INTAREA
Internet-Draft
Intended status: Standards Track
Expires: January 4, 2018

S. Agarwal
S. Kanugovi
Nokia
S. Peng
Huawei
J. Mueller
AT&T

July 03, 2017

Control Plane Message Definitions for Multiple Access Management Services in JSON
draft-agarwal-intarea-mams-protocol-json-00

Abstract

Today, a device can be simultaneously connected to multiple communication networks based on different technology implementations and network architectures like WiFi, LTE, DSL. In such multi-connectivity scenario, it is desirable to combine multiple access networks or select the best one to improve quality of experience for a user and improve overall network utilization and efficiency. This document presents the control plane message definitions and their presentations in JSON, for control plane procedures for configuring the user plane in a multi access management services (MAMS) framework that can be used to flexibly select the combination of uplink and downlink access and core network paths, and user plane treatment for improving network efficiency and enhanced application quality of experience.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 4, 2018.

Table of Contents

1. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Introduction

Multi Access Management Service (MAMS) [I-D.kanugovi-intarea-mams-protocol] is a framework to select and configure network paths when multiple connections can serve a client device. It allows the path selection and configuration to adapt to dynamic network conditions. It is based on principles of user plane interworking that enables the solution to be deployed as an overlay without impacting the underlying networks.

This document presents the JSON based definitions for control plane messages for the MAMS framework. The control plane messages for which have been defined in [I-D.zhu-intarea-mams-control-protocol].

3. Terminology

"Anchor Connection": Refers to the network path from the N-MADP to the Application Server that corresponds to a specific 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.

"Network Connection Manager" (NCM), "Client Connection Manager" (CCM), "Network Multi Access Data Proxy" (N-MADP), and "Client Multi Access Data Proxy" (C-MADP) in this document are to be interpreted as described in [I-D.kanugovi-intarea-mams-protocol].

4. Protocol Specification: General Processing

4.1. Overall Design

Control plane messages for MAMS are exchanged between the CCM and NCM over WebSocket, the content of messages is defined in "Java Script Object Notation" (JSON) format.

4.2. Notation

This document uses JSONString, JSONNumber, and JSONBool to indicate the JSON string, number, and boolean types, respectively. The type JSONValue indicates a JSON value, as specified in Section 3 of [RFC7159].

This document uses an adaptation of the C-style struct notation to define JSON objects. A JSON object consists of name/value pairs. This document refers to each pair as a field. In some context, 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 definition below defines a new type Type4, with three fields named "name1", "name2", and "name3", respectively. The field named "name3" is optional, and the field named "name2" 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 field named "name1", TypeDerived will have a new field named "name1". If TypeBase already has a field named "name1" but with a different type, TypeDerived will have a field named "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.

4.3. Discovery Procedure

4.3.1. MX Discovery Message

This message is the first message sent by CCM to discover the presence of NCM in the network. It contains only the base information as described in Section 5.1 with message_type set as mx_discover.

Following is the representation of the message:

```
object {

} MXDiscover : MXBase;
```

4.4. System Information Procedure

4.4.1. MX System Information Message

This message is sent by NCM to CCM to inform the endpoints that NCM supports for MAMS functionality. In addition to base information (Section 5.1) it contains following information:

a) NCM Connections (described in Section 5.3)

Following is the representation of the message:

```
object {

NCMConnections ncm_connections;

} MXSystemInfo : MXBase;
```

4.5. Capability Exchange Procedure

4.5.1. MX Capability Request

This message is sent by CCM to NCM to indicate the capabilities of the CCM instance available to the NCM indicated in System Info message earlier. In addition to base information (Section 5.1) it contains following information:

(a) Features Activation Status: Described in Section 5.5
(b) Number of anchor connections: Number of anchor connection (towards core) supported by the NCM.
(c) Anchor Connections: Described in sec Section 5.6
(d) Number of Delivery connections: Number of delivery connection (towards access) supported by the NCM.
(e) Delivery Connections: Described in Section 5.7
(f) Convergence Methods: Described in Section 5.9
(g) Adaptation Methods: Described in Section 5.10

Following is the representation of the message:

```
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;
```

4.5.2. MX Capability Response

This message is sent by NCM to CCM to indicate the capabilities of the NCM instance and unique session identifier for CCM. In addition to base information (Section 5.1) it contains following information:

(a) Features Activation Status: Described in Section 5.5
(b) Number of anchor connections: Number of anchor connection (towards core) supported by the NCM.
(c) Anchor Connections: Described in Section 5.6
(d) Number of Delivery connections: Number of delivery connection (towards access) supported by the NCM.
(e) Delivery Connections: Described in Section 5.7
(f) Convergence Methods: Described in Section 5.9
(g) Adaptation Methods: Described in Section 5.10
(h) Unique Session Id: This uniquely identifies the session between CCM and NCM in a network. Described in Section 5.2

Following is the representation of the message:

```
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;
} MXCapabilityRsq : MXBase;
```

4.5.3. MX Capability Acknowledge

This message is sent by CCM to NCM to indicate acceptance of capabilities advertised by NCM in earlier MX Capability Response message. In addition to base information (Section 5.1) it contains following information:

(a) Unique Session Id: Same identifier as provided in MX Capability RSP. Described in Section 5.2.
(b) Capability Acknowledgement: Either Accept or Reject of the capabilities sent by CCM. Can take either "MX_ACCEPT" or "MX_REJECT" as acceptable values.

Following is the representation of the message:

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

4.6. User Plane Configuration Procedure

4.6.1. MX User Plane Configuration Request

This message is sent by NCM to CCM to configure the user plane for MAMS. In addition to base information (Section 5.1) it contains following information:

(a) Number of Anchor Connection: Number of anchor connections supported by NCM.
(b) Setup of Anchor Connections: Described in Section 5.11.

Following is the representation of the message:

```
object {
```

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

4.6.2. MX User Plane Configuration Confirmation

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

(a) Unique Session Id: Same identifier as provided in MX Capability RSP. Described in Section 5.2.
(b) MX Probe Parameters (included if probing is supported)

    (1) Probe Port: UDP port for accepting probe message.
(c) For each delivery connection following is required:

    (1) Connection ID: Delivery connection id supported by UE.
    (2) Client Adaptation Layer Parameters: If UDP adaptation layer is in use then the UDP port to be used at C-MADP side.

Following is the representation of the message:

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

Where ProbeParam is defined as following:

```
object {
JSONNumber probe_port;
} ProbeParam;
```

Where ClientParam is defined as following:

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

Where AdaptationParam is defined as following:

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

4.7. Reconfiguration Procedure

4.7.1. Reconfiguration Request

This message is sent by CCM to NCM in case of reconfiguration of any the connections from user equipment's side. In addition to base information (Section 5.1) it contains following information:

(a) Unique Session Id: Identifier for CCM-NCM association Section 5.2.
(b) Reconfiguration Action: Type of reconfiguration action can be one of "setup", "release" or "modify".
(c) Connection Id: Connection Id for which the reconfiguration is taking place.
(d) IP address: IP address in case of setup and modify type of reconfiguration.
(e) SSID: If the connection type is WiFi, in that case the SSID the UE has attached to is contained in this parameter.
(f) MTU of connection: MTU of the delivery path that is calculated at the UE for use by NCM to configure fragmentation and concatenation procedures at N-MADP.
(g) Connection Status: This parameter informs if the connection is currently "disabled", "enabled" or "connected". Default: "connected".
(h) Delivery Node Id: Identity of the node to which the client is attached. ECGI in case of LTE and WiFi AP Id or MAC address in case of WiFi.

Following is the representation of the message:

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

4.7.2. Reconfiguration Response

This message is sent by NCM to CCM as a confirmation towards reconfiguration requirement after taking the reconfiguration into use and contains only the base information (as defined in Section 5.1).

Following is the representation of the message:]

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

4.8. Path Estimation Procedure

4.8.1. Path Estimation Request

This message is sent by NCM towards CCM to configure the CCM to send path estimation reports. In addition to base information (Section 5.1) it contains 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. [TBD: Range of values]
(c) Init Probe Test Rate: Initial testing rate in Mega Bits per Second. [TBD: Range of values]
(d) Init Probe Size: Size of each packet for initial probe in Bytes. [TBD: Range of values]
(e) Init Probe Ack: If an acknowledgement for probe is required. [Possible values: "yes", "no"]
(f) Active Probe Frequency: Frequency in milliseconds at which the active probes shall be sent. [TBD: Range of values]
(g) Active Probe Size: Size of the active probe in Bytes. [TBD: Range of values]
(h) Active Probe Duration: Duration in seconds for which the active probe shall be performed. [TBD. Range of values]
(i) Active Probe Ack. If an acknowledgement for probe is required. [Possible values: "yes", "no"]

Following is the representation of the message:

```
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;
```

4.8.2. Path Estimation Report

This message is sent by CCM to NCM as report to the probe estimation configured by NCM. In addition to base information (Section 5.1) it contains following information:

(a) Unique Session Id: Same identifier as provided in MX Capability RSP. Described in Section 5.2.
(b) Connection Id: Id of the connection for which the path estimation report is required.
(c) Init Probe Results: Defined in section Section 5.12.
(d) Active Probe Results: Defined in section Section 5.13.

Following is the representation of the message:

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

4.9. Traffic Steering Procedure

4.9.1. Traffic Steering Request

This message is sent by NCM to CCM for enabling traffic steering at delivery side in uplink and downlink configuration. In addition to base information (Section 5.1) it contains following information:

(a) Connection id: Defines the anchor connection number for which the traffic steering is getting define.
(b) Downlink Delivery: Defined in Section 5.14.
(c) Uplink Delivery: Defined in Section 5.15.
(d) Features Activated: Defined in Section 5.5.

Following is the representation of the message:

```
object {
JSONNumber connection_id;
DLDelivery downlink_delivery;
ULDelivery uplink_delivery;
FeaturesActive feature_activation;
} MXTraffiSteeringReq : MXBase;
```

4.9.2. Traffic Steering Response

This message is response to Traffic Steering request from CCM to NCM. In addition to base information (Section 5.1) it contains following information:

(a) Unique Session Id: Same identifier as provided in MX Capability RSP. Described in Section 5.2.
(b) Features Activated: Defined in Section 5.5.

Following is the representation of the message:

```
object {
UniqueSessionId unique_session_id;
FeaturesActive feature_activation;
} MXTraffiSteeringResp : MXBase;
```

4.10. SSID Indication

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

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

Following is the representation of the message:

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

Where each SSID is defined as following:

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

4.11. Measurements

4.11.1. Measurement Configuration

This message is sent from NCM to CCM to configure the period measurement reporting at CCM. The message contains a list of measurement configuration with each element containing 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", "wifi", "5g-nr" etc.
(c) Measurement Report Configuration: Actual report configuration based on the connection type, as defined in Section 5.17

Following is the representation of the message:

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

Where each measurement MeasReportConf is represented by following:

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

4.11.2. Measurement Report

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

(a) Unique Session Id: Same identifier as provided in MX Capability RSP. Described in Section 5.2.
(b) Measurement report for each delivery connection is measured by the client device as defined in Section 5.18.

Following is the representation of the message:

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

4.12. Keep Alive

4.12.1. Keep Alive Request

A Keep Alive Request message can be sent from either NCM or CCM on expiry of MAMS_KEEP_ALIVE timer or a handover event. This request shall be responded by the peer with Keep Alive Response. In case of no response from peer the MAMS connection shall be assumed to be

broken and new connection shall be established again by CCM by sending MX Discover messages.

In addition to the base information it cantains following information:

(a) Keep Alive Reason: Reason for sending this message, can be "Timeout" or "Handover".
(b) Unique Session Id: Identifier for CCM-NCM association Section 5.2.
(c) Connection Id: Connection id for which handover is detected, in case the reason is "Handover".
(d) Delivery Node Id: The target delivery node id (ECGI or WiFi Access Point Id/MAC) to which the handover is executed.

Following is the representation of the message:

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

4.12.2. Keep Alive Response

On receiving Keep Alive Request from peer, NCM/CCM shall immediately respond with a Keep Alive Response message on the same delivery path from where the request arrived. In addition to base information it contains the unique session identifier for the CCM-NCM association (defined in Section 5.2)

Following is the representation of the message:

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

4.13. Session Termination Procedure

4.13.1. Session Terminate Request

In the event where NCM or CCM can no longer handle MAMS for any reason then it can send MX session termination request to the peer. In addition to base information it contains Unique Session Id and reason for termination, this can be "MX_NORMAL_RELEASE", "MX_NO_RESPONSE" or "INTERNAL_ERROR".

Following is the representation of the message:

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

4.13.2. Session Terminate Response

On reception of MX session termination request from peer, NCM/CCM shall respond with MX Session Termination Response on the same delivery path where the request arrived and clean the MAMS related resources and settings. CCM shall re-initiate a new session with MX Discover messages again.

Following is the representation of the message:

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

5. Protocol Specification: Data Types

5.1. MXBase

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

(a) Version: Version of MAMS in used
(b) Message Type: Message type being sent with following as valid values:

  (a) "mx_discover"
  (b) "mx_system_info"
  (c) "mx_capability_req"
  (d) "mx_capability_resp"
  (e) mx_capability_ack"
  (f) "mx_up_setup_conf_req"
  (g) "mx_up_setup_cnf"
  (h) "mx_reconf_req"
  (i) "mx_reconf_rsp"
  (j) "mx_path_est_req"
  (k) "mx_path_est_results"
  (l) "mx_traffic_steering_req"
  (m) "mx_traffic_steering_rsp"
  (n) "mx_ssid_indication"
  (o) "mx_keep_alive_req"

(p) "mx_keep_alive_rsp"
(q) "mx_measurement_conf"
(r) "mx_measurement_report"
(s) "mx_session_termination_req"
(t) "mx_session_termination_resp"

(c) Sequence Number: Sequence number to uniquely identify a transaction of message exchange, e.g. MX Capability REQ/RSP/ ACK.

Following is the representation of this data type:

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

5.2. Unique Session Id

This data type defines the unique session id between a CCM and NCM entity, it contains a NCM id which is unique in the network and a session id allocated by NCM for that session. On reception, of discovery message if the session is existing then the old session id is returned in System Info message otherwise NCM allocates a new session id to the CCM and sends in response with System Info message.

Following is the representation of this data type:

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

5.3. NCM Connections

This data type defines the connection available at NCM for MAMS connectivity towards the User Equipment. It contains a list of NCM connections available where each connection has following information:

(a) Connection Information: As defined in Section 5.4
(b) NCM End Point information: This contains IP Address and Port exposed by NCM end point for CCM.

Following is the representation of this data type:

```
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;
```

5.4. Connection Information

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

(a) Connection Id: Number indicating the connection can be 0,1,2 and 3.
(b) Connection type: Type of connect can be "Wi-Fi, "5G NR", "Multi-Fire" and "LTE".

The two are considered a mapping like 0-"Wi-Fi", 1-"5G NR", 2-"Multi-Fire" and 3-"LTE".

Following is the representation of this data type:

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

5.5. Features Activation Status

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

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

    (a) "lossless_switching"
    (b) "fragmentation"
    (c) "concatenation"
    (d) "uplink_aggregation"
    (e) "downlink_aggregation"
    (f) "measurement"

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

Following is the representation of this data type:

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

where FeatureInfo is defined as:

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

5.6. Anchor Connections

This data type contains the list of Connection Information (Section 5.4) that are supported at anchor (core) side.

Following is the representation of this data type:

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

5.7. Delivery Connections

This data type contains the list of Connection Information (Section 5.4) that are supported at delivery (access) side.

Following is the representation of this data type:

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

5.8. Method Support

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

(a) Method: Name of the method.
(b) Supported: Whether the method named above is supported or not. Possible values are "true" and "false".

Following is the representation of this data type:

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

5.9. Convergence Methods

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

"Trailer_Based"
"MPTCP_Proxy"
"GRE_Aggregation_Proxy"

Following is the representation of this data type:

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

5.10. Adaptation Methods

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

"UDP_without_DTLS"
"UDP_with_DTLS"
"IPSec"
"Client_NAT"

Following is the representation of this data type:

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

5.11. Setup of Anchor Connections

This data type defines the setup configuration for each of the anchor connection that is required at the user equipment side. Each anchor connection contains following information:

(a) Convergence Method: Converge method selected, has to be one of the supported convergence method as listed in section Section 5.9.
(b) Convergence Method Parameters: Described in section Section 5.11.1

(c) Number of Delivery Connections: Number of delivery connections (access side) that are supported for this anchor connection.
(d) Setup Delivery Connections: Described in section Section 5.11.2.

Following is the representation of this data type:

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

where each Anchor connection configuration is defined as following:

```
object {
JSONString convergence_method;
ConvergenceMethodParam convergence_method_params;
JSONNumber num_delivery_connections;
SetupDeliveryConns delivery_connections;
} SetupAnchorConn : ConnectionInfo;
```

5.11.1. Convergence Method Parameters

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

(a) Proxy IP: IP Address of proxy that is provided by Convergence Method selected.
(b) Proxy Port: Port of the proxy that is provided by Convergence Method selected.

Following in the representation of this data type:

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

5.11.2. Setup Delivery Connections

This is the list of delivery connections and their parameters to be configured at the user equipment. Each delivery connection defined by its connection information (Section 5.4) contains optionally following:

(a) Adaptation Method: Selected adaptation method name, this shall be one of the names as listed in Section 5.10.
(b) Adaptation Method Parameters: Depending on the adaptation method one or more of the following parameters shall be provided.

(a) Tunnel IP address
(b) Tunnel Port number
(c) Shared Secret

Following in the representation of this data type:

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

where each Setup Delivery Connection consists of following:

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

where Adaptation Method Param is defined as:

```
object {
JSONString tunnel_ip_addr;
JSONString tunnel_end_port;
JSONString shared_secret;
} AdaptationMethodParam;
```

5.12. Init Probe Results

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

(a) Lost Probes: Percentage or probes lost.
(b) Prode Delay: Average delay of probe message in microseconds.
(c) Probe Rate: Probe rate achieved in Mega Bits per second.

Following in the representation of this data type:

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

5.13. Active Probe Results

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

(a) Average Probe Throughput: Average active probe throughput achieved in Mega Bits per second.

Following in the representation of this data type:

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

5.14. Downlink Delivery

This data type defines the list of connections which are enabled in delivery side to be used in downlink direction.

Following in the representation of this data type:

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

5.15. Uplink Delivery

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

The uplink delivery consists of multiple uplink delivery entities, where each entity consists of a traffic flow template (TFT) Section 5.16 and list of connection ids in uplink where traffic qualifying for such traffic flow template can be redirected.

Following in the representation of this data type:

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

Where each uplink delivery entity consists of following data type:

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

5.16. Traffic Flow Template

Traffic flow template follows in general guidelines specified in 3GPP TS 23.060.

Traffic flow template in MAMS consists of one or more of following:

(a) Remote Address and Mask: IP address and subnet for remote addresses represented in 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 IP packet. e.g. UDP, TCP etc. Default: 255.
(d) Local Port Range: Range of ports for local ports for which the flow template is applicable. Default: Start=0, End=65535.
(e) Remote Port Range: Range of ports for remote ports for which the 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.

Following in the representation of this data type:

```
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 following:

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

5.17. Measurement Report Configuration

This data type defines the configuration done by NCM towards CCM for reporting measurement events.

(a) Measurement Report Parameter: Parameter which shall be measured and reported. This is dependent on the connection type:
    (a) For connection type "wifi" allowed measurement parameters are "WLAN_RSSI", "WLAN_LOAD", "UL_TPUT", "DL_TPUT", "EST_UL_TPUT" and "EST_DL_TPUT".
    (b) For connection type "lte" allowed measurement parameters are "LTE_RSRP", "LTE_RSRQ", "UL_TPUT" and "DL_TPUT".
(b) Threshold: High and Low threshold for reporting.

(c) Period: Period for reporting in milliseconds.

Following is the representation of this data type:

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

Where Threshold is defined as following:

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

5.18. Measurement Report

This data type defines the measurements reported by CCM for each access network measured. This type contains the connection information, delivery node id which identifies the cell (ECGI) or the WiFI Access Point Id or MAC address (or equivalent identifier in other technologies) and the actual measurement performed by CCM in the last measurement period.

Following is the representation of this data type:

```
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 measurement type and value. The exact type and value are defined on a per delivery type and defined in Section 5.17.

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

6. Schemas in JSON

6.1. MX Base Schema

```
   {
     "$schema": "http://json-schema.org/draft-04/schema#",
     "definitions": {
           "message_type_def": {
         "enum": [
           "mx_discover",
                   "mx_system_info",
                   "mx_capability_req",
                   "mx_capability_resp",
                   "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_resp"
           ],
               "type": "string"
       },
       "sequence_num_def": {
         "minimum": 1,
         "type": "integer"
       },
       "version_def": {
         "type": "string"
       }
      },
     "id": "http://www.ietf.org/mams/mx_base_def.json"
   }
```

6.2. MX Definitions

```
{
        "$schema": "http://json-schema.org/draft-04/schema#",
        "definitions": {
```

```
                "adapt_method": {
                        "enum": [
                                "UDP_without_DTLS",
                                "UDP_with_DTLS",
                                "IPSec",
                                "Client_NAT"
                        ],
                        "type": "string"
                },
                "conv_method": {
                        "enum": [
                                "Trailer_Based",
                                "MPTCP_Proxy",
                                "GRE_Aggregation_Proxy"
                        ],
                        "type": "string"
                },
                "supported": {
                        "type": "boolean"
                },
                "active": {
                        "type": "boolean"
                },
                "connection_id": {
                        "type": "integer"
                },
                "feature_name": {
                        "enum": [
                                "lossless_switching",
                                "fragmentation",
                                "concatenation",
                                "uplink_aggregation",
                                "downlink_aggregation",
                                "measurement"
                        ],
                        "type": "string"
                },
                "connection_type": {
                        "enum": [
                                "wi-fi",
                                "5g-nr",
                                "multi-fire",
                                "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_acknowledgement" : {
                "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"
                        ],
                "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" : { "$ref" : "#definitions/ip_add
r_mask" },
                "local_addr_mask" : {  "$ref" : "#definitions/ip_add
r_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"
                        }
                },
                "client_param" : {
                        "connection_id" : {
                                "type" : "integer"
                        },
                        "adapt_param" : {
                            "type" : {"$ref" : "#definitions/adaptation_para
m" }
                        }
                }
        },
        "id": "http://www.ietf.org/mams/definitions.json"
}
```

6.3. MX Discover

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "additionalProperties": false,
  "id": "http://www.ietf.org/mams/mx_discover.json",
  "properties": {
    "message_type" : {"$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"
}
```

6.4. MX System Update

```
{
        "$schema": "http://json-schema.org/draft-04/schema#",
        "additionalProperties": false,
        "id": "http://www.ietf.org/mams/mx_system_info.json",
        "properties": {
                "message_type": {
                        "$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"
}
```

6.5. MX Capability Request

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "additionalProperties": false,
  "id": "http://www.ietf.org/mams/mx_capability_req.json",
  "properties": {
    "message_type" : {"$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"
}
```

6.6. MX Capability Response

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "additionalProperties": false,
  "id": "http://www.ietf.org/mams/mx_capability_resp.json",
  "properties": {
        "message_type" : {"$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"
}
```

6.7. MX Capability Ack

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "definitions": {},
    "id": "http://www.ietf.org/mams/mx_capability_ack.json",
    "properties": {
                "message_type" : {"$ref": "mx_base_def.json#/message_type_de
f"},
                "sequence_num" : {"$ref": "mx_base_def.json#/sequence_num_de
f"},
                "version" : {"$ref": "mx_base_def.json#/version_def"},
                "unique_session_id" : { "$ref": "definitions.json#/unique_se
ssion_id" },
        "capability_ack": {"$ref" : "definitions.json#/capability_acknowledg
ement"}
    },
    "type": "object"
}
```

6.8. MX Reconfiguration Request

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "definitions": {},
  "id": "http://www.ietf.org/mams/mx_reconf_req.json",
  "properties": {
        "message_type" : {"$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"
}
```

6.9. MX Reconfiguration Response

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "definitions": {},
  "id": "http://www.ietf.org/mams/mx_reconf_rsp.json",
  "properties": {
    "message_type" : {"$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"
}
```

6.10. MX UP Setup Configuration

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "additionalProperties": false,
  "definitions": {},
  "id": "http://www.ietf.org/mams/mx_up_setup_conf_req.json",
  "properties": {
        "message_type" : {"$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"
},
          "convergence_method": { "$ref" : "definitions.json#/conv_method" }
,
          "convergence_method_params": {
            "properties": {
              "proxy_ip": { "$ref" : "definitions.json#/ip_address" },
              "proxy_port": {"$ref" : "definitions.json#/port" }
            },
            "type": "object"
          },
                "num_delivery_connections": {
            "type": "integer"
          },
          "delivery_connections": {
            "items":{ "$ref" : "definitions.json#/connection" },
                        "type": "array"
          }
         },
        "type": "object"
      },
      "type": "array"
    }
  },
  "type": "object"
}
```

6.11. MX UP Setup Confirmation

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "definitions": {},
  "id": "http://www.ietf.org/mams/mx_up_setup_cnf.json",
  "properties": {
    "message_type" : {"$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"
}
```

6.12. MX Traffic Steering Request

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "definitions": {
        "conn_list" : {
                "items" : { "$ref" : "definitions.json#/connection_id" },
                "type": "array"
        },
        "ul_delivery" : {
                "ul_tft" : { "$ref" : "definitions.json#/traffic_flow_templa
te"},
                "connection_list" : { "$ref" : "#definitions/conn_list" }
        }
  },
  "id": "http://www.ietf.org/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" },
    "downlink_delivery": {
      "items": { "$ref" : "definitions.json#/connection_id" },
      "type": "array"
    },
    "feature_activation": {
      "items": {"$ref" : "definitions.json#/feature_status" },
      "type": "array"
    },
    "uplink_delivery": {
      "items": { "$ref" : "#definitions/ul_delivery" },
      "type": "array"
    }
  },
  "type": "object"
}
```

6.13. MX Traffic Steering Response

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "definitions": {},
  "id": "http://example.com/example.json",
  "properties": {
    "message_type" : {"$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" : { "$ref": "definitions.json#/unique_session_id
" },
        "feature_activation": {
      "items": {"$ref" : "definitions.json#/feature_status" },
      "type": "array"
    }
  },
  "type": "object"
}
```

6.14. MX Path Estimation Request

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "definitions": {},
  "id": "http://www.ietf.org/mams/mx_path_est_req.json",
  "properties": {
        "message_type" : {"$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"
}
```

6.15. MX Path Estimation Report

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "definitions": {},
  "id": "http://www.ietf.org/mams/mx_path_est_results.json",
  "properties": {
        "message_type" : {"$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"
}
```

6.16. MX SSID Indication

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "definitions": {},
    "id": "http://www.ietf.org/mams/mx_ssid_indication.json",
    "properties": {
                "message_type" : {"$ref": "mx_base_def.json#/message_type_de
f"},
                "sequence_num" : {"$ref": "mx_base_def.json#/sequence_num_de
f"},
                "version" : {"$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"
}
```

6.17. MX Measurements Configuration

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "additionalProperties": false,
  "definitions" : {
        "meas_conf" : {
                "connection_id" : { "$ref" : "definitions.json#/connection_i
d" },
                "connection_type" : { "$ref" : "definitions.json#/connection
_type" },
                "meas_rep_conf" : {
                        "items" : { "$ref" : "definitions.json#/meas_report_
conf" },
                        "type" : "array"
                }
        }
  },
  "id": "http://www.ietf.org/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"
}
```

6.18. MX Measurements Report

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "definitions": {},
    "id": "http://www.ietf.org/mams/mx_measurement_report.json",
    "properties": {
                "message_type" : {"$ref": "mx_base_def.json#/message_type_de
f"},
                "sequence_num" : {"$ref": "mx_base_def.json#/sequence_num_de
f"},
                "version" : {"$ref": "mx_base_def.json#/version_def"},
                "unique_session_id" : { "$ref": "definitions.json#/unique_se
ssion_id" },
        "measurment_reports": {
            "items": {
                "properties": {
                    "connection_id": {
                        "$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_pa
ram"
                                },
                                "measurement_value": {
                                    "type": "integer"
                                }
                            },
                            "type": "object"
                        },
                        "type": "array"
                    }
                },
                "type": "object"
            },
            "type": "array"
        }
    },
    "type": "object"
}
```

6.19. MX Keep Alive Request

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "additionalProperties": false,
  "id": "http://www.ietf.org/mams/mx_keep_alive_req.json",
  "properties": {
    "message_type" : {"$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"
}
```

6.20. MX Keep Alive Response

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "additionalProperties": false,
  "id": "http://www.ietf.org/mams/mx_keep_alive_rsp.json",
  "properties": {
    "message_type" : {"$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"
}
```

6.21. MX Session Termination Request

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "additionalProperties": false,
  "id": "http://www.ietf.org/mams/mx_keep_alive_req.json",
  "properties": {
    "message_type" : {"$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"
}
```

6.22. MX Session Termination Response

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "additionalProperties": false,
  "id": "http://www.ietf.org/mams/mx_session_termination_resp.json",
  "properties": {
    "message_type" : {"$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"
}
```

7. Examples in JSON

7.1. MX Discover

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

7.2. MX System Update

```
  {
    "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" : "wifi",
                                  "ncm_end_point" : {
                                          "ip_address" : "192.168.1.10",
                                          "port" : 1234
                                  }
                          }
          ]
  }
```

7.3. 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" : "wifi"
                }
    ],
    "num_delivery_connections" : 2,
    "delivery_connections" : [
                {
                        "connection_id" : 0,
                        "connection_type" : "lte"
                },
                {
                        "connection_id" : 1,
                        "connection_type" : "wifi"
                }
    ],
    "convergence_methods" : [
                {
                        "method" : "Trailer_Based",
                        "supported" : true
                },
                {
                        "method" : "MPTCP_Proxy",
                        "supported" : false
                }
    ],
    "adaptation_methods" : [
                {
                        "method" : "UDP_without_DTLS",
                        "supported" : false
                },
                {
                        "method" : "UDP_with_TLS",
                        "supported" : false
                },
                {
                        "method" : "IPSec",
                        "supported" : true
                },
                {
                        "method" : "Client_NAT",
                        "supported" : false
                }
    ]
  }
```

7.4. MX Capability Response

```
   {
     "version" : "1.0",
     "message_type" : "mx_capability_resp",
     "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" : "wifi"
                    }
     ],
     "num_delivery_connections" : 2,
     "delivery_connections" : [
                    {
                            "connection_id" : 0,
                            "connection_type" : "lte"
                    },
                    {
                            "connection_id" : 1,
                            "connection_type" : "wifi"
                    }
     ],
     "convergence_methods" : [
                    {
                            "method" : "Trailer_Based",
                            "supported" : true
                    },
                    {
                            "method" : "MPTCP_Proxy",
                            "supported" : false
                    }
     ],
```

```
     "adaptation_methods" : [
                    {
                            "method" : "UDP_without_DTLS",
                            "supported" : false
                    },
                    {
                            "method" : "UDP_with_TLS",
                            "supported" : false
                    },
                    {
                            "method" : "IPSec",
                            "supported" : true
                    },
                    {
                            "method" : "Client_NAT",
                            "supported" : false
                    }
     ],
     "unique_session_id" : {
                    "ncm_id" : 110,
                    "session_id" : 1111
     }
   }
```

7.5. MX Capacity Ack

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

7.6. 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"
}
```

7.7. MX Reconfiguration Response

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

7.8. MX UP Setup Configuration Request

```
{
  "version" : "1.0",
  "message_type" : "mx_up_setup_conf_req",
  "sequence_num" : 5,
  "num_anchor_connections" : 1,
  "anchor_connections" : [
                {
                        "connection_id" : 0,
                        "connection_type" : "lte",
                        "convergence_method" : "MPTCP_Proxy",
                        "convergence_method_params" : {
                        "proxy_ip" : "192.168.1.1",
                        "proxy_port" : 1234
                        },
                        "num_delivery_connections" : 2,
                        "delivery_connections" : [
                                {
                                        "connection_id" : 0,
                                        "connection_type" : "lte"
                                },
                                {
                                        "connection_id" : 1,
                                        "connection_type" : "wifi",
                                        "adaptation_method" : "IPSec",
                                        "adaptation_method_param" : {
                                                "tunnel_ip_addr" : "192.168.
3.3",
                                                "tunnel_end_port" : "NA",
                                                "shared_secret" : "abcdefg12
345"
                                        }
                                }
                        ]
                }
        ]
}
```

7.9. 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
  },
  "num_delivery_conn" : 2,
  "client_params" : [
                {
                        "connection_id" : 0,
                        "adapt_param" : {
                                "udp_adapt_port" : 51000
                        }
                },
                {
                        "connection_id" : 1,
                        "adapt_param" : {
                                "udp_adapt_port" : 52000
                        }
                }
  ]
}
```

7.10. MX Traffic Steering Request

```
{
  "version" : "1.0",
  "message_type" : "mx_traffic_steering_req",
  "sequence_num" : 6,
  "connection_id" : 0,
  "downlink_delivery" : [
        {
                "connection_id" : 0
        },
        {
                "connection_id" : 1
        }
  ],
  "uplink_delivery" : [
        {
                "ul_tft" : {
                        "remote_addr_mask" : "10.10.0.0/24",
                        "local_addr_mask" : "192.168.0.0/24",
```

```
                        "protocol_type" :  6,
                        "loca_port_range" : {
                          "start" : 100,
                          "end" : 1000
                        },
                        "remote_port_range" : {
                                "start" : 100,
                                "end" : 1000
                        },
                        "traffic_class" : 20,
                        "flow_label" : 100
                },
                "conn_list" : [
                        {
                                "connection_id" : 0
                        },
                        {
                                "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" : 0
                        },
                        {
                                "connection_id" : 1
                        }
                ]
        }
    ],
    "feature_activation" : [
```

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

7.11. 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_activation" : [
                {
                        "feature_name" : "lossless_switching",
                        "active" : true
                },
                {
                        "feature_name" : "fragmentation",
                        "active" : false
                }
        ]
}
```

7.12. 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"
}
```

7.13. 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
  }
}
```

7.14. 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"
                }
        ]
}
```

7.15. MX Measurements 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
                        }
                ]
        }
    ]
   }
```

7.16. MX Measurements Report

```
{
        "version" : "1.0",
        "message_type" : "mx_measurement_report",
        "sequence_num" : 11,
        "unique_session_id" : {
                "ncm_id" : 110,
                "session_id" : 1111
    },
        "measurment_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

                        }
                ]
        }
  ]
}
```

7.17. 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"
}
```

7.18. 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
  }
}
```

7.19. 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"
}
```

7.20. MX Session Termination Response

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

8. Contributing Authors

The editors gratefully acknowledge the following additional contributors in alphabetical order: A Krishna Pramod/Nokia, Hannu Flinck/Nokia, Jing Zhu/Intel, Nurit Sprecher/Nokia.

9. References

9.1. 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, <http://www.rfc-editor.org/info/rfc2119>.

[SDO-3GPP.23.060] 3GPP, "General Packet Radio Service (GPRS); Service description; Stage 2", 3GPP TS 23.060 3.17.0, December 2006.

9.2. Informative References

[I-D.kanugovi-intarea-mams-protocol] Kanugovi, S., Vasudevan, S., Baboescu, F., Zhu, J., Peng, S., Mueller, J., and S. Seo, "Multiple Access Management Services", draft-kanugovi-intarea-mams-protocol-04 (work in progress), March 2017.

[I-D.zhu-intarea-mams-control-protocol] Kanugovi, S., Vasudevan, S., Zhu, J., Baboescu, F., Peng, S., and S. Seo, "Control Plane Protocols and Procedures for Multiple Access Management Services", draft-zhu-intarea-mams-control-protocol-01 (work in progress), March 2017.

[RFC7159] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", RFC 7159, DOI 10.17487/RFC7159, March 2014, <http://www.rfc-editor.org/info/rfc7159>.

[RFC7285] Alimi, R., Ed., Penno, R., Ed., Yang, Y., Ed., Kiesel, S., Previdi, S., Roome, W., Shalunov, S., and R. Woundy, "Application-Layer Traffic Optimization (ALTO) Protocol", RFC 7285, DOI 10.17487/RFC7285, September 2014, <http://www.rfc-editor.org/info/rfc7285>.

Appendix A. Implementation Example

A simple client side implementation using python can be as following:

```
#!/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’:"wifi"}],
’convergence_methods’:[{’method’:’trailer_based’, ’supported’:’true’}],
’adaptation_methods’:[{’method’:’client_nat’, ’supported’:’false’}]
}

async def hello():
    async with websockets.connect(’wss://localhost:8765’, ssl=context) as we
bsocket:
     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("Recieved message:{}".format(message["message_type"]),"ver
sion:{}".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())
```

A server client side implementation using python can be as following:

```
#!/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':"wifi"}],
'convergence_methods':[{'method':'trailer_based', '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("Recieved 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")
```

Authors' Addresses

Salil Agarwal
Nokia

Email: salil.agarwal@nokia.com

Satish Kanugovi
Nokia

Email: satish.k@nokia.com

Shuping Peng
Huawei

Email: pengshuping@huawei.com

Julius Mueller
AT&T

Email: jm169k@att.com