type": "object",
        "properties": {
          "num-streams": {
            "type": "integer"
          },
          "stream-feature": {
            "type": "array",
            "items": {
              "$ref": "#/definitions/stream-features"
            }
          }
        }
      },
      "param-name": {
        "enum": [
          "bandwidth",
          "jitter",
          "latency",
          "signal_quality"
        ],
        "type": "string"
      },
      "additional-param-name": {
        "enum": [
          "lte-rsrp",
          "lte-rsrq",
          "nr-rsrp",
          "nr-rsrq",
          "wifi-rssi"
        ],
```

```
      "type": "string"
    },
    "link-parameter": {
      "type": "object",
      "properties": {
        "connection": {
          "$ref": "#/definitions/connection"
        },
        "param": {
          "$ref": "#/definitions/param-name"
        },
        "additional-param": {
          "$ref": "#/definitions/additional-param-name"
        },
        "prediction": {
          "type": "integer"
        },
        "likelihood": {
          "type": "integer"
        },
        "validity_time": {
          "type": "integer"
        }
      }
    },
    "link-params": {
      "type": "array",
      "items": {
        "$ref": "#/definitions/link-parameter"
      }
    },
    "errorModel": {
      "type": "object",
      "description": "Error indication containing the error code and
                      message.",
      "required": [
        "code",
        "message"
      ],
      "properties": {
        "code": {
          "type": "integer",
          "format": "int32"
        },
        "message": {
          "type": "string"
        }
      }
    }
  }
}
```

Appendix E. Implementation Example Using Python for MAMS Client and Server

E.1. Client-Side Implementation

A simple client-side implementation using Python can be as follows:

```
#!/usr/bin/env python
import asyncio
import websockets
import json
import ssl
import time
import sys

context = ssl.SSLContext(ssl.PROTOCOL_TLS)
context.verify_mode = ssl.CERT_REQUIRED
context.set_ciphers("RSA")
context.check_hostname = False
context.load_verify_locations("/home/mecadmin/certs/rootca.pem")

discoverMsg = {'version':'1.0',
'message_type':'mx_discover'}

MXCapabilityRes = {'version':'1.0',
'message_type':'mx_capability_res',
'FeatureActive':[{'feature_name':'fragmentation', 'active':'yes'},
    {'feature_name':'lossless_switching', 'active':'yes'}],
'num_anchor_connections':1,
'anchor_connections':[{'connection_id':0, 'connection_type':'LTE'}],
'num_delivery_connections':1,
'delivery_connections':[{'connection_id':1,
    'connection_type':"Wi-Fi"}],
'convergence_methods':[{'method':'GMA', 'supported':'true'}],
'adaptation_methods':[{'method':'client_nat', 'supported':'false'}]
}

async def hello():
    async with websockets.connect('wss://localhost:8765',
       ssl=context) as websocket:
     try:
      loopFlag=False
      while True:
          await websocket.send(json.dumps(discoverMsg))
          json_message = await websocket.recv()
          message = json.loads(json_message)
          if "message_type" in message.keys():
            print("Received message:{}".format(
               message["message_type"]),
               "version:{}".format(message["version"]))
            if message["message_type"] == "mx_capability_req" :
                await websocket.send(json.dumps(MXCapabilityRes))
                loopFlag=True
                while(loopFlag==True):
                      pass
     except:
          print("Client stopped")

asyncio.get_event_loop().run_until_complete(hello())
```

E.2. Server-Side Implementation

A server-side implementation using Python can be as follows:

```
#!/usr/bin/env python
import asyncio
import websockets
import json
import ssl

ctx = ssl.SSLContext(ssl.PROTOCOL_TLS)
#ctx.set_ciphers("RSA-AES256-SHA")
ctx.load_verify_locations("/home/mecadmin/certs/rootca.pem")
certfile = "/home/mecadmin/certs/server.pem"
keyfile = "/home/mecadmin/certs/serverkey.pem"
ctx.load_cert_chain(certfile, keyfile, password=None)

MXCapabilityReq = {'version':'1.0',
'message_type':'mx_capability_req',
'FeatureActive':[{'feature_name':'fragmentation', 'active':'yes'},
    {'feature_name':'lossless_switching', 'active':'yes'}],
'num_anchor_connections':1,
'anchor_connections':[{'connection_id':0, 'connection_type':'LTE'}],
'num_delivery_connections':1,
'delivery_connections':[{'connection_id':1,
    'connection_type':"Wi-Fi"}],
'convergence_methods':[{'method':'GMA', 'supported':'true'}],
'adaptation_methods':[{'method':'client_nat', 'supported':'false'}]
}

async def hello(websocket, path):
 try:
   while True:
     name = await websocket.recv()
     msg = json.loads(name)
     if "message_type" in msg.keys():
        print("Received message:{}".format(msg["message_type"]),
           "version:{}".format(msg["version"]))
        if msg['message_type'] == 'mx_discover':
           await websocket.send(json.dumps(MXCapabilityReq))

 except:
      print("Client disconnected")

try:
 start_server = websockets.serve(hello, 'localhost', 8765,ssl=ctx)

 asyncio.get_event_loop().run_until_complete(start_server)
 asyncio.get_event_loop().run_forever()
except:
 print("Server stopped")
```

Acknowledgments

This protocol is the outcome of work by many engineers, not just the authors of this document. The people who contributed to this project, listed in alphabetical order by first name, are Barbara Orlandi, Bongho Kim, David Lopez-Perez, Doru Calin, Jonathan Ling, Lohith Nayak, and Michael Scharf.

Contributors

The authors gratefully acknowledge the following additional contributors, in alphabetical order by first name: A Krishna Pramod/Nokia Bell Labs, Hannu Flinck/Nokia Bell Labs, Hema Pentakota/Nokia, Julius Mueller/AT&T, Nurit Sprecher/Nokia, Salil Agarwal/Nokia, Shuping Peng/Huawei, and Subramanian Vasudevan/Nokia Bell Labs. Subramanian Vasudevan has been instrumental in conceptualization and development of solution principles for the MAMS framework. Shuping Peng has been a key contributor in refining the framework and control-plane protocol aspects.

Authors' Addresses

Satish Kanugovi
Nokia Bell Labs
Email: satish.k@nokia-bell-labs.com

Florin Baboescu
Broadcom
Email: florin.baboescu@broadcom.com

Jing Zhu
Intel
Email: jing.z.zhu@intel.com

SungHoon Seo
Korea Telecom
Email: sh.seo@kt.com